Continuous optimization

Michael Lindsey

Latest update: April 14, 2026

Contents

Part I
Preliminaries

1 What is optimization?

In attempting to answer this question, we will define a scope for the course and highlight some key themes as we go.

A general optimization problem can be written in the form:

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

Here f is called the **objective** function, and C defines the **constraints**. In practice, the **feasible set** C is usually specified via a list of equality and/or inequality constraints. (In semidefinite programming, C can be defined additionally in terms of linear matrix inequalities involving matrix variables.) A point satisfying $x \in C$ is called **feasible**.

1.1 Scope of the course

In this class, *we will be concerned with* ***continuous*** *optimization problems* where $C \subset \mathbb{R}^n$ for some n[1]. If simply $C = \mathbb{R}^n$, then we say that the problem is **unconstrained**. Unconstrained optimization will be our first focus. Constrained optimization is in general more difficult, and the theory of **duality** plays a key role.

Our general formulation conceals boundless diversity of optimization problems. There is no 'one algorithm' that achieves all desiderata for all problems. Often one faces serious tradeoffs between accuracy and efficiency/scalability, dictated by the problem dimension as well as the dimension and structure of the constraints.

Indeed, there are *so many* topics we will not even touch! A quick glance at the reference texts (some of which are quite lengthy!) will make this clear. But this course intends to hit most of the *big ideas* that are practically relevant. By the end of the course, when faced with a continuous optimization problem, you should have an intuition about what one can hope to achieve for this problem, and with what kind of computational cost for a given target accuracy. You should also have the tools you need to code up your own optimizer from scratch (though this may not always be necessary).

At the risk of some oversimplification it is useful to keep in mind two regimes:

- **Low-dimensional problems:** direct linear algebra is cheap (recall that solving $n \times n$ linear systems in general costs $O(n^3)$ operations via Gaussian elimination, cf. Math 128A). In this regime, we expect algorithms that converge in a few iterations, each of which has a cost that scales steeply with dimension.
  - Think **Newton's method** for unconstrained problems; **interior point methods** and **augmented Lagrangian methods** for constrained problems.

[1] Note that it may be awkward to phrase some given optimization problem in terms of a single variable $x \in \mathbb{R}^n$. In particular it may require viewing x as a composite variable via a concatenation and/or flattening of primitive matrix/tensor variables. It is worth keeping in mind that doing so may obscure some special structure that is useful for your particular problem. But there is also a lot of power in designing general approaches.

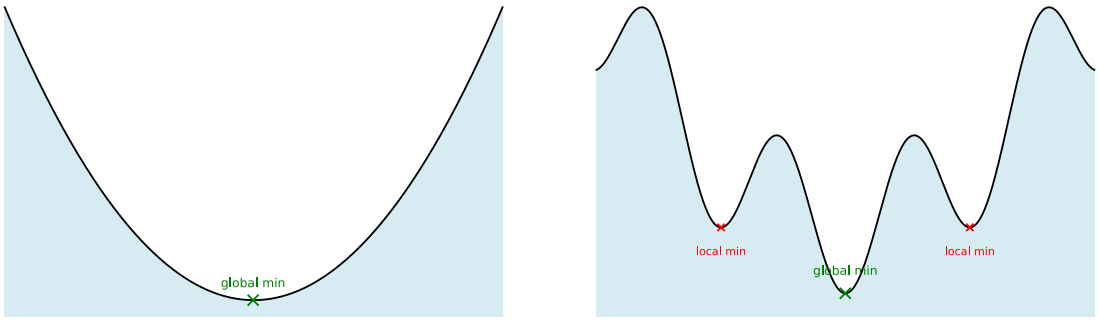

Figure 1.1: Convex (left) vs. nonconvex (right) optimization landscapes.

- **High-dimensional problems:** direct linear algebra is too expensive, and possibly anything beyond $O(n)$ computational scaling is intractable. In this regime, we expect algorithms that converge slowly over many iterations, each of which is cheap.
  - Think **gradient descent** for unconstrained problems. For constrained problems, the recipe is less clear. **(Inexact) augmented Lagrangian methods** define a general framework, but often there is special structure to consider.

The cutoff between these two regimes is murky to define, and there are many methods that aim for somewhere in between. For example, **quasi-Newton methods** (e.g., **L-BFGS**) attempt to extend the magic of Newton's method to higher-dimensional problems, by using some kind of **low-rankness** to accelerate the linear algebra.

Indeed, for general unconstrained optimization problems of at least moderate dimension in many settings, L-BFGS has been a fairly standard choice for a long time. But in deep learning, a new suite of variations on gradient descent (e.g., especially, Adam) have taken over. It is worth noting that modern deep learning essentially always involves training with **stochastic gradients** and moreover that success consists not merely of lowering the objective but also of finding neural network weights that generalize well beyond the training set. Understanding how the training dynamics influence 'generalization error' is a major topic of contemporary inquiry, but we won't focus on it in this class. That said, hopefully we will get to a discussion of stochastic gradient methods from a purely optimization-theoretic point of view.

1.2 When is optimization tractable?

Now when is it possible to 'win' at our goal of optimization? Let us highlight the key difficulties standing in the way.

First of all, it is important to draw a distinction between **global** and **local** optimization. Global optimization, as a general problem, is NP-complete, and it is not reasonable to hope for a general efficient solver. In the continuous setting, the major fundamental difficulty is the possible existence of **many local optimizers**. (For discrete optimization problems, it is less clear in general how to define local optimality, but a similar difficulty is often present.) Local optimization (i.e., finding some local optimizer) is typically a tractable goal, and special structure (namely, **convexity**) can guarantee that there is one local optimizer, which is the global optimizer.

We probably do not have time to study algorithms for nonconvex global optimization. They are by necessity based on some heuristics for exploring many local optima, cf. simulated annealing, basin hopping, and 'just trying a bunch of random initializations.'

Secondly, we need to assume some kind of regularity in the objective function. Intuitively, we should not be getting our hopes up about finding local optima of extremely rough functions, and the analysis needs to rule out certain pathologies. It is also important to remember that optimization algorithms are typically iterative algorithms, which furnish a solution as a limit of a sequence. The **rate** of convergence of this sequence is very important, even if we know that the sequence is guaranteed to converge. Some quantitative notion of **smoothness**, sometimes together with quantitative notions of convexity, is important for controlling this rate.

In summary, we highlight two key, quantifiable properties which we will study before getting to any optimization aglorithms.

- **Convexity:** rules out the possibility of local optimizers that are not global optimizers. Convexity is a property of both the objective function and the feasible set. When both hold, the problem is a **convex optimization problem**. For many global convergence results, we will assume convexity somehow. However, even for nonconvex problems, convexity typically holds effectively near a local optimizer. Thus the assumption of convexity is not as severe as it may seem, if we care about studying the local convergence properties on general problems.
- **Smoothness:** an appropriate quantification of smoothness will appear in our quantitative rates of convergence. Later we will also see how certain types of structured non-smoothness can be accommodated for some problems.

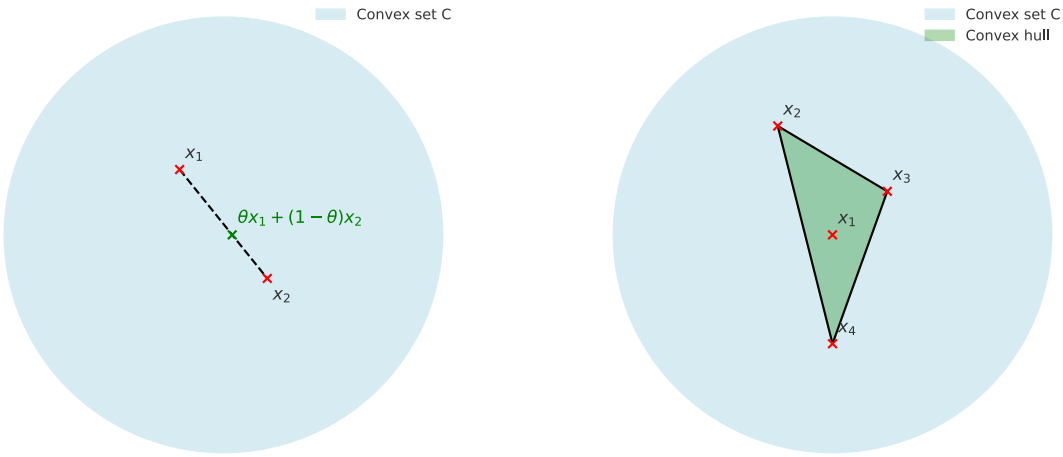

Figure 2.1: **Left:** illustration of the definition of convexity of a set C. **Right:** depiction of the convex hull of a set $\{x_1, x_2, x_3, x_4\} \subset C$.

2 Convexity

2.1 Convex sets

We begin with the definition of a convex set.

Definition 2.1 (Convex sets)**.** A set $C \subset \mathbb{R}^n$ is called *convex* if for all $x, y \in C$ and all $\theta \in [0, 1]$, the point $(1 - \theta)x + \theta y$ also lies in A. (See Figure 2.1.)

Exercise 2.2 (Intersections of convex sets)**.** Show that the intersection of two convex sets is convex. Show that the intersection of an arbitrary finite number of convex sets is convex.

We have the following immediate consequence of the definition which furnishes an equivalent definition.

Lemma 2.3 (Convex combinations)**.** *A set $C \subset \mathbb{R}^n$ is convex if and only if for any choice of $x_1, \ldots, x_k \in C$ for any positive integer k and $\theta_1, \ldots, \theta_k \geq 0$ such that $\sum_{i=1}^k \theta_i = 1$, it holds that $\sum_{i=1}^k \theta_i x_i \in C$.*

Remark 2.4. Such a linear combination with nonnegative coefficients that sum to 1 is called a **convex combination**. The collection of all convex combinations of elements from some set S is called the **convex hull** of S. See Figure 2.1. The convex hull of S is the smallest convex set containing S (in the sense that it is a subset of any other convex set containing S). It is a good exercise to prove this claim.

Proof. The reverse direction is immediate (taking $k = 2$), so we prove the forward direction.

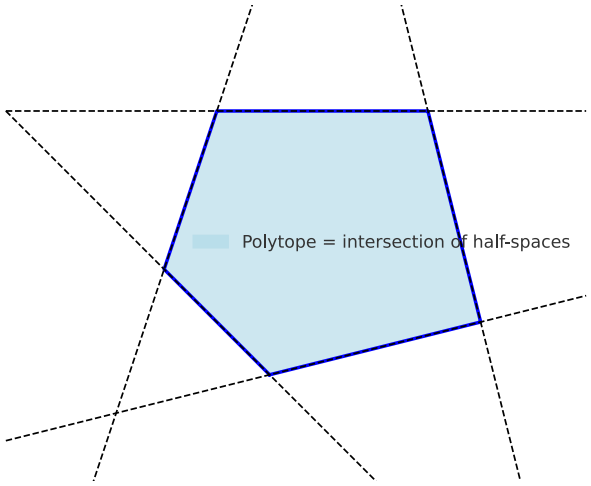

Figure 2.2: Convex polytope as an intersection of half-spaces.

Thus assume C is a convex set and let $x_1, \ldots, x_k \in C$ and $\theta_1, \ldots, \theta_k \geq 0$ such that $\sum_{i=1}^k \theta_i = 1$. We want to show that $\overline{x} := \sum_{i=1}^k \theta_i x_i \in C$. Without loss of generality assume that $\theta_i > 0$ for all $i = 1, \ldots, k$ (otherwise we can just omit some terms from our sums and reduce k accordingly).

We can proceed by induction on k. Let $\theta = \theta_k$ and $y = x_k$, so then

$$1 - \theta = \sum_{i=1}^{k-1} \theta_i.$$

Also define

$$x = \frac{\sum_{i=1}^{k-1} \theta_i x_i}{1-\theta} = \sum_{i=1}^{k-1} \frac{\theta_i}{1-\theta} x_i \tag{2.1}$$

so that we can rewrite

$$\overline{x} = (1-\theta)x + \theta y. \tag{2.2}$$

By the inductive hypothesis, $x \in C$ since we have written it in (2.1) as a convex combination of $x_1, \ldots, x_{k-1}$. Then by the definition of convexity applied to (2.2), we know that $\overline{x} \in C$ as well. □

Exercise 2.5 (Half-spaces are convex)**.** For a given $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$, the set $\{x \in \mathbb{R}^n : a \cdot x \leq b\}$ is called a **half-space**. Show that half-spaces are convex.

In light of this exercise and the fact (Exercise 2.2) that intersections of convex sets are convex, we can understand the convexity of 'convex polytopes' as following from a presentation as an intersection of half-spaces. See Figure 2.2.

Exercise 2.6 (Affine subspaces are convex)**.** Show that any subspace of $\mathbb{R}^n$ is a convex set. Moreover, we can define an **affine subspace** as a translation of a subspace by a fixed vector. More concretely, any affine subspace can be presented as $\{Ay + b : y \in \mathbb{R}^m\}$ where $A \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$. (This is the translation by b of the column space of A.) Show that any affine subspace is a convex set.

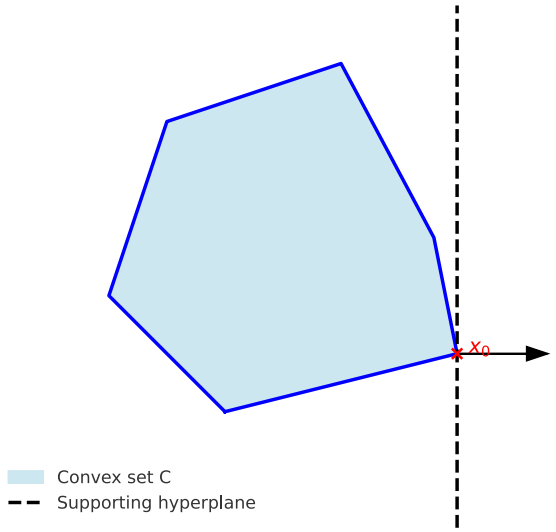

Figure 2.3: A supporting hyperplane.

In fact, the next exercise proves something more general: the image of a convex set under an affine-linear transformation is a convex set.

Exercise 2.7 (Affine-linear transformations preserve convexity)**.** An **affine-linear transformation** is a map $F : \mathbb{R}^m \to \mathbb{R}^n$of the form $F(y) = Ay + b$ where $A \in \mathbb{R}^{n\times m}$ and $b \in \mathbb{R}^n$. Show that if $C \subset \mathbb{R}^m$ is a convex set, then $F(C) := \{F(y) : y \in C\}$ is a convex subset of $\mathbb{R}^n$.

In the next section, we will give further examples of how convex sets can be furnished from sublevel sets of convex functions. We close this section with a classic theorem which conveys further intuition about the nature of convex sets. We omit the proof since it is not essential for us.

Theorem 2.8 (Supporting hyperplane theorem)**.** *Let $C \subset \mathbb{R}^n$ be a convex set and let $x_0 \in \partial C$ (the boundary of C). Then there exists $a \in \mathbb{R}^n$ nonzero such that $a \cdot x \leq a \cdot x_0$ for all $x \in C$. In other words, for any $x_0 \in \partial C$ there exists a 'supporting half-space' $\{x \in \mathbb{R}^n : a \cdot x \leq a \cdot x_0\}$ which contains all of C and which contains x_0 on its boundary (the supporting hyperplane). Conversely, any closed convex set is equal to the intersection of all of its supporting half-spaces.*

2.2 Convex functions

2.2.1 Definition and basic properties

Definition 2.9 (Convex functions)**.** A function $f : C \to \mathbb{R}$ defined on a convex domain $C \subset \mathbb{R}^n$ is called convex if for every $x, y \in C$ and $\theta \in [0, 1]$, it holds that

$$f\left((1-\theta)x + \theta y\right) \leq (1-\theta)f(x) + \theta f(y). \tag{2.3}$$

(See Figure 2.4 for illustration.) We say that f is *strictly* convex if the inequality 2.3 holds strictly for all $\theta \in (0, 1)$ whenever $x \neq y$.

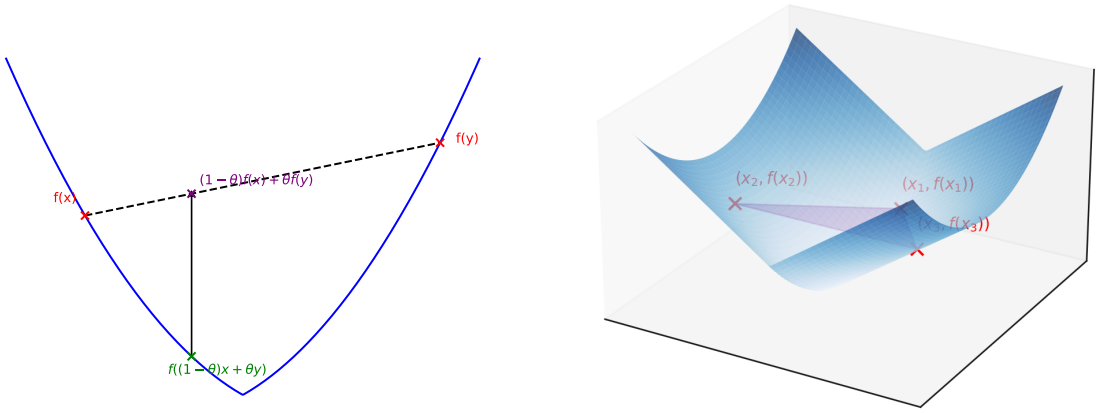

Figure 2.4: **Left:** illustration of the definition of convexity of a function f. **Right:** depiction of the equivalent definition in terms of arbitrary convex combinations.

In general we won't talk much about *concave* functions, which are simply those functions f such that $-f$ is convex. 'Concave maximization' problems can be formulated as convex minimization problems by negating the objective.

The following exercise establishes an equivalent definition.

Exercise 2.10 (Convex functions in terms of convex combinations)**.** A function $f : C \to \mathbb{R}$ defined on a convex domain $C \subset \mathbb{R}^n$ is convex if and only if for any choice $x_1, \dots, x_k \in C$ for any positive integer k and $\theta_1, \dots, \theta_k \geq 0$ such that $\sum_{i=1}^k \theta_i = 1$, it holds that

$$f\left(\sum_{i=1}^k \theta_i x_i\right) \leq \sum_{i=1}^k \theta_i f(x_i). \tag{2.4}$$

(See Figure 2.4 for illustration.) **Hint:** Mimic the proof of Lemma 2.3.

Exercise 2.11 (Sums and nonnegative scaling preserve convexity)**.** Suppose $f, g : C \to \mathbb{R}$ are convex functions on a convex domain $C \subset \mathbb{R}^n$ and $\alpha \geq 0$ is a nonnegative scalar. Then $f + g$ and αf are both convex functions. More generally, suppose that $f_1, \dots, f_k : C \to \mathbb{R}$ are convex and $\alpha_1, \dots, \alpha_k \geq 0$ are nonnegative scalars. Then $\sum_{i=1}^k \alpha_i f_i$ is convex.

Exercise 2.12 (Composition with affine-linear transformations preserves convexity)**.** Let $f : \mathbb{R}^n \to \mathbb{R}$ be convex on $\mathbb{R}^n$, $A \in \mathbb{R}^{n \times m}$ and $b \in \mathbb{R}^n$. Then the map $g : \mathbb{R}^m \to \mathbb{R}$ defined by

$$g(x) = f(Ax + b)$$

is convex.

Remark 2.13. This result can be viewed as saying that the restriction of a convex function to any affine subspace is itself a convex function. As an additional exercise, try to make this precise!

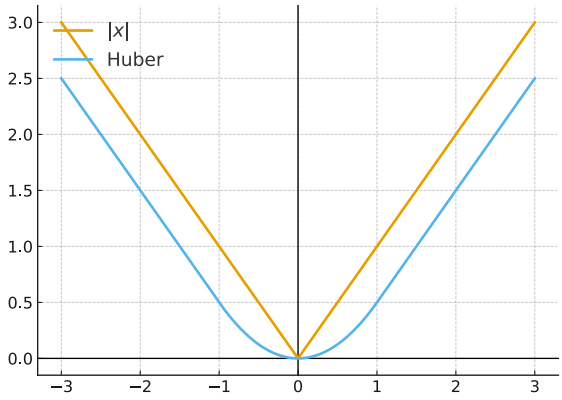

Figure 2.5: Illustration of convex functions that are not strictly convex.

Exercise 2.14 (Pointwise maxima of convex functions are convex)**.** Suppose $f, g : C \to \mathbb{R}$ are convex functions on a convex domain $C \subset \mathbb{R}^n$. Then the pointwise maximum $h(x) := \max(f(x), g(x))$ is convex on C. More generally, suppose that $f_1, \ldots, f_k : C \to \mathbb{R}$ are convex. Then the pointwise maximum $h(x) := \max_{i=1,\ldots,k} f_i(x)$ is convex on C.

Exercise 2.15 (Sublevel sets of convex functions are convex)**.** The α-**sublevel set** of a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as $\{x \in \mathbb{R}^n : f(x) \leq \alpha\}$. **(a)** Show that if f is a convex function, then any sublevel set of f is convex (as a set). **(b)** Show that the converse is not true.

2.2.2 Strict convexity and strong convexity

Definition 2.9 of convex functions introduces a notion of strict convexity that rules out the possibility of 'flat segments' in the graph of the function. For example $f(x) = |x|$ is not strictly convex. The Huber function

$$f(x) = \begin{cases} \frac{1}{2}x^2, & |x| \leq 1 \\ |x| - \frac{1}{2}, & |x| > 1, \end{cases}$$

which is by contrast differentiable, also fails strict convexity. See Figure 2.5.

The notion of strict convexity is unfortunately *not quantitative*, and for optimization analysis it is often important to assume something stronger. Here we use $\|\cdot\|$ to denote the Euclidean norm

$$\|z\|^2 = z \cdot z = \sum_{i=1}^{n} z_i^2, \quad z \in \mathbb{R}^n,$$

and often we adopt this convention unless specified otherwise. It is always useful to think of elements of $\mathbb{R}^n$ as column vectors so that we can view the dot product as a $(1 \times n)$ by $(n \times 1)$ matrix-matrix product: $u \cdot v = u^\top v$.

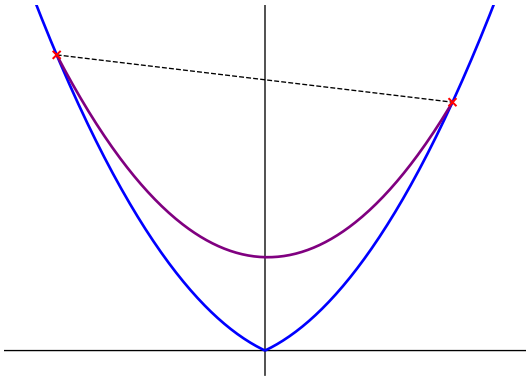

Figure 2.6: Illustration of the definition of strong convexity. The blue curve is the graph of f while the purple curve indicates the quadratic on the right-hand side of (2.5).

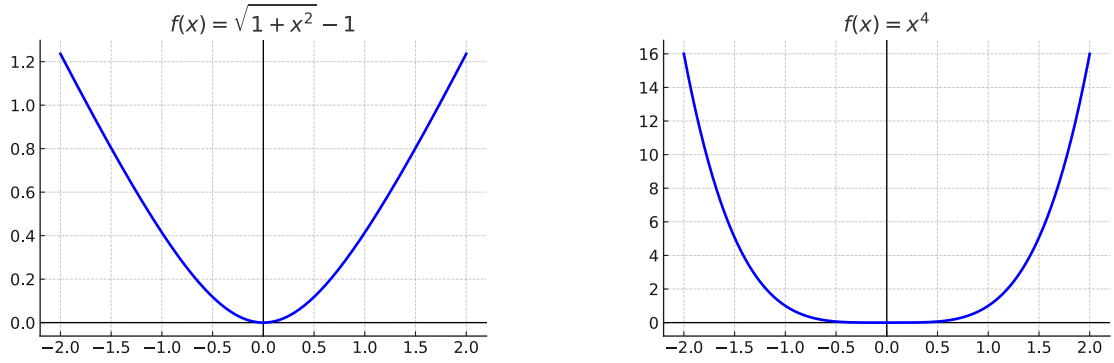

Figure 2.7: Two functions which are strictly convex but not strongly convex. Can a function be strongly convex but not differentiable?

Definition 2.16 (Strong convexity)**.** A function $f : C \to \mathbb{R}$ defined on a convex domain $C \subset \mathbb{R}^n$ is called μ-strongly convex for $\mu \geq 0$ if for every $x, y \in C$ and $\theta \in [0, 1]$, it holds that

$$f\left((1-\theta)x + \theta y\right) \leq (1-\theta)f(x) + \theta f(y) - \frac{\mu}{2}\theta(1-\theta)\|x-y\|^2. \tag{2.5}$$

(See Figure 2.6 for illustration.) Note that strong convexity with $\mu = 0$ recovers the ordinary definition of convexity. When we just say that a function is strongly convex, we mean that it is μ-strongly convex for some $\mu > 0$.

In Figure 2.7 we show several examples of strictly convex functions on $\mathbb{R}$ that fail to be strongly convex.

2.2.3 Continuity

The definition of convexity itself automatically imposes some regularity on convex functions, without any additional assumptions. The key tool for understanding this point is the following lemma, which establishes that the slopes of secant lines only increase as we move either of their endpoints from left to right.

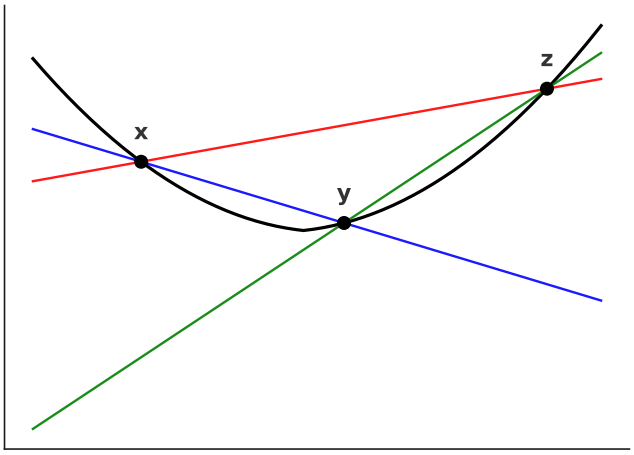

Figure 2.8: Illustration of Lemma 2.17. The slope of the blue line is less than the slope of the red line, which is less than the slope of the green line.

Lemma 2.17 (Secant slopes are increasing)**.** *Suppose $f : \mathbb{R} \to \mathbb{R}$ is convex and $x < y < z$. Then*

$$\frac{f(y) - f(x)}{y - x} \leq \frac{f(z) - f(x)}{z - x} \leq \frac{f(z) - f(y)}{z - y}.$$

Proof. Write y as a convex combination of x and z:

$$y = (1 - \theta)x + \theta z, \quad \text{where } \theta := \frac{y - x}{z - x} \in (0, 1).$$

Then apply convexity to deduce that

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

Subtracting $f(x)$ from both sides and simplifying, we obtain

$$f(y) - f(x) \leq \theta\,[f(z) - f(x)].$$

Recall that we defined $\theta = (y - x)/(z - x)$, so dividing both sides by $y - x$ yields

$$\frac{f(y) - f(x)}{y - x} \leq \frac{f(z) - f(x)}{z - x}.$$

This proves the first half of the lemma.

To deduce the other inequality, apply the first half of the lemma to the convex function $g(t) := f(-t)$, with points $u := -z$, $v := -y$, $w := -x$ (so $u < v < w$) in place of x, y, z:

$$\frac{g(v) - g(u)}{v - u} \leq \frac{g(w) - g(u)}{w - u}.$$

Unpacking everything in terms of our original function f and points x, y, z, we see that

$$\frac{f(y) - f(z)}{y - z} \leq \frac{f(x) - f(z)}{x - z}.$$

Multiplying both sides by -1 proves the second half of the lemma. □

In fact this result implies that any convex function on $\mathbb{R}^n$ is continuous (even locally Lipschitz). The case $n = 1$ is most immediate to show, but with additional work the general case can be deduced from the one-dimensional case. We will only sketch the proof.

Theorem 2.18 (Convex functions are continuous)**.** *A convex function $f : \mathbb{R}^n \to \mathbb{R}$ is continuous.*

Proof. Hand-waving at the board! □

Remark 2.19. In fact it is possible to show that a convex function is continuous on any open set (or even on any 'relatively open' set within an affine subspace). But somewhat annoyingly, convex functions may fail to be continuous, e.g., on closed sets. Try to think of a counterexample! In convex analysis, the standard framework for dealing with convex functions allows them to take values in the 'extended reals,' i.e., $\mathbb{R} \cup \{+\infty\}$. The definition of convexity still makes sense for such functions when interpreted suitably. Then typically one restricts attention to lower semi-continuous' (l.s.c.) functions, which have closed sublevel sets. For our purposes, we will avoid such technicalities for the most part. We will typically work with convex functions that are either continuous on closed domains or defined only on an open domain (due to an asymptote at the boundary). When the domain is $\mathbb{R}^n$ as in unconstrained optimization, there is nothing to worry about.

2.2.4 Characterization in terms of the gradient

Under the assumption of differentiability, we have the following alternative and important characterization of convexity.

Theorem 2.20. *Suppose that $f : C \to \mathbb{R}$ is differentiable on an open convex domain C. Then f is convex if and only if*

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0)$$

for all $x, x_0 \in C$.

Remark 2.21. We can interpret the theorem as saying that convex functions always lie above their linear approximation about any point. We can argue for this graphically via Lemma 2.17.

We will actually prove a stronger theorem, phrased in terms of a general strong convexity parameter. (The simpler case is recovered by setting $\mu = 0$.) See Figure 2.9 for an illustration of both results.

Theorem 2.22 (Convexity in terms of the gradient)**.** *Suppose that $f : C \to \mathbb{R}$ is differentiable on an open convex domain C. Then f is μ-strongly convex if and only if*

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{\mu}{2}\|x - x_0\|^2 \tag{2.6}$$

for all $x, x_0 \in C$.

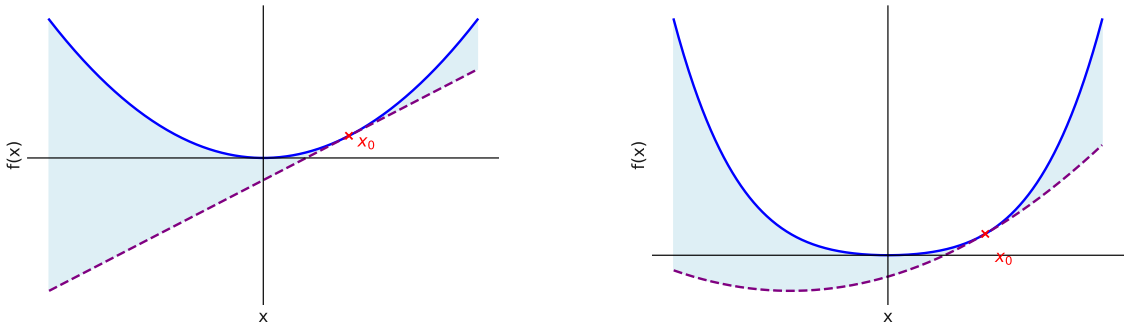

Figure 2.9: Illustration of Theorems 2.20 and 2.22.

Remark. In fact our arguments will show that if f is μ-strongly convex, then (2.6) holds for all $x \in C$ and any $x_0 \in C$ where f is differentiable. (In other words, we only require differentiability at the point x_0 to get this global inequality.)

Proof. First we prove the reverse direction. Suppose (2.6) holds for all $x, x_0 \in C$. Let $x, y \in C$ and let $\theta \in [0,1]$. Define $w = (1-\theta)x + \theta y$ for convenience. To establish that f is μ-strongly convex, we want to show that

$$f(w) \leq (1-\theta)\, f(x) + \theta f(y) - \frac{\mu}{2}\theta(1-\theta)\,\|y-x\|^2. \tag{2.7}$$

Now apply (2.6) with y, w in the place of x, x_0 to obtain

$$\begin{aligned} f(y) &\geq f(w) + \nabla f(w)\cdot(y-w) + \frac{\mu}{2}\|y-w\|^2 \\ &= f(w) + (1-\theta)\,\nabla f(w)\cdot(y-x) + \frac{\mu}{2}(1-\theta)^2\|y-x\|^2, \end{aligned} \tag{2.8}$$

where in the last line we have used the fact that $y - w = (1-\theta)\,(y-x)$.

Repeating the same argument with x, w in the place of y, w and using $x - w = -\theta(y-x)$, we obtain

$$f(x) \geq f(w) - \theta\,\nabla f(w)\cdot(y-x) + \frac{\mu}{2}\theta^2\|y-x\|^2. \tag{2.9}$$

Then by taking a convex combination of our inequalities (2.8) and (2.9), we see that

$$(1-\theta)\, f(x) + \theta f(y) \geq f(w) + \frac{\mu}{2}\theta(1-\theta)\,\|y-x\|^2,$$

which implies (2.7) and completes the reverse direction of the proof.

Now we turn to the proof of the forward direction. Let f be μ-strongly convex and let $x, x_0 \in C$. We want to show that (2.6) holds. For $\theta \in [0,1]$, let

$$x_\theta = x_0 + \theta(x - x_0) = (1-\theta)x_0 + \theta x,$$

and apply strong convexity:

$$f(x_\theta) \leq (1-\theta)f(x_0) + \theta f(x) - \frac{\mu}{2}\theta(1-\theta)\|x-x_0\|^2.$$

Rearrange to solve for $f(x)$:

$$\begin{aligned} f(x) &\geq \frac{f(x_\theta) - (1-\theta)f(x_0) + \frac{\mu}{2}\theta(1-\theta)\|x - x_0\|^2}{\theta} \\ &= f(x_0) + \frac{\mu}{2}(1-\theta)\|x - x_0\|^2 + \frac{f(x_\theta) - f(x_0)}{\theta}. \end{aligned}$$

But then taking the limit as $\theta \to 0$, we obtain

$$\begin{aligned} f(x) &\geq f(x_0) + \frac{\mu}{2}\|x - x_0\|^2 + \frac{d}{d\theta}\Big|_{\theta=0} f(x_0 + \theta(x - x_0)) \\ &= f(x_0) + \frac{\mu}{2}\|x - x_0\|^2 + \nabla f(x_0) \cdot (x - x_0), \end{aligned}$$

where in the last line we have used the chain rule. Evidently the last line is equivalent to (2.6) by rearranging terms. □

2.2.5 Characterization in terms of the Hessian

In a typical single-variable calculus course, we say that a univariate function $f : \mathbb{R} \to \mathbb{R}$ is convex if $f''(x) \geq 0$ for all x. But note that our definition in this course does not require *any* derivatives, let alone two! Fortunately, when a function *is* twice differentiable, we will see that our definition is equivalent to this simpler one in the univariate setting. Moreover, we will see how the condition on the second derivatives generalizes to functions of several variables.

Definition 2.23 (Positive (semi)definite matrices)**.** We say that an $n \times n$ matrix $A \in \mathbb{R}^{n \times n}$ is positive semidefinite (PSD) if (1) A is symmetric (i.e., $A^\top = A$) and (2) $z^\top A z \geq 0$ for all $z \in \mathbb{R}^n$. If the inequality holds strictly for all $z \neq 0$, we say that the matrix is positive definite (PD).

Exercise 2.24. Show that if $A \in \mathbb{R}^{m \times n}$ is an arbitrary (possibly rectangular) $m \times n$ matrix, then $A^\top A \in \mathbb{R}^{n \times n}$ is PSD.

Positive definiteness of a matrix can be interpreted in terms of the eigenvalues via the *spectral theorem*. Recall that the spectral theorem says that symmetric matrices can be 'orthogonally diagonalized,' i.e., admit an orthonormal basis of eigenvectors. In compact linear algebra notation, this means that if A is symmetric, we can write

$$A = U \Lambda U^\top,$$

where $\Lambda = \operatorname{diag}(\lambda_1, \ldots, \lambda_n)$ and the columns of $U = [u_1, \ldots, u_n]$ form an orthonormal basis of eigenvectors $u_1, \ldots, u_n$ satisfying $A u_i = \lambda_i u_i$ for $i = 1, \ldots, n$.

You may recall the standard notation for diagonalization $A = PDP^{-1}$, where the columns of P are the eigenvectors of A and D is a diagonal matrix with corresponding eigenvalues on the diagonal. In the case where $U = P$ has orthonormal columns, U is an orthogonal matrix, so $U^\top = U^{-1}$, and the notations coincide.

It is useful sometimes to write the spectral theorem in a different (but equivalent way):

$$A = \sum_{i=1}^{n} \lambda_i u_i u_i^\top.$$

This format follows from the following more general fact, by viewing $A = (U\Lambda)U^\top$ and recalling that multiplication from the right by a diagonal matrix Λ rescales the columns suitably by the diagonal entries of Λ:

Exercise 2.25. Show that if $A = XY^\top$ where $X = [x_1, \ldots, x_k] \in \mathbb{R}^{m\times k}$ and $Y = [y_1, \ldots, y_k] \in \mathbb{R}^{n\times k}$, then we can write $A = \sum_{i=1}^k x_i y_i^\top$.

Then we can prove the following characterization of positive definiteness in terms of the eigenvalues.

Lemma 2.26. *A symmetric matrix $A \in \mathbb{R}^{n\times n}$ is positive semidefinite if and only if all of its eigenvalues are nonnegative. It is positive definite if and only if all of its eigenvalues are positive.*

Proof. We will just prove the first statement and leave the second as an exercise. First we will prove the forward direction, so let A be PSD. Let A be orthogonally diagonalized according to the above notation, so $Au_i = \lambda_i u_i$ for $i = 1, \ldots, n$. Then we fix i and want to show that $\lambda_i \geq 0$. By the definition of PSD, we know that $u_i^\top A u_i \geq 0$. But

$$u_i^\top A u_i = u_i^\top (\lambda_i u_i) = \lambda_i u_i^\top u_i = \lambda_i \|u_i\|^2 = \lambda_i.$$

Therefore $\lambda_i \geq 0$ as was to be shown.

For the reverse direction, let A be a symmetric matrix with nonnegative eigenvalues. We want to show that A is PSD, so letting $z \in \mathbb{R}^n$ it suffices to show that $z^\top A z \geq 0$. Let A be orthogonally diagonalized according to the above notation, and write

$$A = \sum_{i=1}^n \lambda_i u_i u_i^\top.$$

By assumption, the eigenvalues $\lambda_i \geq 0$ are nonnegative. Then

$$z^\top A z = \sum_{i=1}^n \lambda_i (z^\top u_i)(u_i^\top z) = \sum_{i=1}^n \lambda_i (z \cdot u_i)^2 \geq 0,$$

as was to be shown. □

Exercise 2.27. Prove the second statement of the lemma, regarding PD matrices.

To relate these definitions to convexity, we need to define one more thing:

Definition 2.28 (Hessian)**.** If f is twice differentiable at a point $x \in \mathbb{R}^n$, then define the Hessian matrix

$$\nabla^2 f(x) = \left(\frac{\partial^2 f}{\partial x_i \partial x_j}(x) \right)_{i,j=1}^n,$$

i.e., the matrix of all mixed second partial derivatives.

Remark 2.29 (Twice differentiability). There are some subtleties to point out about what we mean by differentiability here. When we say that f is twice differentiable at x_0, we always interpret this in the sense of Fréchet differentiability. What this means is that there exists a gradient vector $\nabla f(x_0)$ and a Hessian matrix $\nabla^2 f(x_0)$ such that the resulting second-order Taylor expansion $f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{1}{2}(x - x_0)^\top \nabla^2 f(x_0)\,(x - x_0)$ agrees with $f(x)$ near x_0 up to a term that goes to zero faster than $\|x - x_0\|^2$ as $x \to x_0$. We will state what we need more concretely in our proofs. Now when a function is twice differentiable in this sense, in fact the Hessian matrix is *automatically symmetric*.

The confusing point is that higher-order multivariate differentiability is sometimes defined in multivariate calculus courses in terms of the existence of partial derivatives. This actually yields a weaker notion, and one needs an additional assumption that the partial derivatives are continuous to ensure that partial derivatives commute—and that in turn the Hessian is symmetric. This is the content of the famous Clairaut-Schwarz theorem. However, we need not adopt this additional assumption if we define differentiability in the more natural way.

Finally, we remark that there is a difficult theorem (**Alexandrov's theorem**) saying that convex functions are twice differentiable almost everywhere! We will not prove this theorem, but it is reassuring to keep in mind.

Theorem 2.30 (Convexity in terms of the Hessian)**.** *A function $f : C \to \mathbb{R}$ which is twice differentiable on an open convex domain $C \subset \mathbb{R}^n$ is convex if and only if $\nabla^2 f(x)$ is PSD for all $x \in C$. In fact, it is μ-strongly convex if and only if $\nabla^2 f(x) - \mu I_n$ is PSD for all $x \in C$.*

Remark. In fact our arguments will show that if f is μ-strongly convex, then $\nabla^2 f(x) - \mu I_n$ is PSD for any point $x \in C$ where f is twice differentiable, even if f is not twice differentiable on all of C.

The fact that $\nabla^2 f(x) - \mu I_n$ is PSD is also equivalent to saying that $z^\top \nabla^2 f(x) z \geq \mu \|z\|^2$ for all $z \in \mathbb{R}^n$. (*Why?*) In light of the following exercise, it is also equivalent to saying that the eigenvalues of $\nabla^2 f(x)$ are all at least μ.

Exercise 2.31. Suppose that $A \in \mathbb{R}^{n \times n}$ is diagonalizable with eigenvalues $\lambda_1, \ldots, \lambda_n$. Show for any $\mu \in \mathbb{R}$, the matrix $A - \mu I_n$ is diagonalizable with eigenvalues $\lambda_1 - \mu, \ldots, \lambda_n - \mu$.

Proof of Theorem 2.30. We will prove the second statement for general $\mu \geq 0$, since it recovers the first statement in the case $\mu = 0$.

We start with the forward direction. Let $f : C \to \mathbb{R}$ be twice differentiable and μ-strongly convex. Let $x_0 \in C$. We want to show that $\nabla^2 f(x_0) - \mu I_n$ is PSD. The matrix is symmetric since the Hessian must be symmetric. Then fix $z \in \mathbb{R}^n$, so it suffices to show that $z^\top \left[\nabla^2 f(x_0) - \mu I_n\right] z \geq 0$.

The definition of differentiability means that for all x near x_0, we have

$$f(x) = f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{1}{2}(x - x_0)^\top \left[\nabla^2 f(x_0)\right](x - x_0) + \psi(x - x_0)\|x - x_0\|^2,$$

where $\lim_{z \to 0} \psi(z) = 0$. (We will unpack this a bit in lecture.)

Meanwhile, strong convexity means that

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{1}{2}(x - x_0)^\top [\mu I_n](x - x_0).$$

If we subtract inequalities and rearrange, we obtain

$$\frac{1}{2}(x-x_0)^\top \left[\nabla^2 f(x_0) - \mu I_n\right] (x-x_0) \geq \psi(x-x_0)\|x-x_0\|^2$$

for all x in a neighborhood of x_0.

Since this holds for any x in a neighborhood of x_0, we can choose $x = x_0 + tz$ for any $t \neq 0$ sufficiently small, yielding (after dividing both sides by t^2):

$$\frac{1}{2} z^\top \left[\nabla^2 f(x_0) - \mu I_n\right] z \geq \psi(tz)\|z\|^2.$$

for all $t \neq 0$ sufficiently small. Then taking the limit as $t \to 0$, we obtain the desired result.

Now we prove the reverse direction. Suppose that $f : C \to \mathbb{R}$ be twice differentiable and that $\nabla^2 f(x) - \mu I_n$ is PSD for all $x \in C$. Fix $x, x_0 \in C$. To show μ-strong convexity it suffices to show that

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{1}{2}(x-x_0)^\top [\mu I_n] (x-x_0).$$

Now define $x_t = (1-t)x_0 + tx \in C$ for $t \in [0,1]$, and define $h : [0,1] \to \mathbb{R}$ by

$$h(t) = f(x_t) = f(x_0 + t(x - x_0)).$$

By applying the fundamental theorem of calculus (*twice!*) we deduce:

$$\begin{aligned} h(t) &= h(0) + \int_0^t h'(s)\, ds \\ &= h(0) + \int_0^t \left[h'(0) + \int_0^s h''(u)\, du\right] ds \\ &= h(0) + th'(0) + \int_0^t \left[\int_0^s h''(u)\, du\right] ds. \end{aligned} \tag{2.10}$$

Now

$$h(0) = f(x_0), \quad h'(0) = \nabla f(x_0) \cdot (x - x_0), \quad h''(u) = (x-x_0)^\top \nabla^2 f(x_t)(x - x_0).$$

Since $\nabla^2 f(x) - \mu I_n$ is PSD, it follows that $h''(u) \geq \mu \|x - x_0\|^2$.

Plugging these identities into (2.10) and evaluating at $t = 1$, we conclude that

$$\begin{aligned} f(x) &= h(1) \\ &\geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \int_0^1 \left[\int_0^s \mu \|x - x_0\|^2\, du\right] ds \\ &= f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \mu \|x - x_0\|^2 \int_0^1 s\, ds \\ &= f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{\mu}{2}\|x - x_0\|^2, \end{aligned}$$

as was to be shown. □

2.2.6 Examples

Now we have enough theory to demonstrate a variety of examples of convex functions. By way of Exercises 2.11, 2.12, and 2.14, these building blocks can be used to construct most examples of convex functions appearing in the wild. In turn, our dictionary of convex functions allows us, by way of Exercise 2.15 ('sublevel sets of convex functions are convex'), to understand most types of convex constraint sets of practical interest.

Exercise 2.32 (Exponential is convex)**.** $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = e^x$ is convex.

Exercise 2.33 (Powers greater than 1 are convex)**.** $f : [0, \infty) \to \mathbb{R}$ defined by $f(x) = x^p$ is convex when $p > 1$.

Exercise 2.34 (Powers in $(0, 1)$ are concave)**.** $f : [0, \infty) \to \mathbb{R}$ defined by $f(x) = -x^p$ is convex when $p \in (0, 1)$.

Exercise 2.35 (Log is concave)**.** $f : (0, \infty) \to \mathbb{R}$ defined by $f(x) = -\log x$ is convex.

Exercise 2.36 (Negative entropy is convex)**.** $f : [0, \infty) \to \mathbb{R}$ defined by $f(x) = x \log x$ for $x > 0$ and $f(0) = 0$ is convex.

Exercise 2.37 (ReLU is convex)**.** $f : [0, \infty) \to \mathbb{R}$ defined by $f(x) = \max(0, x)$ is convex.

Exercise 2.38 (Absolute value is convex)**.** $f : [0, \infty) \to \mathbb{R}$ defined by $f(x) = |x|$ is convex. **Hint:** Write as a pointwise maximum of two convex functions.

Exercise 2.39 (PSD quadratics are convex)**.** Consider $f : \mathbb{R}^n \to \mathbb{R}$ defined by

$$f(x) = \frac{1}{2} x^\top A x + b^\top x + c$$

where $A \in \mathbb{R}^{n \times n}$, $b \in \mathbb{R}^n$, and $c \in \mathbb{R}$. In general, we can always assume that A is symmetric because if it's not, we can replace $A \leftarrow \frac{1}{2}(A + A^\top)$, which does not alter the function value. (*Show this.*) Also show that if A is PSD, then f is convex. **Hint:** To start, you should verify that $\nabla f(x) = Ax + b$ and then $\nabla^2 f(x) = A$ for all $x \in \mathbb{R}^n$.

Exercise 2.40 (Least squares objective is convex)**.** Consider $f : \mathbb{R}^n \to \mathbb{R}$ defined by $f(x) = \|Ax - b\|^2$ where $A \in \mathbb{R}^{m \times n}$ as an $m \times n$ (possibly rectangular) matrix and $b \in \mathbb{R}^m$. Show that f is convex. **Hint:** By using the identity $\|z\|^2 = z^\top z$ and matrix algebra, you can reduce to a quadratic function in the sense of the last exercise. The matrix will be $A^\top A$, so you can conclude by Exercise 2.24.

In fact there is another way to see that the least squares objective is convex, based on the fact that all norms are convex. First we review the definition of a norm.

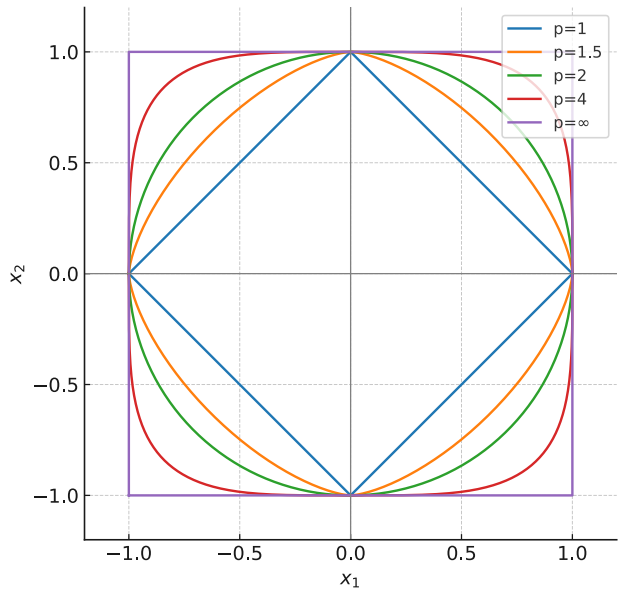

Figure 2.10: Unit p-norm ball boundaries $\{x \in \mathbb{R}^2 : \|x\|_p = 1\}$ in the plane for several values of p.

Definition 2.41 (Norm)**.** A norm is a function $\|\cdot\| : \mathbb{R}^n \to \mathbb{R}$ satisfying three properties

1. $\|x\| \geq 0$ for all $x \in \mathbb{R}^n$ with equality if and only if $x = 0$.
2. Homogeneity: $\|\lambda x\| = |\lambda|\|x\|$ for all $\lambda \in \mathbb{R}$ and $x \in \mathbb{R}^n$.
3. Triangle inequality: $\|x + y\| \leq \|x\| + \|y\|$ for all $x, y \in \mathbb{R}^n$.

Remark 2.42. In most situations, for simplicity we will use $\|\cdot\|$ to denote the 2-norm or Euclidean norm $\|x\|_2^2 = x \cdot x$. But in the current context, we use the notation to indicate a more general norm.

Exercise 2.43 (Norms are convex)**.** Any norm is a convex function.

The most important norms are the p-norms (cf. Figure 2.10), defined by

$$\|x\|_p := \left(\sum_{i=1}^{n} |x_i|^p \right)^{1/p}$$

for $p \in [1, \infty)$.[2] Note that the case $p = 2$ recovers the Euclidean norm, our typical default choice. In the limiting case $p = \infty$, we recover the ∞-norm

$$\|x\|_\infty = \max_{i=1,\ldots,n} |x_i|.$$

Note that we can also verify directly that the map $x \mapsto \|x\|_\infty$ is convex because it is a pointwise maximum of convex functions!

[2] When $p \in (0, 1)$, this formula does *not* define a norm. The triangle inquality does not hold.

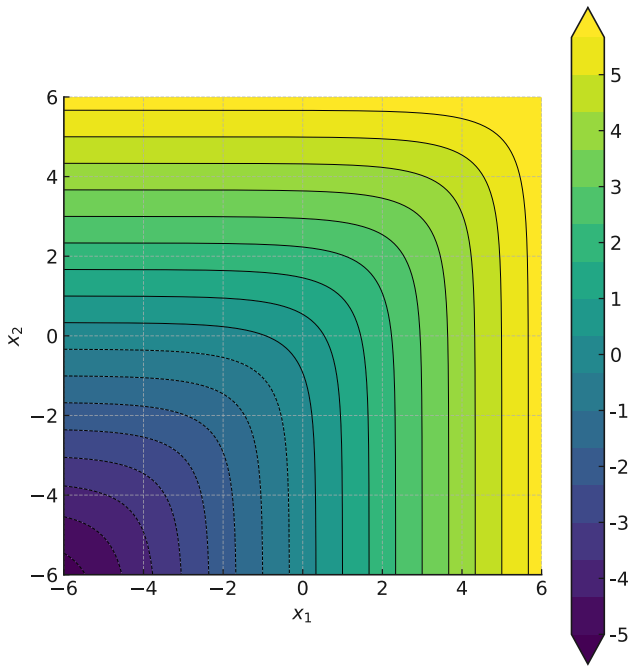

Figure 2.11: Contour plot of LogSumExp on $\mathbb{R}^2$, i.e., $f(x_1, x_2) = \log(e^{x_1} + e^{x_2})$.

Exercise 2.44. Verify that the p-norm is actually a norm in the cases $p = 1, 2, \infty$. **Hint:** In the case $p = 2$, you will have to use the Cauchy-Schwarz inequality, which says that $|x \cdot y| \leq \|x\|_2 \|y\|_2$ for all $x, y \in \mathbb{R}^n$.

Example 2.45. It follows then from Exercise 2.43, together with Exercise 2.12 (composition with affine-linear functions), that $f(x) := \|Ax - b\|_p$ defines a convex function for any p-norm.

Thus we have seen that both $x \mapsto \|Ax - b\|_2$ and $x \mapsto \|Ax - b\|_2^2$ are convex functions. In fact the first fact implies the second fact via the following result.

Exercise 2.46. Suppose $h : D \to \mathbb{R}$ is convex and non-decreasing on a convex domain $D \subset \mathbb{R}$ and $g : C \to D$ is convex on a convex domain $C \subset \mathbb{R}^n$. Then the composition $f : C \to \mathbb{R}$ defined by $f(x) = h(g(x))$ is convex. **Hint:** If you want, you can prove this using the chain rule, under the extra assumption that everything is as differentiable as you like. (Easiest to start in the simplest special case $C = \mathbb{R}$, and note that the fact that h is non-decreasing means that $h' \geq 0$.) But it is also good to try proving directly using the definition of convexity, without any differentiability assumptions.

We will introduce one more building block. It looks a bit funny, but it is motivated by our case study of softmax / logistic regression below. It also has an important role in statistical mechanics, which we won't really touch on.

Example 2.47 (LogSumExp is convex)**.** $f : \mathbb{R}^n \to \mathbb{R}$ defined by $f(x) = \log\left(\sum_{i=1}^n e^{x_i}\right)$ is convex. Sometimes this function is denoted $\mathrm{LSE}(x)$. We will verify convexity directly by computing the Hessian, though see the remark below for an alternative point of view.

First compute using the chain rule:

$$\frac{\partial f}{\partial x_i}(x) = \frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}}, \qquad \frac{\partial^2 f}{\partial x_i \partial x_j}(x) = \delta_{ij}\frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}} - \frac{e^{x_i}e^{x_j}}{\left(\sum_{k=1}^n e^{x_k}\right)^2},$$

where

$$\delta_{ij} = \begin{cases} 1, & i = j \\ 0, & i \neq j \end{cases}$$

is the Kronecker delta.

It is helpful to define a 'probability vector' $\pi(x)$ with entries

$$\pi_i(x) = \frac{e^{x_i}}{\sum_{k=1}^n e^{x_k}},$$

so that $\pi(x) \geq 0$ (entrywise) and $\sum_{i=1}^n \pi_i(x) = 1$. In terms of $\pi(x)$, we can rewrite the gradient and Hessian as

$$\frac{\partial f}{\partial x_i}(x) = \pi_i(x), \qquad \frac{\partial^2 f}{\partial x_i \partial x_j}(x) = \delta_{ij}\pi_i(x) - \pi_i(x)\,\pi_j(x),$$

or more compactly as

$$\nabla f(x) = \pi(x), \qquad \nabla^2 f(x) = \mathrm{diag}\left[\pi(x)\right] - \pi(x)\,\pi(x)^\top.$$

Then to guarantee that $\nabla^2 f(x)$ is PSD, let $z \in \mathbb{R}^n$. We want to show that $z^\top \nabla^2 f(x)\, z \geq 0$, so compute

$$\begin{aligned} z^\top \nabla^2 f(x)\, z &= \sum_{i,j=1}^n \frac{\partial^2 f}{\partial x_i \partial x_j}(x)\, z_i z_j \\ &= \sum_{i=1}^n z_i^2 \pi_i(x) - \left(\sum_{i=1}^n z_i \pi_i(x)\right)^2. \end{aligned}$$

Note that for any fixed x, we can define convex combination coefficients $\theta_i = \pi_i(x)$ and interpret

$$z^\top \nabla^2 f(x)\, z = \sum_{i=1}^n \theta_i\, g(z_i) - g\left(\sum_{i=1}^n \theta_i\, z_i\right),$$

where $g : \mathbb{R} \to \mathbb{R}$ is simply the univariate convex function defined by $g(z) := z^2$. Then by our equivalent characterization of convex functions in terms of convex combinations (Exercise 2.10), we conclude that $z^\top \nabla^2 f(x)\, z \geq 0$, as was to be shown. □

Remark 2.48. In fact, it is possible understand LogSumExp as the 'convex conjugate' or Legendre transform of the negative entropy $y \mapsto \sum_{i=1}^n y_i \log y_i$. This offers a different way of proving convexity, and it is connected to the Gibbs variational principle in statistical mechanics. We will talk about convex conjugates in our study of duality.

Remark 2.49. LogSumExp can be viewed as a smooth approximation of the maximum function in the sense that if any of the entries $x_1, \dots, x_n$ are large, then it returns an approximation of $\max(x_1, \dots, x_n)$. More formally, for any $x \in \mathbb{R}^n$, the limit $\beta^{-1} f(\beta x) \to \max(x_1, \dots, x_n)$ holds as $\beta \to \infty$.

2.2.7 Further reading

It may seem strange, but we have actually covered most of the major ingredients appearing in practical convex optimization problems. The references of Boyd and Vandenberghe highlighted in the syllabus (including their very helpful lecture slides) are more exhaustive. The main points we have left out are those motivated by semidefinite programming (SDP). Specifically, two important facts for SDP are (1) that the set of PSD matrices is convex and (2) that the function $\log\det X$ is concave on the set of PD matrices. In this course, we will not get into SDP.

2.3 Regression and classification

We digress here to describe some important applications of convex optimization: regression and classification. Specifically we will discuss linear and softmax / logistic regression, which are widely used frameworks defining convex optimization problems. In fact, modern deep learning approaches to regression and classification can be viewed as extending these basic building blocks, although they define nonconvex optimization problems.

2.3.1 Unconstrained quadratic optimization and least squares

We are headed toward a discussion of linear regression, but first we pause to make some mathematical observations about some of the few nontrivial multivariate optimization problems that can be solved in *closed form*.

First consider the unconstrained optimization problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}}\ f(x), \quad \text{where} \quad f(x) := \frac{1}{2} x^\top A x - b \cdot x, \quad A \in \mathbb{R}^{n \times n} \text{ is positive definite.}$$

The objective is strongly convex, so (as we will prove later in Section 2.4), we are guaranteed that there exists a unique optimizer, which is the unique critical point. We can solve for the critical point as

$$0 = \nabla f(x) = Ax - b,$$

meaning that the unique solution is given by $x = A^{-1} b$, or alternatively, we can think of the solution as being given by solving the linear system

$$Ax = b.$$

Depending on your point of view, you can think of the linear system as being easy or hard. If $O(n^3)$ cost is tractable, then we can obtain a solution via Gaussian elimination. However, many

methods for solving such PD linear systems are based on the optimization perspective indicated above, most notably the *conjugate gradient* (CG) method. This is covered in Math 128B, so we will not talk about it for now. Depending on the *condition number* of the matrix A (most simply, the ratio of largest to smallest eigenvalue), as well its sparsity, the system can potentially be solved with $O(n)$ cost. The obstacle of conditioning is related to difficulties we will encounter in general convex optimization, as we will highlight later.

Meanwhile, consider also the least squares problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}}\ f(x), \quad \text{where} \ \ f(x) := \|Ax-b\|_2^2, \quad A\in\mathbb{R}^{m\times n},\ b\in\mathbb{R}^m.$$

We can rewrite the objective (cf. the algebraic manipulations required by Exercise 2.40) as

$$f(x) = (Ax-b)^\top(Ax-b) = x^\top A^\top Ax - 2(A^\top b)\cdot x + b^\top b, \tag{2.11}$$

which, up to an irrelevant constant term, can be viewed as a quadratic optimization problem by putting $A^\top A$ in the place of A and $A^\top b$ in the place of b. This means that the unique solution should be given by

$$x = (A^\top A)^{-1}A^\top b,$$

or alternatively as the solution of the linear system of equations

$$(A^\top A)x = A^\top b.$$

The equations defining the latter system are called the **normal equations**.

There is a catch! What if $A^\top A$ fails to be invertible? Indeed, if $m < n$, so the linear system $Ax = b$ is *underdetermined*, then since the rank of $A^\top A$ is at most m, it follows that $A^\top A$ cannot be invertible. This corresponds to the possibility the least squares solution is not unique. We know that $A^\top A$ is invertible when $m \geq n$ and A has full rank (i.e., rank n, meaning that the columns of A are linearly independent). In this case, there exists a unique solution to the least squares problem, even though the system of equations "$Ax = b$" may be *overdetermined*.

In order to resolve the possibility that the system is underdetermined, or more generally to 'regularize' the solution of a least squares problem, it is typical to include a regularization term in the problem, defining a modified objective

$$f_\lambda(x) := \|Ax-b\|_2^2 + \lambda\|x\|_2^2, \tag{2.12}$$

where $\lambda > 0$ is a 'regularization' parameter. We will discuss the purpose of regularization below in Section 2.3.2, but for now we note that such a modified objective still admits a closed-form optimizer, because it is still quadratic.

By (2.11) and the fact that $\|x\|_2^2 = x^\top I_n x$, we see that

$$f_\lambda(x) = x^\top(A^\top A + \lambda I_n)x - 2(A^\top b)\cdot x + b^\top b.$$

Since $\nabla^2 f_\lambda(x) = A^\top A + \lambda I_n$, it follows that f_λ is λ-strongly convex and therefore admits a unique optimizer whenever $\lambda > 0$, as we will prove later in Section 2.4. The modified normal equations are

$$(A^\top A + \lambda I_n)x = A^\top b.$$

It is worth pointing out now that the related objective

$$\|Ax-b\|_2^2 + \lambda\|x\|_1, \tag{2.13}$$

which regularizes the least squares problem in a different way, *does not* admit a closed-form optimizer and requires more general convex optimization tools to minimize! Moreover, the objective is not everywhere differentiable, and the optimizer is typically obtained at a point where the gradient does not exist (i.e., where $x_i = 0$ for some i). In fact, the purpose of this regularization term is to force this to happen, as we shall discuss below.

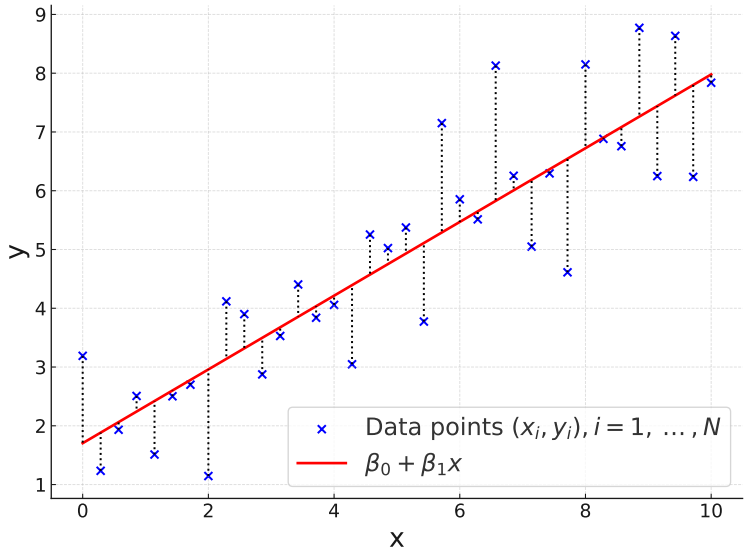

Figure 2.12: Graph of affine-linear least squares fit for a 1-dimensional regression problem. (No regularization.)

2.3.2 Linear regression

To talk about regression, we have to shift our notation a bit in order to match the literature. Somewhat annoyingly, the letters x and y will be used here to denote fixed quantities ('data'), *not* optimization variables.

Now suppose we are given $x_i \in \mathbb{R}^p$, $y_i \in \mathbb{R}$, for $i = 1, \ldots, N$. The goal of regression, at a very high level, is to determine a function $F : \mathbb{R}^p \to \mathbb{R}$ that fits the data well, i.e., achieves $F(x_i) \approx y_i$ for $i = 1, \ldots, N$, and morever can be used to generate a prediction $F(x)$ for the value of y that should be associated to a new arbitrary input x.

Specifically, we will look for an affine-linear transformation of the form $F(x) = \beta_0 + \beta \cdot x$ where $\beta = (\beta_1, \ldots, \beta_p)^\top \in \mathbb{R}^p$ ('slopes') and $\beta_0 \in \mathbb{R}$ ('intercept') are unknown parameters for an arbitrary affine-linear map. The goal is to find parameters that fit the data in the sense that

$$\beta_0 + \beta \cdot x_i \approx y_i, \quad i = 1, \ldots, N.$$

We penalize the approximation error in the least squares sense, yielding the optimization problem

$$\underset{\beta \in \mathbb{R}^p,\, \beta_0 \in \mathbb{R}}{\text{minimize}} \quad \frac{1}{N} \sum_{i=1}^{N} [\beta_0 + \beta \cdot x_i - y_i]^2 + \mathcal{R}(\beta), \tag{2.14}$$

where $\mathcal{R} : \mathbb{R}^p \to \mathbb{R}$ is a **regularization** term (typically convex) that is designed to prevent overfitting. We will discuss several choices below, but first we note that regardless of the choice, it is possible to eliminate the intercept variable β_0 in a way that is equivalent to preprocessing the dataset.

Indeed, suppose that we freeze the slope parameters β and (given this choice) simply optimize the intercept parameter β_0 in the above optimization problem. This amounts to a univariate, smooth convex optimization problem which can be solved exactly by taking the partial derivative of the least squares objective with respect to β_0 and setting the resulting expression equal to zero. (Note that the regularization term is constant with respect to β_0, hence can be ignored.)

Exercise 2.50. Show that for fixed β in the optimization problem (2.14), the optimal β_0 is given by $\beta_0 = \overline{y} - \beta \cdot \overline{x}$, where

$$\overline{x} := \frac{1}{N}\sum_{i=1}^{N} x_i, \qquad \overline{y} := \frac{1}{N}\sum_{i=1}^{N} y_i$$

are the empirical means of the x and y datasets. **Hint:** Refer to the strategy outlined in the preceding discussion.

Based on the result of this exercise, we conclude that we can eliminate the variable β_0 from (2.14) by plugging $\beta_0 = \overline{y} - \beta \cdot \overline{x}$ into the objective function, yielding the optimization problem

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{N}\sum_{i=1}^{N} [\beta \cdot (x_i - \overline{x}) - (y_i - \overline{y})]^2 + \mathcal{R}(\beta).$$

Note that this is equivalent to a *linear* regression problem on a modified dataset $\tilde{x}_i := x_i - \overline{x}$, $\tilde{y}_i := y_i - \overline{y}$, $i = 1, \ldots, N$. We say that such a modified dataset has been *de-meaned* because the modified means $\frac{1}{N}\sum_{i=1}^{N} \tilde{x}_i = 0$, $\quad \frac{1}{N}\sum_{i=1}^{N} \tilde{y}_i = 0$ are both zero by construction. (*Check this!*)

Thus we typically always assume before proceeding further that our dataset (x_i, y_i), $i = 1, \ldots, N$ is already de-meaned in the sense that $\frac{1}{N}\sum_{i=1}^{N} x_i = 0$ and $\frac{1}{N}\sum_{i=1}^{N} y_i = 0$, and without loss of generality we can then consider linear regression in which $\beta_0 = 0$:

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{N}\sum_{i=1}^{N} [\beta \cdot x_i - y_i]^2 + \mathcal{R}(\beta), \tag{2.15}$$

It is useful to collect the dataset into the following objects:

$$X = \begin{pmatrix} x_1^\top \\ \vdots \\ x_N^\top \end{pmatrix} \in \mathbb{R}^{N\times p}, \quad y = \begin{pmatrix} y_1 \\ \vdots \\ y_N \end{pmatrix} \in \mathbb{R}^N,$$

so that the i-th row of X is the row vector $x_i^\top$ for each $i = 1, \ldots, N$. Then the objective of (2.15) can be written more compactly to yield

$$\underset{\beta\in\mathbb{R}^p}{\text{minimize}} \quad \frac{1}{N}\|X\beta - y\|_2^2 + \mathcal{R}(\beta).$$

Now we discuss the choice of the regularizer. Recall that regularization is necessary when $N \leq p$, so that the system $X\beta = y$ is underdetermined, but it may be important in any case to improve the generalization error of the regressor, i.e., to avoid overfitting.

When simply $\mathcal{R}(\beta) = 0$, the recipe for producing β is simply called **ordinary least squares (OLS)**. More generally, linear regression with an "L_2" or "Tikhonov" regularizer

$$\mathcal{R}(\beta) := \lambda\|\beta\|_2^2, \quad \lambda > 0$$

is called **ridge regression** and yields a problem that is mathematically equivalent to a problem of the form (2.12).

Meanwhile, linear regression with an "L_1" regularizer

$$\mathcal{R}(\beta) := \lambda\|\beta\|_1, \quad \lambda > 0$$

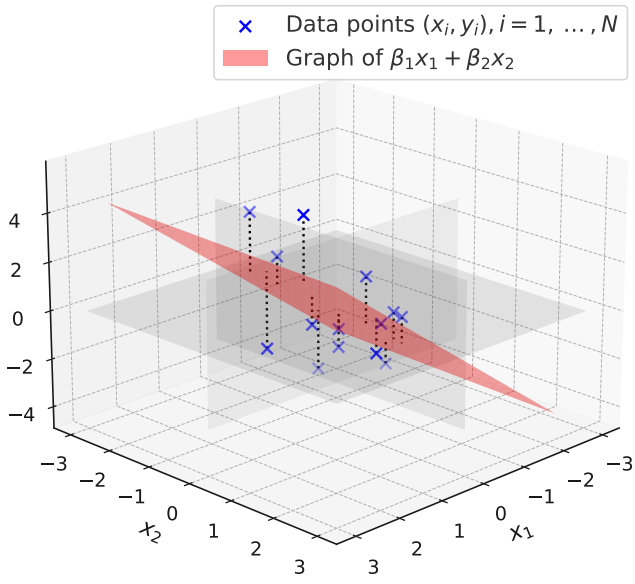

Figure 2.13: Graph of linear least squares fit for a 2-dimensional regression problem. The data has been de-meaned, and there is no intercept parameter, so the graph passes through the origin. (No regularization.).

is called **LASSO** ('least absolute shrinkage and selection operator') and is mathematically equivalent to optimizing an objective of the form (2.13), which cannot be achieved with a closed-form solution.

Generally the purpose of L_1 regularization is to promote a *sparse* solution vector β in which some or many components are zero. Thus LASSO can be viewed as a tool for *feature selection*, in which all but the most informative columns of the dataset X are discarded. It is common to perform a second OLS fit using only the features that were selected.

Do we really need to worry about overfitting even for linear regression? Yes! Linear regression underlies many studies in the social and biomedical sciences, where broad failures to generalize have been observed recently in what is known as the *replication crisis*.

Think of each data point $x_i \in \mathbb{R}^p$ as a potentially high-dimensional feature vector corresponding to a person, e.g., in a clinical trial. Indeed, the number of features p can be made arbitrarily high by passing an initial dataset through an arbitrary feature map $\phi : \mathbb{R}^p \to \mathbb{R}^{p'}$ and then performing linear regression using the transformed data points $x_i' = \phi(x_i) \in \mathbb{R}^{p'}$, $i = 1, \ldots, N$, in place of the original data points. Regularization may be necessary to prevent overfitting with these extra dimensions and/or to extract an interpretable and limited subset of informative features.

In modern deep learning, a layer of linear regression often appears as the final layer. Regression by deep learning can be interpreted as finding a feature map $\phi : \mathbb{R}^p \to \mathbb{R}^{p'}$ and then performing linear regression using the transformed data $x_i' \in \mathbb{R}^{p'}$. Typically, regularization does not appear directly in modern neural network training, as it is believed that the dynamics of training itself using stochastic gradients acts as an implicit regularizer. Our perspectives on this point continue to evolve. It is worth pointing out that the datasets used in the most spectacular examples of modern deep learning are extremely large relative to those available in the social and biomedical sciences.

2.3.3 Softmax and logistic regression

As we have seen in the last section, regression is about finding a function $F : \mathbb{R}^p \to \mathbb{R}$ such that $F(x_i) \approx y_i$ on a given dataset (x_i, y_i), $i = 1, \ldots, N$. Then when we see a new input x, we can predict an outcome $F(x)$.

But what if we have a dataset in which the y values are not real numbers, but rather *labels* from a discrete set. For example, each point x_i might be a vector encoding of an image of an animal, and $y_i \in \{1, 2, 3\}$ may be a discrete label: 1 if cat, 2 if dog, 3 if other. More generally we will allow $y_i \in \{1, \ldots, k\}$ for arbitrary positive integer k. How can we construct a classifier that assigns a label to a new image x? Again we think of the training data (x_i, y_i), $i = 1, \ldots, N$ as fixed, not as optimization variables.

Softmax regression [3] is perhaps the most basic framework for classification, but it is of fundamental importance in machine learning because the last layer of essentially all deep classifiers is based on softmax regression. Classification by deep learning can typically be viewed as finding a feature map that transforms the input data, then performing softmax regression on the transformed data. Note the analogy to our comments above about regression with deep neural networks.

Softmax regression consists of a composition of two layers: an affine-linear layer and a softmax layer. The affine-linear layer will map $\mathbb{R}^p \to \mathbb{R}^k$ (recall: k is the number of label classes) according to

$$x \mapsto Wx + b,$$

where now we adopt the machine learning conventions: $W \in \mathbb{R}^{k \times p}$ for 'weights' and $b \in \mathbb{R}^k$ for 'biases.' These are the model parameters that we need to 'learn' or optimize.

It is helpful to remove the biases from the notation via the following observation:

$$Wx + b = \tilde{W}\tilde{x}, \quad \text{where } \tilde{W} := (W, b) \in \mathbb{R}^{k \times (p+1)}, \ \tilde{x} := \begin{pmatrix} x \\ 1 \end{pmatrix} \in \mathbb{R}^{p+1}.$$

Thus by appending an extra entry of 1 to each our input data points x_i, $i = 1, \ldots, N$, we can simply consider *linear* maps instead of affine-linear maps. Without loss of generality, we assume that this preprocessing step ('homogenizing the coordinates') has already been performed to furnish our given dataset. Then we seek simply a vector of weights, which will transform our input data as

$$x \mapsto Wx.$$

Next we define the softmax function, which generally appears as the final layer in most machine learning classifiers:

Definition 2.51 (Softmax)**.** The softmax function $\mathbb{R}^k \to \mathbb{R}^k$ is defined coordinate-wise as

$$[\text{softmax}(z)]_i = \frac{\exp(z_i)}{\sum_{j=1}^k \exp(z_j)}.$$

Note that for any $z \in \mathbb{R}^k$, softmax(z) is a probability vector in the sense that is has nonnegative entries which sum to 1.

[3] Although 'regression' appears in the name, usually we distinguish between the overall tasks of regression and classification. Softmax / logistic regression are ultimately doing classification.

Remark 2.52. The name 'softmax' derives from the fact that the function can be viewed as a smooth approximation to the 'argmax' function.[a] Indeed, if for some entry i, we have that $z_i \gg z_j$ for all $j \neq i$, then $\text{softmax}(z) \approx e_i$, the suitable standard basis vector, and the softmax function puts nearly all its weight on the maximizing index i.

[a]The nomenclature is somewhat confusing because it might be argued that LogSumExp deserves the name 'softmax' as a smooth approximation of the 'max' function. But that's just how it is.

Exercise 2.53. Show that the softmax function is the gradient of LogSumExp, i.e., $\text{softmax}(z) = \nabla \text{LSE}(z)$ for all $z \in \mathbb{R}^k$.

Then softmax regression attempts to model the conditional distribution of $y \in \{1, \dots, k\}$ given $x \in \mathbb{R}^p$ in terms of model parameters W. Since y is a discrete label in $\{1, \dots, k\}$, this conditional distribution can be viewed as a probability vector which we define in terms of W as

$$\pi(x; W) := \text{softmax}(Wx) \in \mathbb{R}^k. \tag{2.16}$$

The entries $\pi_j(x; W)$ are meant to achieve the following approximation of conditional probabilities:

$$\pi_j(x; W) \approx \mathbb{P}(y = j \,|\, x), \quad j = 1, \dots, k.$$

The key idea is that as a stepping stone toward classifying a new input x (i.e., assigning it a label $y \in \{1, \dots, k\}$), we assign it a probability vector $\pi(x; W)$ reflecting our guess about the probabilities of each of the labels that could be assigned.

Our model for this map of any input to a vector of probabilities is parametrized by W, which must somehow be determined by fitting it to our given data. Specifically, we will find parameters W that maximize the **likelihood**[4] of our observations (x_i, y_i), $i = 1, \dots, N$:

$$\prod_{i=1}^{N} \pi_{y_i}(x_i; W).$$

It is equivalent to minimize the **negative log-likelihood**, which we can view as defining an objective function over our parameters w:

$$\mathcal{L}(W) := -\sum_{i=1}^{N} \log \pi_{y_i}(x_i; W),$$

which is sometimes called the **cross-entropy loss**.

Unpacking the definition of π (2.16) and in turn the softmax function (Definition 2.51), we can expand the loss function in terms of the vectors $z_i = Wx_i \in \mathbb{R}^k$, $i = 1, \dots, N$ (each with entries $z_{i,j}$, $j = 1, \dots, k$):

$$\mathcal{L}(W) = -\sum_{i=1}^{N} \log \left(\frac{\exp(z_{i,y_i})}{\sum_{j=1}^{k} \exp(z_{i,j})} \right) = \sum_{i=1}^{N} \left[\text{LSE}(z_i) - z_{i,y_i} \right],$$

where we recall that LSE (LogSumExp) is defined in Example 2.47. Now we can write $z_{i,y_i} = e_{y_i} \cdot z_i$ and substitute $z_i = Wx_i$ to further calculate:

$$\mathcal{L}(W) = \sum_{i=1}^{N} \left[\text{LSE}(Wx_i) - e_{y_i}^\top Wx_i \right]. \tag{2.17}$$

[4]Technically, this principle is based on an assumption that the data are sampled independently from some joint distribution over (x, y).

Here we have used $e_y \in \mathbb{R}^k$ to denote the standard basis vector in $\mathbb{R}^k$ corresponding to any label $y \in \{1, \dots, k\}$.

Notice that $\mathcal{L} : \mathbb{R}^{k \times p} \to \mathbb{R}$ is a function with a matrix-valued input. However, we can discuss its convexity by identifying $\mathbb{R}^{k \times p} \simeq \mathbb{R}^{kp}$.

Theorem 2.54. *The cross-entropy loss is convex.*

Proof. Note that for each $i = 1, \dots, N$, the map $W \mapsto \mathrm{LSE}(Wx_i)$ is a composition of a convex function (LSE, see Example 2.47) with a linear transformation (hence convex), and the map $W \mapsto -e_{y_i}^\top W x_i$ is linear (hence convex). Therefore the loss function $\mathcal{L}$ is a sum of convex functions (hence convex). □

It is worth injecting a word of caution about evaluating the LogSumExp and softmax maps. Note that direct evaluation of $\mathrm{LSE}(z)$ by the formula of Example 2.47 may result in numerical overflow or underflow if $\max(z_1, \dots, z_n) \gg 0$ or $\max(z_1, \dots, z_n) \ll 0$, respectively, due to the intermediate exponential depenencies on the entries of z. But we know that the final values satisfy reasonable bounds:

Exercise 2.55. For any $z = (z_1, \dots, z_n)^\top \in \mathbb{R}^n$, we have the bounds

$$\max(z_1, \dots, z_n) \le \mathrm{LSE}(z) \le \max(z_1, \dots, z_n) + \log n,$$

as well as

$$0 \le \mathrm{softmax}(z) \le 1,$$

interpreted entrywise.

A more stable evaluation strategy is based on the following insight.

Exercise 2.56. Let $\mathbf{1}_n \in \mathbb{R}^n$ denote the vector of all ones. Then for any $z \in \mathbb{R}^n$ and $\alpha \in \mathbb{R}$,

$$\mathrm{LSE}(z) = \alpha + \mathrm{LSE}(z - \alpha \mathbf{1}_n).$$

In other words, shifting all the entries of the input by α results in shifting the output by α. Meanwhile,

$$\mathrm{softmax}(z) = \mathrm{softmax}(z - \alpha \mathbf{1}_n).$$

Remark 2.57. Based on this result, we can choose $\alpha = \max(z_1, \dots, z_n)$ for a safe evaluation of either function.

Alternative presentations. We conclude with some remarks connecting our presentation with some more standard presentations.

First, there is a slight redundancy in our parametrization due to the fact that the softmax function automatically normalizes the entrywise exponential. Therefore, we can simply fix the first entry that we feed into the softmax layer to be 0, and parametrize the *remaining* entries $2, \dots, k$ linearly in terms of x, i.e., define

$$\pi(x; W) = \mathrm{softmax} \begin{pmatrix} 0 \\ Wx \end{pmatrix},$$

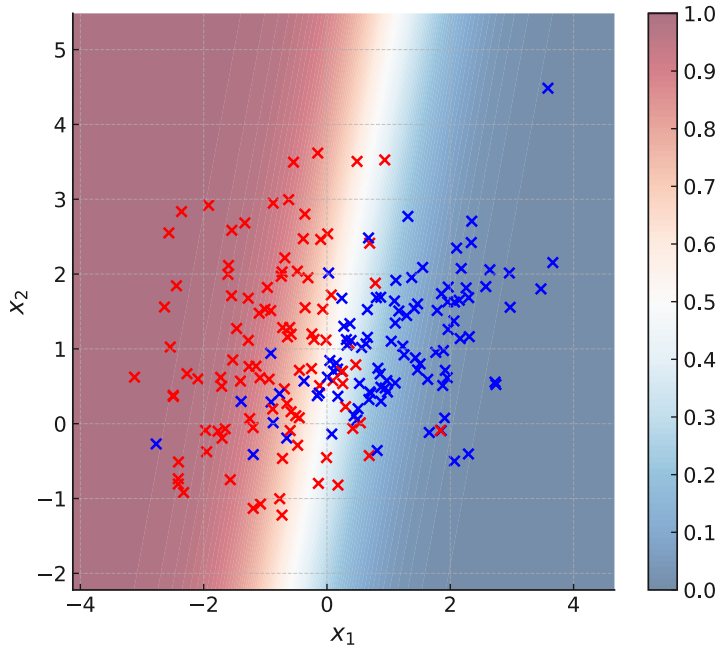

Figure 2.14: Depiction of logistic regression. The given data are $x_i \in \mathbb{R}^2$ and $y_i \in \{0, 1\}$, $i = 1, \ldots, N$, where the labels 0 and 1 are indicated with blue and red crosses, respectively. The background color plots the function $x \mapsto \sigma(w \cdot x + b)$ using the trained parameters w, b. (Here we maintain a bias parameter for illustrative purposes.) When we are given a new input point $x \in \mathbb{R}^2$ with unknown label y, the value $\sigma(w \cdot x + b)$ is our estimate of the probability that the unknown label of x is 1.

where now $W \in \mathbb{R}^{(k-1)\times p}$ (i.e., we explicitly zeroed out the first row of our original specification of W).

Second, softmax regression is often first presented in the special case of *binary classification* where the set of labels is $\{0, 1\}$, rather than $\{1, \ldots, k\}$. In this case, it is called **logistic regression**. Eliminating the redundancy as in the last paragraph, $W = w^\top \in \mathbb{R}^{1\times p}$ can be viewed as a single row vector, and we define

$$\pi(x; w) = \text{softmax}\begin{pmatrix} 0 \\ w \cdot x \end{pmatrix} = \begin{pmatrix} \dfrac{1}{1 + e^{w\cdot x_i}} \\ \dfrac{e^{w\cdot x_i}}{1 + e^{w\cdot x_i}} \end{pmatrix} = \begin{pmatrix} 1 - \sigma(w \cdot x_i) \\ \sigma(w \cdot x_i) \end{pmatrix},$$

where we have defined the **logistic function**

$$\sigma(t) := \frac{e^t}{1 + e^t} = \frac{1}{1 + e^{-t}}$$

and used the fact that it satisfies

$$1 - \sigma(t) = \sigma(-t).$$

Then in turn the **logistic loss** (which is the negative log-likelihood in this special case) reads concretely as

$$\mathcal{L}(w) = -\sum_{i=1}^{N} \{y_i \log [\sigma(w \cdot x_i)] + (1 - y_i) \log [1 - \sigma(w \cdot x_i)]\},$$

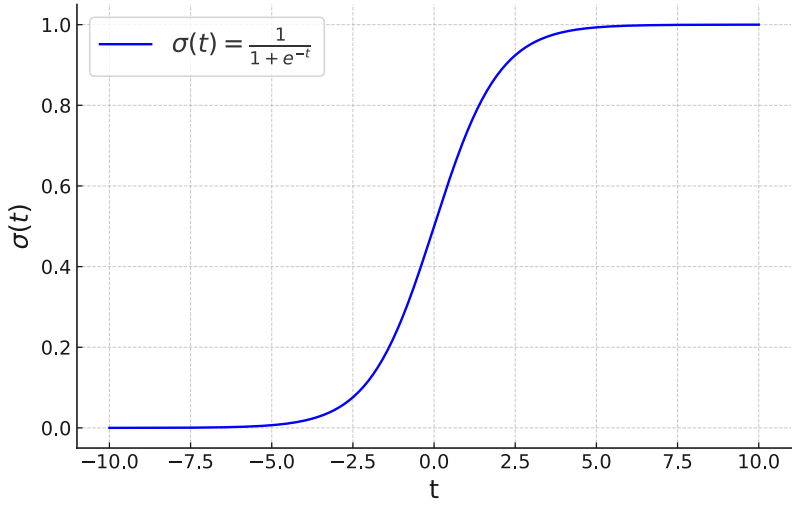

Figure 2.15: Plot of the logistic function.

where we have used the fact that $y_i \in \{0, 1\}$. This formula might be found more commonly in an introductory presentation. The name 'logistic regression' is based on the presence of the logistic function in this expression for the loss. The more general case with an arbitrary number of labels is sometimes called **multinomial logistic regression**.

Third, recall that if we have *not* preprocessed the input data x_i, $i = 1, \ldots, N$ by homogenizing the coordinates, we need to retain a vector of bias parameters b with the same number of rows as W. The logistic loss can then be viewed as a function $\mathcal{L}(W, b)$, in which all appearances of Wx_i have been replaced with $Wx_i + b$. (In our discussion of binary classification, b is a scalar and $w \cdot x_i$ should be replaced with $w \cdot x_i + b$.)

Practical discusssion. Why then have we chosen to present things that way we have?

In my opinion, removing the redundant degrees of freedom actually complicates the presentation by breaking the symmetry among the dimensions of the softmax layer. Relatedly, starting with binary classification also obscures some of the general points, especially via *ad hoc* expressions involving y_i and $(1 - y_i)$.

The presentation we have chosen sticks much more closely to how you will think of things when you train a deep classifier at some point in your life. Even more to the point, you will probably build the loss as something like

$$\mathcal{L}(W, b) = -\sum_{i=1}^{N} e_{y_i} \cdot \log \operatorname{softmax}(Wx_i + b), \tag{2.18}$$

where the log is interpreted entrywise. In particular, you will probably use the built-in implementation of the 'log softmax' function available in a machine learning library such as JAX or PyTorch, without expanding in terms of LogSumExp. Meanwhile, the vector $e_{y_i} \in \mathbb{R}^k$ is called a **one-hot encoding** of the label $y_i \in \{1, \ldots, k\}$ and also typically appears in implementation. Note that in current practice, it is actually *not typical* to remove the redundant degrees of freedom in the weights and biases W and b.

Finally, after training (W, b) according to the loss (2.18), how do we infer the label of a new input $x \in \mathbb{R}^p$? Recall that the vector $\pi(x; W, b) = \operatorname{softmax}(Wx + b) \in \mathbb{R}^k$ encodes our estimate

of the probabilities of each of labels $1, \ldots, k$. If we want to make a deterministic prediction, we simply choose the label to be the maximizing index of this vector of probabilities. However, in some generative modeling tasks, for example, we might use the entire vector of probabilities to sample a label at random with the appropriate probability.

2.4 Convex optimization problems

Our next task is to define more carefully what we mean by a convex optimization problem. We will then discuss several key properties that follow from the definition. Finally we will go over some important examples to supplement the regression and classification problems outlined above.

2.4.1 Definition and standard form

Definition 2.58 (Convex optimization problems)**.** A convex optimization problem is an optimization problem of the form

$$\begin{aligned} \text{minimize} \quad & f(x) \\ \text{subject to} \quad & x \in C, \end{aligned}$$

where $C \subset \mathbb{R}^n$ is a convex set $f : C \to \mathbb{R}$ is a convex function. We call f the *objective* and C the *feasible set*. We may say that the problem is strictly (resp., strongly or μ-strongly) convex if the objective is strictly (resp., strongly or μ-strongly) convex. If we just say strongly convex, this means μ-strongly convex for some $\mu > 0$. If C is empty, we say that the problem is *infeasible*.

Remark 2.59. More technically, we assume that convex optimization problems are defined by lower semi-continuous (l.s.c.) objectives $f : \mathbb{R}^n \to \mathbb{R}$, cf. Remark 2.19. The feasible set can always be encoded into the objective by setting $f(x) = +\infty$ for all $x \notin C$. An equivalent definition of lower semi-continuity is that for any x and any sequence $\{x_k\}_{k=0}^{\infty}$ converging to x, we have that $f(x) \leq \liminf_{k\to\infty}(x_k)$.

The assumption that f so defined is l.s.c. rules out the possibility of pathologies at the boundary of the domain C, cf. Figure 2.16. Note that the assumption is always satisfied as long as C is closed f is continuous on all of C. However, it also allows for 'barrier functions' that tend to $+\infty$ as we approach the boundary of C, cf. Figure 2.17.

Typically, the convex set C is specified as an intersection $C = \bigcap_{i=1}^m C_i$ of sets cut out by convex inequality constraints:

$$C_i := \{x \in \mathbb{R}^n \,:\, f_i(x) \leq 0\}, \quad i = 1, \ldots, m,$$

where each $f_i : \mathbb{R}^n \to \mathbb{R}$ is a convex function. Since intersections of convex sets are convex (Exercise 2.2) and sublevel sets of convex functions are convex (Exercise 2.15), it follows that any such C is a convex set. You may wonder why we consider only 0-sublevel sets, but note that by adding a constant to a function (which does not alter its convexity), we can view any sublevel set as a 0-sublevel set.

What about equality constraints? Later on we will consider general equality constraints of the form $h(x) = 0$ where $h : \mathbb{R}^n \to \mathbb{R}$ is fairly arbitrary. But we can only view such constraints as *convex* constraints when h is affine-linear. Indeed, think of an equality constraint as an intersection of two inequality constraints $h(x) \leq 0$ and $-h(x) \leq 0$. These are both convex constraints if and only if both h and $-h$ are convex functions. The following exercise establishes that this is only the case for affine-linear functions.

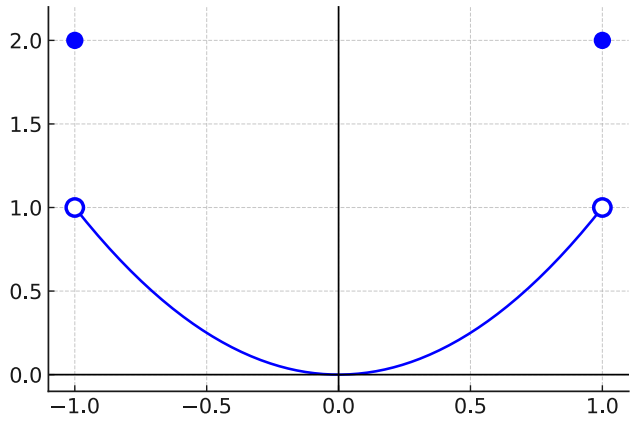

Figure 2.16: A convex function on the domain $[-1, 1]$ which is not l.s.c. We typically rule out such pathological behavior.

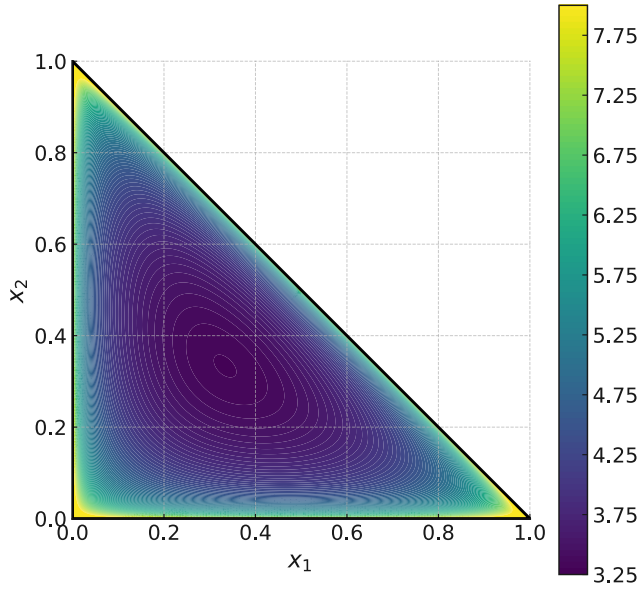

Figure 2.17: Illustration of the convex barrier function $f(x_1, x_2) = -\log x_1 - \log x_2 - \log(1 - x_1 - x_2)$ for the convex domain defined by the constraints $x_1 \geq 0$, $x_2 \geq 0$, and $x_1 + x_2 \leq 1$. Note that although the barrier function blows up to $+\infty$ near the boundary, it is still l.s.c. in the sense of Figure 2.59.

Exercise 2.60. Consider $f : \mathbb{R}^n \to \mathbb{R}$. Show that if both f and $-f$ are convex, then f is an affine-linear function, i.e., there exist $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$ such that $f(x) = a \cdot x - b$. **Hint:** If you want, you can just assume that there exists some point in C where f is differentiable, then apply the remark after Theorem 2.22 to both f and $-f$. For a challenge, try using Exercise 2.10 to construct an open region on which f is affine-linear, hence in particular differentiable. Then you can conclude using the first part of the hint.

Remark 2.61 (Standard form). In light of the preceding observations, we often present a general convex optimization problem in the **standard form**:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & f_i(x) \le 0, \quad i = 1, \dots, m, \\ & a_i \cdot x = b_i, \quad i = 1, \dots, p. \end{aligned}$$

where $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \dots, m$ are convex functions defining the inequality constraints and $a_i \in \mathbb{R}^n$, $b_i \in \mathbb{R}$, $i = 1, \dots, p$ define the equality constraints. The equality constraints can be written more compactly as

$$Ax = b,$$

where

$$A = \begin{pmatrix} a_1^\top \\ \vdots \\ a_p^\top \end{pmatrix} \in \mathbb{R}^{p\times n}, \quad b = \begin{pmatrix} b_1 \\ \vdots \\ b_p \end{pmatrix}.$$

Technically, the **optimal value** of an optimization problem, often denoted[5] $p^\star$, is defined as the *infimum*:

$$p^\star := \inf_{x\in C} f(x).$$

Ideally, there exists some $x^\star$ such that $p^\star = f(x^\star)$. In this case, the infimum is a bona fide minimum, and we say that optimal value or minimum is **attained**, and $x^\star$ is an **optimal solution** or **minimizer** / **optimizer**. However, as we shall explain below, the optimal value may not always be attained. In fact, the objective f may not even be bounded below on C, in which case we allow formally for the infimum to attain the value $-\infty$. Note moreover, that even if the optimal value is attained, there may not be a unique optimal solution.

2.4.2 Uniqueness and existence of minimizers

First we observe that strict convexity guarantees uniqueness (not existence!) for the optimal solution.

Lemma 2.62 (Minimizers are unique for strictly convex problems)**.** *Consider a convex optimization problem with a strictly convex objective. Any minimizer must be unique.*

Proof. Let f be the objective and C the feasible set. Suppose $x, y \in C$ are both minimizers, so $f(x) = f(y) = p^\star$. It suffices to show that $x = y$.

[5] P is for 'primal,' by contrast with the 'dual' that we will discuss later.

Suppose for contradiction that $x \neq y$. Then by strict convexity,

$$f\left(\frac{1}{2}x + \frac{1}{2}y\right) < \frac{1}{2}f(x) + \frac{1}{2}f(y) = p^\star.$$

But this contradicts the optimality of $p^\star$. □

More generally, the set of optimal solutions is convex (even if it is not a singleton).

Lemma 2.63 (The set of minimizers is convex)**.** *The set of minimizers of a convex optimization problem is a convex set.*

Proof. Let f be the objective and C the feasible set. Suppose $x, y \in C$ are both minimizers, so $f(x) = f(y) = p^\star$. Let $\theta \in [0, 1]$. It suffices to show that $(1-\theta)x + \theta y$ is a minimizer, i.e., that $f((1-\theta)x + \theta y) = p^\star$.

Now by convexity

$$f((1-\theta)x + \theta y) \leq (1-\theta)f(x) + \theta f(y) = (1-\theta)p^\star + \theta p^\star = p^\star.$$

But of course $f((1-\theta)x + \theta y) \geq p^\star$ because $p^\star$ is the optimal value. Therefore we have $f((1-\theta)x + \theta y) = p^\star$, as was to be shown. □

Let us provide some illustration of the loopholes allowed by the above results.

Example 2.64. Consider the convex optimization problem with feasible set $C = \mathbb{R}$ and objective $f(x) = x$. The optimal value is $-\infty$.

Example 2.65. Consider the strictly convex optimization problem with feasible set $C = (0, \infty)$ and objective $f(x) = -\log x$. The optimal value is $-\infty$.

Example 2.66. Consider the strictly convex optimization problem with feasible set $C = \mathbb{R}$ and objective $f(x) = e^x$. The optimal value is 0, but it is not attained.

Example 2.67. Consider the convex optimization problem with feasible set $C = \mathbb{R}$ and objective $f(x) = \max(0, x)$. The optimal value is 0, which is attained nonuniquely by all $x \in (-\infty, 0]$.

At least we have a sufficient condition for uniqueness of the minimizer. Can we offer any sufficient condition for existence? There are two convenient options.

Lemma 2.68 (Compact feasible sets admit minimizers)**.** *Consider a convex optimization problem with a compact (i.e., closed and bounded) and nonempty feasible set C and an objective which is continuous on C. Then the optimal value is attained.*

Proof. This follows from the classical real analysis fact that a continuous function on a compact domain attains its minimum. In fact, we do not even need that the optimization problem be convex in order for this result to hold. □

Remark 2.69. More generally, in light of our assumption that the objective is always assumed to be l.s.c. in the sense of Remark 2.59, we can show that any convex optimization problem with a *bounded* and nonempty feasible set C must admit a minimizer.

Indeed, since C is nonempty, there exists $z \in C$. Then with f denoting the objective, we can consider the sublevel set $S := \{x \in C : f(x) \leq f(z)\}$ within C. If we can show that f attains its minimum over S, then it follows that it also attains its minimum over C, since $f(y) > f(z)$ for all $y \in C \backslash S$. Now by the l.s.c. property, the sublevel set S is closed. Since it is contained within C, it is also bounded, hence compact. Then we can consider a minimizing sequence $\{x_k\}_{k=0}^{\infty}$ such that $p^\star = \lim_{k\to\infty} f(x_k)$. (*Why?*) By compactness, we know that there exists a convergent subsequence with limit point x. Replacing $\{x_k\}_{k=0}^{\infty}$ with this subsequence and using the l.s.c. property, we know that $f(x) \leq \lim_{k\to\infty} f(x_k) = p^\star$, hence x is an optimizer. □

Another sufficient condition can be phrased in terms of strong convexity, which guarantees that a function 'bends upward' sufficiently so that its sublevel sets must be compact.

Lemma 2.70 (Strongly convex problems admit minimizers)**.** *The optimal value of a strongly convex optimization problem (with nonempty feasible set) is attained.*

Proof. Let f be the objective, and assume it is μ-strongly convex for some $\mu > 0$. Let C be the feasible set.

The key idea is that strongly convex functions grow sufficiently fast such that their sublevel sets are bounded, hence compact. We will adopt a simplifying assumption that f is differentiable at some point $x_0 \in C$. Then by the remark following Theorem 2.22, we know that

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{\mu}{2}\|x - x_0\|^2 \tag{2.19}$$

for all $x \in C$. (More generally, for any x_0 there always exists some vector, called a subgradient, such that this inequality holds with this subgradient in the place of the gradient. We will come to subgradients later when we discuss non-smooth optimization.)

Now consider the $f(x_0)$-sublevel set

$$S := \{x \in C : f(x) \leq f(x_0)\}$$

within C. It suffices to show that S is compact. (To see why this suffices, see the argument in Remark 2.69.) We know that S is closed because the objective is always assumed to be l.s.c. in the sense of Remark 2.59.

Therefore we only need to show that S is bounded. Suppose that $x \in S$, so $f(x) \leq f(x_0)$. Then by (2.19), we know that

$$\nabla f(x_0) \cdot (x - x_0) + \frac{\mu}{2}\|x - x_0\|^2 \leq 0,$$

and by 'completing the square' we deduce that

$$\|x - x_0 + \mu^{-1}\nabla f(x_0)\| \leq \mu^{-2}\|\nabla f(x_0)\|^2.$$

Thus there exists $y := x_0 - \mu^{-1}\nabla f(x_0)$, $R := \mu^{-1}\|\nabla f(x_0)\|$ such that

$$\|x - y\| \leq R$$

for all $x \in S$, or in other words $S \subset B_R(y)$, the ball of radius R about y. This means that S is bounded. □

Remark 2.71. This argument has a lot in common with our later analysis of gradient descent for strongly convex objectives.

2.4.3 Critical points and the boundary

The following result formalizes the idea that the 'first derivative test' $\nabla f(x) = 0$ is necessary and sufficient for optimality within the *interior* of the feasible set.

Theorem 2.72. *Suppose that the optimizer of a convex optimization problem is attained at a point in the interior of the feasible set where the objective is differentiable. Then this is a critical point. Conversely, any critical point of the objective in the interior of the feasible set must be an optimizer.*

Remark 2.73. The forward direction doesn't require convexity.

Proof. We start with the forward direction. Consider an objective f and feasible set C, and suppose that x_0 is an optimizer in the interior of C where f is differentiable. By differentiability, we know that there exists a function ψ defined on the neighborhood of the origin such that

$$f(x) = f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \psi(x - x_0)\,\|x - x_0\|$$

for all x near x_0 and $\lim_{z\to 0} \psi(z) = 0$.

Now since x_0 lies in the interior of C, for all $\varepsilon > 0$ sufficiently small we have that $x_0 - \varepsilon\nabla f(x_0) \in C$. Plugging in $x = x_0 - \varepsilon\nabla f(x_0)$ and rearranging, we obtain

$$\begin{aligned} f(x) &= f(x_0) - \varepsilon\|\nabla f(x_0)\|^2 + \varepsilon\,\psi(-\varepsilon\nabla f(x_0))\,\|\nabla f(x_0)\| \\ &= f(x_0) - \varepsilon\|\nabla f(x_0)\|\,[\|\nabla f(x_0)\| - \psi(-\varepsilon\nabla f(x_0))]\,. \end{aligned}$$

By optimality, we know that $f(x) \geq f(x_0)$, so we conclude, after substituting and rearranging, that

$$\|\nabla f(x_0)\| \leq \psi(-\varepsilon\nabla f(x_0))$$

for all $\varepsilon > 0$ sufficiently small. Then taking the limit as $\varepsilon \to 0$ estbablishes that $\|\nabla f(x_0)\| = 0$, so x_0 is a critical point.

Now for the reverse direction, consider a convex objective f defined on a convex feasible set C, and suppose that $\nabla f(x_0) = 0$ for some x_0 in the interior of C. By the remark following Theorem 2.22, we know that

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0)$$

for all $x \in C$. But since $\nabla f(x_0) = 0$, this means that $f(x) \geq f(x_0)$ for all $x \in C$, and x_0 is an optimizer. □

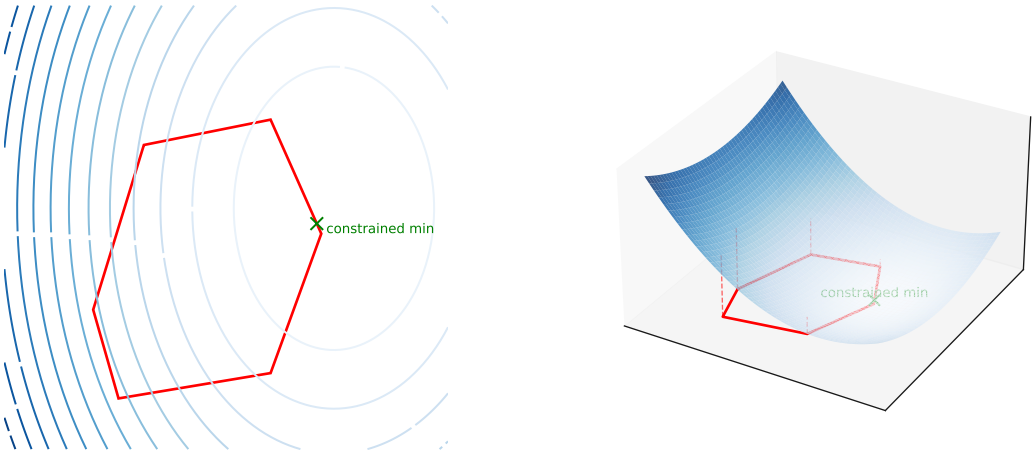

Figure 2.18: Convex optimization problem over a polygonal feasible set. The minimum is attained on the boundary. **Left**: sublevel sets of the obejctive. **Right**: graph of the objective.

In general, we *can* have an optimizer lying on the boundary which is not a critical point, cf. Figure 2.18. We will see how to generalize the first derivative test using Lagrange multipliers when we study constrained optimization problems below.

2.4.4 Removing linear equality constraints

Consider a convex optimization problem in the standard form

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & f_i(x) \leq 0, \quad i = 1, \dots, m, \\ & Ax = b, \end{aligned}$$

where $A \in \mathbb{R}^{p\times n}$, $b \in \mathbb{R}^p$ define the equality constraints. LeWhatt us assume for simplicity that $p < n$, so that the system A is underdetermined, and that A has full rank p, so that $AA^\top \in \mathbb{R}^{p\times p}$ is invertible. (*Why? Recall the SVD.*)

Given a particular solution x_0 solving $Ax_0 = b$, we can always parameterize a feasible point x satisfying $Ax = b$ as $x = x_0 + z$ where $z \in \text{null}(A)$. Now let $B \in \mathbb{R}^{n\times(n-p)}$ be a matrix whose columns form a basis for the null space of A. (*Why must it have $n - p$ columns?*) Then any $z \in \text{null}(A)$ can be written as By for some $y \in \mathbb{R}^{n-p}$, and we can reformulate our equality-constrained optimization problem

$$\begin{aligned} \underset{y\in\mathbb{R}^{n-p}}{\text{minimize}} \quad & f(x_0 + By) \\ \text{subject to} \quad & f_i(x_0 + By) \leq 0, \quad i = 1, \dots, m. \end{aligned}$$

In this equivalent problem, we have removed all equality constraints. See Figure 2.19 for an illustration. The feasible set defined by the new problem may now have a *nonempty interior*, which allows us to directly apply the interior point methods that we shall study later.

Note that removing these equality constraints comes at the price of certain dense linear-algebraic operations and may or may not be a good idea for large-scale problems. For very large problems, interior point methods may become intractable and more scalable (but slower-converging) methods may be required, cf. the discussion in Section 1.1.

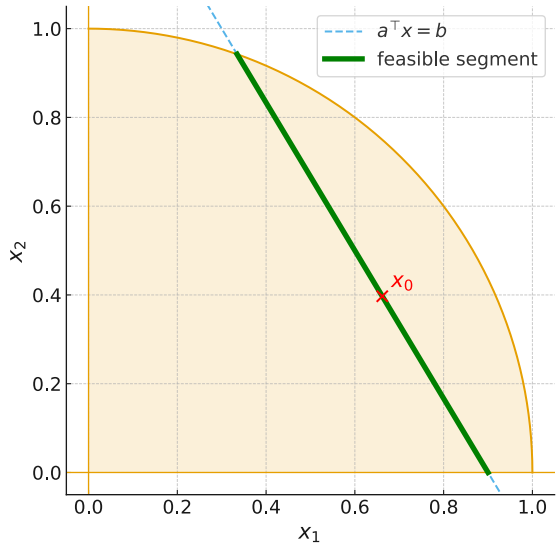

Figure 2.19: Feasible set defined by the inequality constraints $f_1(x) \leq 0$, $f_2(x) \leq 0$, $f_3(x) \leq 0$, where $f_1(x) = -x_1$, $f_2(x) = -x_2$, $f_3(x) = x_1^2 + x_2^2 - 1$, together with a single linear equality constraint of the form $a^\top x = b$. We would like to consider an equivalent 1-dimensional problem defined along the green segment which indicates the feasible set. Here x_0 denotes a particular solution of $a^\top x = b$, specifically the minimum norm solution.

To make things more concrete, observe that in practice, we can always construct a particular solution as $x_0 = A^\top (AA^\top)^{-1} b$. (You can verify directly that this is a solution, but in fact it is the solution with minimal 2-norm. We may rigorously justify this later when we talk about constrained optimization.)

Meanwhile, we can construct B by applying $P = I - A^\top (AA^\top)^{-1} A$, the orthogonal projector onto $\text{null}(A)$, to a random matrix $Z \in \mathbb{R}^{n \times (n-p)}$. Indeed, we can verify directly that $B := PZ = Z - A^\top (AA^\top)^{-1} A \in \mathbb{R}^{n \times (n-p)}$ so defined automatically satisfies $AB = 0$, so the columns of B are in $\text{null}(A)$. We can possibly adjust B by orthonormalizing its columns, via the Gram-Schmidt algorithm or QR factorization, without changing their span.

2.4.5 Examples

Example 2.74 (Linear regression)**.** The linear regression problem defined in Section 2.3.2 with either L2 or L1 regularization defines an unconstrained convex optimization problem. Note that the objective is not smooth in the case of L1 regularization.

Example 2.75 (Softmax regression)**.** The softmax regression problem defined in Section 2.3.3 defines an unconstrained convex optimization problem. More generally, there is a broader class of maximum likelihood estimation problems, for inferring the unknown parameters within a so-called *exponential family*, that can be solved with convex optimization.

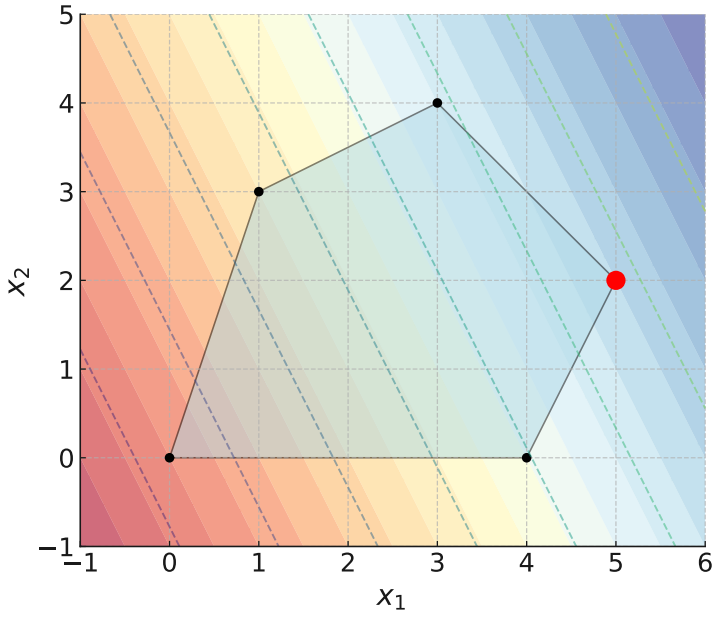

Figure 2.20: Illustration of the fact that the optimal value of a linear program is always attained (typically uniquely) at an extremal point. (It takes a little more work to state this fact carefully, but hopefully you get the idea.) The colors indicate level sets of the linear objective, and the polygonal region depicts the feasible set. The optimal value is attained at the red vertex.

Definition 2.76 (Linear program)**.** A linear program is a convex optimization problem in which the objective function is linear and the inequality constraints are all affine-linear.

Example 2.77 (Diet problem)**.** This is one of the oldest examples of a linear program and actually motivated the development of the first numerical method for linear programming, since it could not be solved by hand. It conveys the flavor of many linear programs motivated by industrial optimization and resource allocation.

Suppose that we want to construct a diet consisting of a combination of n different foods. The j-th entry $x_j \geq 0$ of our optimization $x \in \mathbb{R}^n$ denotes the amount of the j-th food to be included in the diet, which costs c_j per unit, $j = 1, \ldots, n$. Also suppose that we want to ensure that our diet contains a sufficient amount of each of m possible nutrients. We let b_i denote the minimum allowed amount for the i-th nutrient, $i = 1, \ldots, m$. Finally, for each i, j, let a_{ij} be the amount of nutrient i contained in a unit of food j. Then consider linear program

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & c^\top x \\ \text{subject to} \quad & Ax \geq b, \\ & x \geq 0, \end{aligned}$$

where $A = (a_{ij})$, $b = (b_i)$, and $c = (c_j)$ and the inequalities are interpreted entrywise. The problem seeks the lowest-cost diet that achieves our nutrient requirements.

The following two exercises study different ways of solving overdetermined linear systems, using

the 1-norm and ∞-norm in place of the 2-norm as an error penalty. In fact both of these choices lead to linear programs! As part of these exercises, you will practice putting these linear programs into a standard form, which is a preprocessing step that will allow us to focus later on designing a *general* solver for linear programs.

Exercise 2.78. Consider the following approach to 'solving' an overdetermined linear system $Ax = b$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, as an alternative to least squares:

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \|Ax - b\|_\infty.$$

Show that it is equivalent to the following linear program

$$\begin{aligned} \underset{x \in \mathbb{R}^n,\, t \in \mathbb{R}}{\text{minimize}} \quad & t \\ \text{subject to} \quad & Ax - b \leq t\,\mathbf{1}_m, \\ & b - Ax \leq t\,\mathbf{1}_m, \end{aligned}$$

over the composite optimization variable $y = (x^\top, t)^\top \in \mathbb{R}^{n+1}$. Here $\mathbf{1}_m \in \mathbb{R}^m$ denotes the vector of all ones. Show how to construct c, d, and C of appropriate shapes so that this linear program can be written in the form:

$$\begin{aligned} \underset{y \in \mathbb{R}^{n+1}}{\text{minimize}} \quad & c \cdot y \\ \text{subject to} \quad & Cy \leq d. \end{aligned}$$

Exercise 2.79. Consider the problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \quad \|Ax - b\|_1,$$

where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Show that it is equivalent to the following linear program

$$\begin{aligned} \underset{x \in \mathbb{R}^n,\, z \in \mathbb{R}^m}{\text{minimize}} \quad & \sum_{i=1}^{m} z_i \\ \text{subject to} \quad & Ax - b \leq z, \\ & b - Ax \leq z, \end{aligned}$$

over the composite optimization variable $y = (x^\top, z^\top)^\top$. Show how to construct c, d, and C of appropriate shapes so that this linear program can be written in the form:

$$\begin{aligned} \underset{y \in \mathbb{R}^{n+m}}{\text{minimize}} \quad & c \cdot y \\ \text{subject to} \quad & Cy \leq d. \end{aligned}$$

Exercise 2.80. Consider a general linear program

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & c \cdot x \\ \text{subject to} \quad & Cx \leq d, \\ & Ax = b, \end{aligned}$$

where $c \in \mathbb{R}^n$, $C \in \mathbb{R}^{m \times n}$, $d \in \mathbb{R}^m$, $A \in \mathbb{R}^{p \times n}$, $b \in \mathbb{R}^p$. You can assume that $p < n$ and that A has full rank. Show how, by eliminating equality constraints, we can construct a solution in terms of the solution of a linear program of the form

$$\begin{aligned} \underset{y \in \mathbb{R}^{n-p}}{\text{minimize}} \quad & \tilde{c} \cdot y \\ \text{subject to} \quad & \tilde{C}y \leq \tilde{d}. \end{aligned}$$

In particular, specify how $\tilde{c}$, $\tilde{C}$, and $\tilde{d}$ should be constructed.

Example 2.81 (Portfolio optimization)**.** The following convex optimization problem is known as the Markowitz model for portfolio optimization:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & x^\top \Sigma x \\ \text{subject to} \quad & r^\top x \geq R_{\min}, \\ & \mathbf{1}_n^\top x = 1, \\ & x \geq 0. \end{aligned}$$

Here x is a vector indicating our allocation of a fixed amount of money (1 unit total) among n different assets, r is a vector of expected returns for each of the assets, and we constrain our choice of x to deliver a minimum expected return of $R_{\min}$. Subject to this constraint, we minimize the variance of our return, which is computed in terms of Σ, the (PSD) covariance matrix of the asset returns.

Example 2.82 (Optimal transport)**.** Suppose there are m bakeries and n cafes in a town. Every day, the i-th bakery produces a supply of $\mu_i \geq 0$ bagels for $i = 1, \ldots, m$, and the j-th cafe demands $\nu_j \geq 0$ bagels for $j = 1, \ldots, n$. The supply and demand are evenly matched, i.e., $\sum_{i=1}^m \mu_i = \sum_{j=1}^n \nu_j$, and the cost of transporting a bagel from the i-th bakery to the j-th cafe is C_{ij}. The optimal transport problem finds the transportation plan π that allocates the bagels from bakeries to cafes in the most cost-effective way:

$$\begin{aligned} \underset{\pi \in \mathbb{R}^{m \times n}}{\text{minimize}} \quad & \sum_{i=1}^m \sum_{j=1}^n C_{ij} \pi_{ij} \\ \text{subject to} \quad & \pi \geq 0, \\ & \pi \mathbf{1}_n = \mu, \\ & \pi^\top \mathbf{1}_m = \nu. \end{aligned} \tag{2.20}$$

Note that this is a linear program. We interpret $\pi_{ij} \geq 0$ as the number of bagels sent from the i-th bakery to the j-th cafe in our plan. (Unfortunately, we may have to allow for fractional bagels.) The equality constraints specify that $\sum_{j=1}^n \pi_{ij} = \mu_i$ for all $i = 1, \ldots, m$ (i.e., the i-th bakery exports all of the bagels it produces) and $\sum_{i=1}^m \pi_{ij} = \nu_j$ for all $j = 1, \ldots, n$ (i.e., the j-th cafe imports all the bagels it demands).

In fact this optimization problem has a much deeper significance in probability theory, where it allows us to define useful metrics between probability distributions, and it has countless applications in modern data science. In practice it is often useful to consider an 'entropy-regularized' version of the problem, which includes the Shannon entropy

$$S(\pi) = \sum_{i,j} \pi_{ij} \log \pi_{ij}$$

as a regularization term:

$$\begin{aligned} \underset{\pi \in \mathbb{R}^{m \times n}}{\text{minimize}} \quad & \sum_{i=1}^m \sum_{j=1}^n C_{ij} \pi_{ij} + \beta^{-1} S(\pi) \\ \text{subject to} \quad & \pi \geq 0, \\ & \pi \mathbf{1}_n = \mu, \\ & \pi^\top \mathbf{1}_m = \nu. \end{aligned} \tag{2.21}$$

Here $\beta \in (0, \infty)$ is a regularization parameter, and the unregularized problem is recovered as $\beta \to +\infty$. By Exercise 2.36, this is still a convex optimization problem. We will see that the dual problem has a special structure that permits a fast specialized solver.

Exercise 2.83. Show that the optimal transport problem is always feasible, for arbitrary C, $\mu \geq 0$, and $\nu \geq 0$ satisfying $\sum_{i=1}^m \mu_i = \sum_{j=1}^n \nu_j$. **Hint:** Try to construct a feasible point π as a rank one matrix. What is the intepretation of this feasible point?

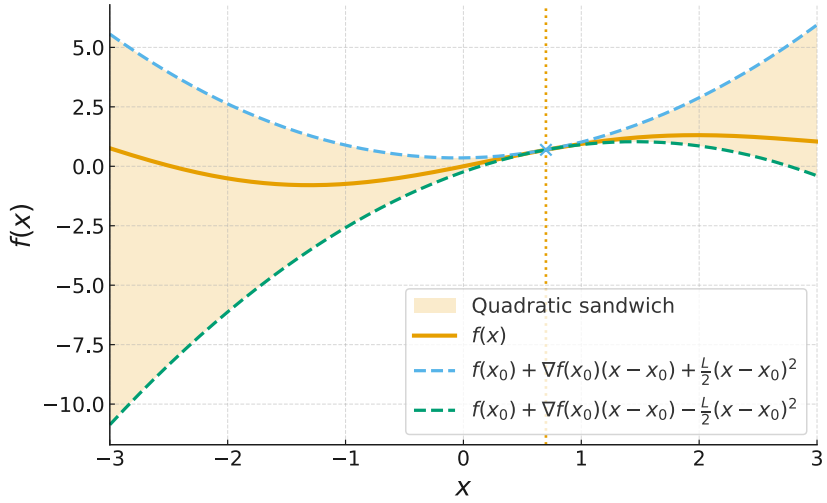

Figure 3.1: Illustration of Theorem 3.3, the basic consequence of L-smoothness.

3 Smoothness

In our analysis of gradient descent, a quantitative notion of *smoothness* will appear. We prepare for this analysis in this section. Note that in optimization theory, 'smoothness' has a very particular meaning which is different from the typical meaning that a function possesses arbitrarily many derivatives.

3.1 Definition and basis consequences

Definition 3.1 (L-smoothness)**.** A differentiable function $f : C \to \mathbb{R}$ defined on an open domain $C \subset \mathbb{R}^n$ is L-smooth for some $L \geq 0$ if $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$ for all $x, y \in C$.

Remark 3.2. It is equivalent to say that the gradient $\nabla f : C \to \mathbb{R}^n$ is L-Lipschitz.

This notion of smoothness controls the discrepancy between a function and its linear approximation.

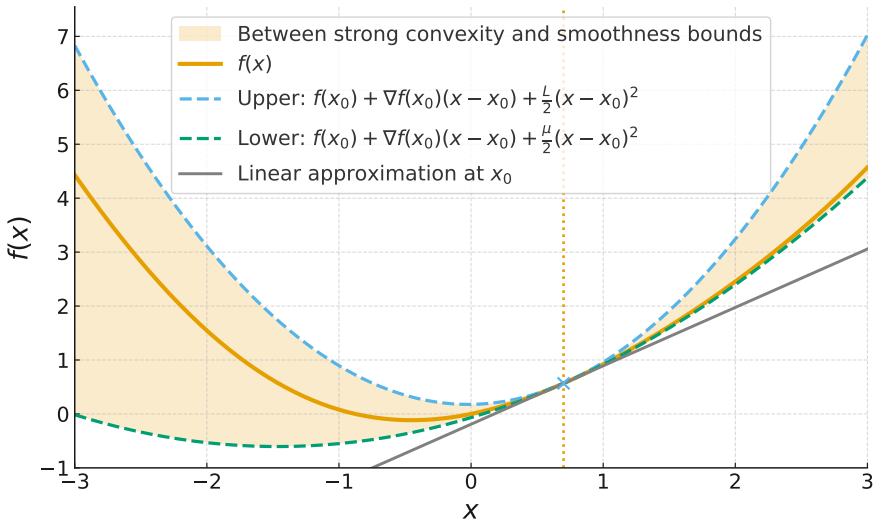

Figure 3.2: Illustration of Remark 3.4, depicting the upper and lower bounds furnished by L-smoothness together with μ-strong convexity.

Theorem 3.3. *Suppose that $f : C \to \mathbb{R}$ defined on an open convex domain $C \subset \mathbb{R}^n$ is L-smooth. Then*

$$|f(x) - [f(x_0) + \nabla f(x_0) \cdot (x - x_0)]| \leq \frac{L}{2}\|x - x_0\|^2$$

for all $x_0, x \in C$.

Remark 3.4. If f is μ-strongly convex (possibly with $\mu = 0$), we automatically know the lower bound

$$f(x) \geq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{\mu}{2}\|x - x_0\|^2,$$

which is stronger than the lower bound on $f(x)$ furnished by L-smoothness alone. Already given convexity, what L-smoothness really buys us is the *upper* bound

$$f(x) \leq f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \frac{L}{2}\|x - x_0\|^2.$$

Together these conditions tell us how tightly we can squeeze our function $f(x)$ between two quadratics.

Proof. Let $x_0, x \in C$ and define $x_t := x_0 + t(x - x_0) = (1 - t)x_0 + tx \in C$ for $t \in [0, 1]$. Then use

the fundamental theorem of calculus to write

$$\begin{aligned} f(x) &= f(x_0) + \int_0^1 \frac{d}{dt}\left[f(x_t)\right]\, dt \\ &= f(x_0) + \int_0^1 \nabla f(x_t)\cdot(x-x_0)\, dt \\ &= f(x_0) + \nabla f(x_0)\cdot(x-x_0) + \int_0^1 \left[\nabla f(x_t) - \nabla f(x_0)\right]\cdot(x-x_0)\, dt, \end{aligned}$$

where in the last equation we have added and subtracted $\nabla f(x_0)\cdot(x-x_0)$ and absorbed the subtracted copy inside the integral, since it is independent of t.

Then we can rearrange and use the triangle inequality to deduce:

$$\begin{aligned} |f(x) - [f(x_0) + \nabla f(x_0)\cdot(x-x_0)]| &\le \int_0^1 |[\nabla f(x_t) - \nabla f(x_0)]\cdot(x-x_0)|\, dt \\ &\overset{\text{(i)}}{\le} \int_0^1 \|\nabla f(x_t) - \nabla f(x_0)\|\, \|x-x_0\|\, dt \\ &\overset{\text{(ii)}}{\le} L\int_0^1 \|x_t - x_0\|\, \|x-x_0\| \\ &\overset{\text{(iii)}}{\le} L\|x-x_0\|^2 \int_0^1 t\, dt \\ &= \frac{L}{2}\|x-x_0\|^2, \end{aligned}$$

where in (i) we have used Cauchy-Schwarz, in (ii) we have used the definition of L-smoothness, and in (iii) we have used the fact that $x_t - x_0 = t(x-x_0)$. Finally in the last step we have just computed a definite integral. This completes the proof. □

3.2 Characterization in terms of the Hessian

We will see that much like μ-strong convexity, L-smoothness can be characterized equivalently in terms of the Hessian.

Definition 3.5 (Operator norm)**.** Given a norm $\|\cdot\|$ on $\mathbb{R}^n$, we use the same notation to denote the associated *operator norm* or *induced norm*, defined on arbitrary $A \in \mathbb{R}^{n\times n}$ as

$$\|A\| := \max_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \max_{\|z\|=1} \|Az\|.$$

In the case where $\|\cdot\| = \|\cdot\|_2$, the operator norm is called the *spectral norm*. For simplicity of notation, we may use $\|\cdot\|$ to denote this special case, as we have been doing for the 2-norm throughout, unless otherwise specified.

Exercise 3.6. Show that the two expressions appearing above in the definition of $\|A\|$ are actually equal, for an arbitrary norm $\|\cdot\|$.

Exercise 3.7 (Matrix-vector sub-multiplicativity)**.** Show that for an arbitrary norm $\|\cdot\|$, we have $\|Ax\| \leq \|A\| \, \|x\|$ for all $A \in \mathbb{R}^{n \times n}$ and $x \in \mathbb{R}^n$. (*This is the functionally important property of the operator norm.*)

Exercise 3.8 (Spectral norm as top singular value)**.** Use the SVD to show that $\|A\|_2$ is equal to the largest singular value of $A \in \mathbb{R}^{n \times n}$. **Hint:** It is helpful to expand $\|Az\|_2^2 = z^\top A^\top A z$. You may also want to use the fact that $\|Ox\|_2 = \|x\|_2$ for an orthogonal matrix O and any $x \in \mathbb{R}^n$. (Why is this true?)

Exercise 3.9. Use the spectral theorem to show that for a symmetric matrix $A \in \mathbb{R}^{n \times n}$,

$$\|A\|_2 = \max\{|\lambda| \, : \, \lambda \text{ is an eigenvalue of } A\}.$$

Hint: Same as last exercise.

Exercise 3.10 (The operator norm is a norm)**.** Show that for any norm $\|\cdot\|$ on $\mathbb{R}^n$, the operator norm actually defines a norm on $\mathbb{R}^{n \times n}$, in the sense of Definition 2.41 suitably interpreted.

Theorem 3.11 (L-smoothness in terms of the Hessian)**.** *A twice differentiable function $f : C \to \mathbb{R}$ defined on an open convex domain $C \subset \mathbb{R}^n$ is L-smooth if and only if $\|\nabla^2 f(x)\|_2 \leq L$ for all $x \in C$.*

Remark 3.12. In other words, L-smoothness tells us that all eigenvalues of the Hessian matrix $\nabla^2 f(x)$ are all bounded by L in absolute value,. Convexity already tells us that the eigenvalues are nonnegative, and μ-strong convexity even tells us that they are bounded below by μ. Hence L-smoothness and μ-strong convexity together tell us that the eigenvalues of the Hessian all lie in $[\mu, L]$.

Proof. First we prove the backward direction. Thus we suppose $f : C \to \mathbb{R}$ satisfies $\|\nabla^2 f(x)\| \leq L$ for all $x \in C$. We let $x, y \in C$, and we want to show that $\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|$.

Let $x_t = x + t(y - x) \in C$ for $t \in [0, 1]$. Then use the fundamental theorem of calculus to write

$$\begin{aligned} \nabla f(y) &= \nabla f(x) + \int_0^1 \frac{d}{dt} \left[\nabla f(x_t)\right] \, dt \\ &= \nabla f(x) + \int_0^1 \nabla^2 f(x_t) \, (y - x) \, dt. \end{aligned}$$

It is a good exercise in the chain rule to verify the second step. Then rearranging and applying the triangle inequality we deduce that

$$\begin{aligned} \|\nabla f(x) - \nabla f(y)\| &\leq \int_0^1 \|\nabla^2 f(x_t) \, (y - x)\| \, dt \\ &\leq \int_0^1 \|\nabla^2 f(x_t)\| \, \|y - x\| \, dt \\ &\leq L \, \|x - y\|, \end{aligned}$$

where in the penultimate step we used Exercise 3.7. This completes the proof of the backward direction.

Now for the forward direction, suppose $f : C \to \mathbb{R}$ is L-smooth and twice differentiable. Let $x_0 \in C$. We want to show that $\|\nabla^2 f(x_0)\| \leq L$.

Now the differentiability of the gradient means precisely that for x near x_0, we have

$$\nabla f(x) = \nabla f(x_0) + \nabla^2 f(x_0)\,(x - x_0) + \|x - x_0\|\Psi(x - x_0)$$

where Ψ is now a vector-valued function defined near the origin such that $\lim_{z\to 0} \Psi(z) = 0$. (This fact follows from applying the definition of differentiability to each component $\partial f/\partial x_i$ of the gradient.)

Now we rearrange, take norms, apply the triangle inequality, and use L-smoothness to bound

$$\begin{aligned}\|\nabla^2 f(x_0)\,(x - x_0)\| &= \|\nabla f(x) - \nabla f(x_0) - \|x - x_0\|\Psi(x - x_0)\| \\ &\leq \|\nabla f(x) - \nabla f(x_0)\| + \|x - x_0\|\,\|\Psi(x - x_0)\| \\ &\leq \|x - x_0\|\,(L + \|\Psi(x - x_0)\|)\end{aligned}$$

for all x near x_0.

Then plugging in $x = x_0 + tz$ for all $t \neq 0$ sufficiently small and $z \in \mathbb{R}^n$ an arbitrary vector with $\|z\| = 1$, we see that

$$|t|\,\|\nabla^2 f(x_0)\,z\| \leq |t|\,(L + \|\Psi(tz)\|)\,.$$

Then dividing by $|t|$ and taking the limit as $t \to 0$ we deduce that

$$\|\nabla^2 f(x_0)\,z\| \leq L.$$

Recall that z was an arbitrary unit vector, so by the definition of the operator norm $\|\nabla^2 f(x_0)\| \leq L$, as was to be shown. □

It is useful to think of L-smoothness in terms of a model quadratic function:

Corollary 3.13. *The arbitrary quadratic function $f(x) = \frac{1}{2}x^\top Ax + b \cdot x + c$ where $A \in \mathbb{R}^{n\times n}$, $b \in \mathbb{R}^n$, $c \in \mathbb{R}$, is L-smooth for $L \geq 0$ if and only if $\|A\|_2 \leq L$.*

Proof. We simply compute $\nabla^2 f \equiv A$ and apply the preceding theorem. □

We also provide some examples of how L-smoothness can fail to hold for any $L \geq 0$, even for continuously differentiable f. The first shows how L-smoothness can fail to hold due to a *local* obstruction.

Example 3.14. Consider $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = |x|^{3/2}$. Note that f is differentiable and $f'(x) = \frac{3}{2}\text{sign}(x)\sqrt{|x|}$. Moreover $f''(x) = \frac{3}{4}\frac{1}{\sqrt{|x|}}$ is defined for $x \neq 0$. Since $|f''|$ is unbounded on $(0, \infty)$, by Theorem 3.11 there does not exist any $L \geq 0$ such that f is L-smooth on $(0, \infty)$, hence neither can f be L-smooth on the larger set $\mathbb{R}$.

The next example demonstrates how L-smoothness can fail due to fast asymptotic growth.

Example 3.15. Consider $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x^3$. Then $|f''(x)| = 6|x|$ is unbounded over $x \in \mathbb{R}$, so by Theorem 3.11 f cannot be L-smooth on $\mathbb{R}$ for any $L \geq 0$.

Part II
Gradient descent and its variants

We are finally ready to introduce and analyze some algorithms for optimization. First we consider unconstrained optimization problems, in which the feasible set is simply $\mathbb{R}^n$ and we denote the objective by $f : \mathbb{R}^n \to \mathbb{R}$. As usual, the optimal value will be denoted by $p^\star$, and if the optimal value is attained, we will use $x^\star \in \mathbb{R}^n$ to denote some optimizer.

4 Gradient descent

We start with gradient descent. Like all the algorithms we will consider, **gradient descent** is an *iterative algorithm* defining a sequence $\{x_t\}_{t=0}^\infty$ of *iterates* $x_t \in \mathbb{R}^n$.

Given an *initial guess* $x_0 \in \mathbb{R}^n$, the sequence is completely specified by the update rule

$$x_{t+1} = x_t - \eta \nabla f(x_t), \quad t = 0, 1, 2, \ldots \tag{4.1}$$

Here $\eta > 0$ is a *step size* (or *learning rate*), which we take to be a fixed 'hyperparameter' of the method. We will study the conditions on $\eta > 0$ (in terms of the objective f) that we need to guarantee convergence.

A practical pseudocode is offered in Algorithm 1. It gives some idea of how to write an implementation with a maximum number of iterations and a reasonable (if not canonical) stopping condition. This is just one implementation. It may make sense to store the entire history of $f(x)$ and $\|\nabla f(x)\|$, for example, over the optimization trajectory.

Algorithm 1 Gradient descent (fixed step size)

Input: $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$, initial guess $x_0 \in \mathbb{R}^n$, step size $\eta > 0$, maximum number of iterations T, residual tolerance ε

Output: Estimates for optimal x and $f(x)$, convergence flag

```
Set x ← x_0 and flag ← false
for t = 1, ..., T do
    Set v ← ∇f(x)
    Set x ← x − η v
    if ‖v‖ ≤ ε then
        flag ← true
        break
    end if
end for
return x, f(x), flag
```

For most of our careful analysis, we will view η as fixed *a priori*. However, it is also possible to take the step size η_t to be dependent on the iteration counter t, yielding the update

$$x_{t+1} = x_t - \eta_t \nabla f(x_t), \quad t = 0, 1, 2, \ldots \tag{4.2}$$

A choice of sequence $\{\eta_t\}_{t=0}^\infty$ can be fixed *a priori*, in which case we may say that it is specified *offline*. But it can also be chosen *online*, e.g., via a line search.

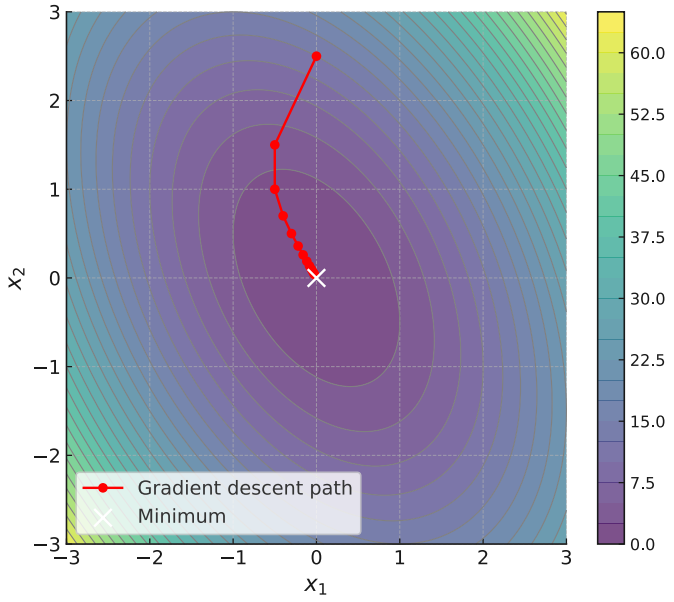

Figure 4.1: Illustration of gradient descent (4.1). The update direction is always perpendicular to the level set of the current iterate.

4.1 Modern implementation

In recent years, automatic differentiation has transformed the practical implementation of optimization algorithms, especially in machine learning. Several machine learning libraries such as JAX have emerged, allow for easy automatic differentiation while making efficient use of hardware acceleration for batched operations. This **Colab notebook** is a JAX-based introduction to several key operations of widespread use. It covers applications to regression and classification introduced in Section 2.3.

4.2 The descent lemma

The *descent lemma* guarantees that gradient descent decreases the objective for smooth functions, provided that the step size is sufficiently small. It is of fundamental importance in our analysis of gradient descent.

Lemma 4.1 (Descent lemma)**.** *If f is L-smooth on $\mathbb{R}^n$, then the iterates of gradient descent (4.1) satisfy*

$$f(x_{t+1}) \leq f(x_t) - \eta \left(1 - \frac{\eta L}{2}\right) \|\nabla f(x_t)\|^2, \quad t = 0, 1, 2, \ldots$$

In particular, if we choose $\eta \leq \frac{1}{L}$, we have

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta}{2} \|\nabla f(x_t)\|^2, \quad t = 0, 1, 2, \ldots$$

Remark 4.2. The descent lemma gives us guidance on choosing the step size in terms of the smoothness parameter L. (In general, if we take $\eta > \frac{2}{L}$, gradient descent may diverge.) Note that if a function is L-smooth, then it is also L'-smooth for any $L' \geq L$. Therefore, for simplicity we will often just take $\eta = \frac{1}{L}$ in our analysis.

Proof. By Theorem 3.3,

$$f(x_{t+1}) \leq f(x_t) + \nabla f(x_t) \cdot (x_{t+1} - x_t) + \frac{L}{2}\|x_{t+1} - x_t\|^2.$$

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

$$f(x_{t+1}) \leq f(x_t) - \eta\|\nabla f(x_t)\|^2 + \frac{L\eta^2}{2}\|\nabla f(x_t)\|^2.$$

The desired result follows via simplification by factoring. □

The descent lemma directly implies that gradient descent finds critical points of smooth functions. This theorem does not require any convexity!

Note that any critical point is a *fixed point* of the gradient descent iteration, i.e., if x_t is a critical point, then $\nabla f(x_t) = 0$ and automatically $x_{t+1} = x_t$ by (4.1). Thus gradient descent stays in place forever if it ever hits a critical point.

In fact, gradient descent will generically converge to *local minima*, which are special critical points. This makes sense because gradient descent is always trying to find a direction that locally decreases the value of the function. However, this fact is harder to formulate and prove because one has to rule out non-generic behavior (i.e., landing right on a critical point that is not a local minimum).

Now we present our theorem about convergence to critical points. The statement is a bit subtle, so read carefully.

Theorem 4.3 (Gradient descent finds critical points)**.** *Let f be L-smooth on $\mathbb{R}^n$ and consider gradient descent (4.1) with step size $\eta = \frac{1}{L}$. For any positive integer T, there exists some $t \in \{0, 1, \ldots, T-1\}$ such that*

$$\|\nabla f(x_t)\|^2 \leq \frac{2L}{T}\left[f(x_0) - p^\star\right].$$

Remark 4.4. One way of phrasing the result is as follows. If we want to find x such that $\|\nabla f(x)\| \leq \varepsilon$, where $\varepsilon > 0$ is arbitrary, we can always achive this by running gradient descent and terminating when we hit an iterate x_t such that $\|\nabla f(x_t)\| \leq \varepsilon$. The theorem says that we will terminate by the time we reach the iterate number $t = T - 1$, where

$$T = \left\lceil \frac{2L}{\varepsilon^2}\left[f(x_0) - p^\star\right] \right\rceil.$$

Remark 4.5. Note that $f(x_0) - p^\star \geq 0$ is the difference between the initial objective to the optimal value. If $p^\star = -\infty$, then the theorem holds vacuously, so we may assume that $p^\star$ is finite in the proof.

Proof. By rearrangement of the descent lemma (Lemma 4.1), we deduce that

$$f(x_t) - f(x_{t+1}) \geq \frac{1}{2L}\|\nabla f(x_t)\|^2, \quad t = 0, 1, 2, \ldots$$

Taking the sum of both sides from $t = 0$ to $T - 1$, we see that

$$\sum_{t=0}^{T-1} [f(x_t) - f(x_{t+1})] \geq \frac{1}{2L} \sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2.$$

The left-hand side is a telescoping sum, yielding

$$\frac{1}{2L} \sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq f(x_0) - f(x_T) \leq f(x_0) - p^\star,$$

where we have used the fact that $f(x_T) \geq p^\star$. Then in turn

$$\frac{1}{T} \sum_{t=0}^{T-1} \|\nabla f(x_t)\|^2 \leq \frac{2L}{T} \left[f(x_0) - p^\star\right].$$

Now the left-hand side is the average value of $\|\nabla f(x_t)\|^2$ over the first T iterations, which must be greater than minimum value. Thus there exists some minimizing index $t \in \{0, 1, \ldots, T-1\}$ such that

$$\|\nabla f(x_t)\|^2 \leq \frac{2L}{T} \left[f(x_0) - p^\star\right].$$

This completes the proof. □

Example 4.6 (Gradient descent on a positive definite quadratic form)**.** The following problem is useful to keep in mind, since all functions with enough derivatives look similar to such a model quadratic near a (strict) local optimizer, where the Hessian must be positive definite. Concretely, consider the objective

$$f(x) = \frac{1}{2} x^\top A x - b \cdot x,$$

where A is positive definite. Note that f is L-smooth with $L = \|A\|_2$, cf. Theorem 3.3 and Corollary 3.11. By Exercise 3.9, all of the eigenvalues of A are bounded by L in absolute value. But since A is positive definite, we know more specifically that the eigenvalues of A all lie in $(0, L]$. Moreover note that

$$x^\star = A^{-1} b \tag{4.3}$$

defines the unique optimizer.

Now let us choose the initial condition $x_0 = 0$ for simplicity. Now $\nabla f(x) = Ax - b$, so gradient descent (4.1) reads as

$$x_{t+1} = x_t - \eta(Ax_t - b) = (I - \eta A)x_t + \eta b,$$

or equivalently

$$x_{t+1} = Tx_t + c,$$

where

$$T := I - \eta A, \quad c := \eta b. \tag{4.4}$$

We can then compute the first few terms as

$$x_1 = Tx_0 + c = c,$$

$$x_2 = Tx_1 + c = Tc + c,$$

$$x_3 = Tx_2 + c = T^2 c + Tc + c,$$

etc., and deduce the pattern

$$x_t = \left(\sum_{k=0}^{t-1} T^k \right) c,$$

which can be verified by induction. (Note that $T^0 = I$.) Finally we can use the formula for the sum of a finite geometric series to compute

$$x_t = (I - T^t)(I - T)^{-1} c.$$

Substituting (3.9) and (4.3), we see that

$$x_t = (I - T^t)\, x^\star.$$

Therefore the convergence of gradient descent to the optimal solution amounts to the question of whether we have $\lim_{t \to \infty} T^t = 0$.

Note that the eigenvalues of $T = I - \eta A$ lie in $[1 - \eta L, 1)$ because the eigenvalues of A lie in $(0, L]$. Diagonalizing $T = U \Lambda U^\top$ by the spectral theorem and noting that $T^t = U \Lambda^t U^\top$, we see that $\lim_{t \to \infty} T^t = 0$ if and only if all the eigenvalues of T lie in $(-1, 1)$, which holds provided that $\eta L < 2$, or $\eta < \frac{2}{\|A\|_2}$. This is precisely the condition required for strict descent in the descent lemma (Lemma 4.1). In general, if $\eta > \frac{2}{\|A\|_2}$, gradient descent can certainly diverge!

4.3 Variational interpetation of gradient descent

Next we introduce another fundamental perspective on gradient descent. To wit, we will rederive the same update from a different point of view.

Suppose our current guess for the optimizer is given by $y \in \mathbb{R}^n$ and we want to update to improve the objective. Consider optimizing the following proxy function, defined in terms of the basepoint y and an arbitary $\eta > 0$:

$$f_{\eta,y}(x) := f(y) + \nabla f(y) \cdot (x - y) + \frac{1}{2\eta}\|x - y\|^2, \tag{4.5}$$

consisting of a linear approximation of f near the guess y, plus a *proximal-point penalty* that prevents us from taking too large of a step. (If there were no such penalty, the proxy $f_{\eta,y}$ would be affine-linear, hence unbounded below in general.) Optimization of the proxy can be performed exactly as follows.

Lemma 4.7. *Let f be differentiable on $\mathbb{R}^n$, $y \in \mathbb{R}^n$, and $\eta > 0$. Then*

$$y - \eta\nabla f(y) = \operatorname*{argmin}_{x\in\mathbb{R}^n} \{f_{\eta,y}(x)\}.$$

Moreover, the optimal value is given by

$$\min_{x\in\mathbb{R}^n} \{f_{\eta,y}(x)\} = f(y) - \frac{\eta}{2}\|\nabla f(y)\|^2.$$

Remark 4.8. Throughout we will use 'argmin' to denote an optimizer of the objective appearing within. It is often implied that the optimizer is unique, and we show that uniqueness holds here in the proof of the lemma.

Proof. For any $y \in \mathbb{R}^n$ and $\eta > 0$, let us define $f_{\eta,y} : \mathbb{R}^n \to \mathbb{R}$ by

$$f_{\eta,y}(x) := f(y) + \nabla f(y) \cdot (x - y) + \frac{1}{2\eta}\|x - y\|^2.$$

Then observe that $f_{\eta,y}$ is η^{-1}-strongly convex. (*Why?*) Hence by Lemma 2.62, Lemma 2.70, and Theorem 2.72, the objective $f_{\eta,y}$ admits a unique minimizer which is the unique critical point. We can solve for the critical point by computing the gradient

$$\nabla f_{\eta,y}(x) = \nabla f(y) + \frac{1}{\eta}(x - y)$$

and setting it equal to zero, yielding

$$x = y - \eta\nabla f(y).$$

We deduce the optimal value by plugging in

$$\begin{aligned} f_{\eta,y}(y - \eta\nabla f(y)) &= f(y) - \nabla f(y) \cdot [-\eta\nabla f(y)] + \frac{1}{2\eta}\| - \eta\nabla f(y)\|^2 \\ &= f(y) - \eta\|\nabla f(y)\|^2 + \frac{\eta}{2}\|\nabla f(y)\|^2 \\ &= f(y) - \frac{\eta}{2}\|\nabla f(y)\|^2. \end{aligned}$$

This completes the proof. □

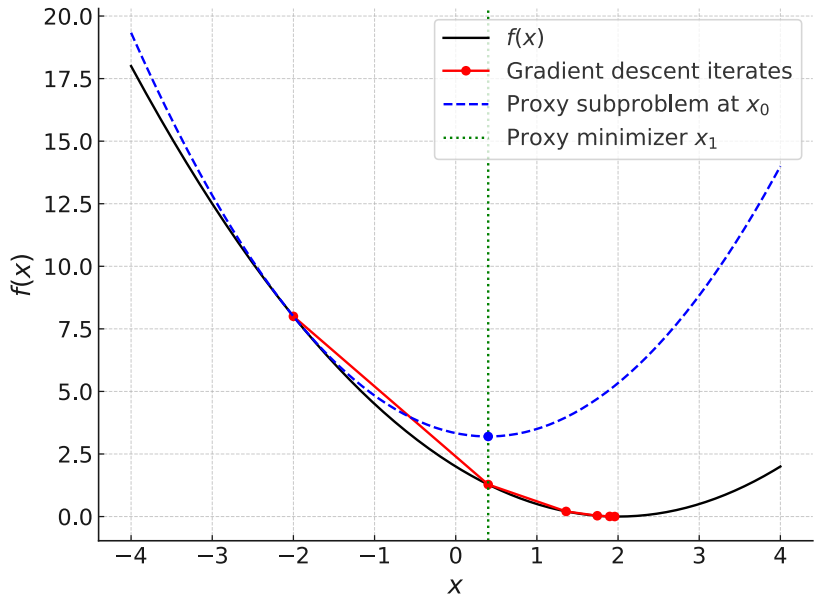

Figure 4.2: Illustration of the variational interpretation of gradient descent, Lemma 4.9.

Thus, exact optimization of the proxy $f_{\eta,y}$ precisely performs a gradient descent update from y with step size η. We state this fact formally.

Corollary 4.9 (Variational interpretation of gradient descent)**.** *If f is differentiable on $\mathbb{R}^n$, then the iterates of gradient descent (4.1) satisfy*

$$x_{t+1} = \operatorname*{argmin}_{x\in\mathbb{R}^n} \{f_{\eta,x_t}(x)\} = \operatorname*{argmin}_{x\in\mathbb{R}^n} \left\{ f(x_t) + \nabla f(x_t)\cdot(x-x_t) + \frac{1}{2\eta}\|x-x_t\|^2 \right\}.$$

Remark 4.10. This interpretation of gradient descent provides the inspiration for a broad generalization, namely the proximal gradient method, which we will consider later.

Note that by Theorem 3.3, if f is L-smooth and $\eta \le \frac{1}{L}$, then in fact $f_{\eta,y}(x) \ge f(x)$ for all $x \in \mathbb{R}^n$, while $f_{\eta,y}$ agrees with f exactly up to linear order near y. In the optimization literature, we say that $f_{\eta,y}$ *majorizes* f, and we can interpret gradient descent with $\eta \le \frac{1}{L}$ as a **majorization-minimization** algorithm. See Figure 4.2 for illustration.

4.4 Optimality metrics

The last theorem raises an interesting question. As we run gradient descent, how should we measure our convergence quantitatively? Given a guess $x \in \mathbb{R}^n$, there are three key quantities to consider.

- **Optimality gap:** $f(x) - f(x^\star)$
- **Distance to optimizer:** $\|x - x^\star\|$
- **Gradient norm:** $\|\nabla f(x)\|$

Note that since $x^\star$ is unknown to us *a priori*, only the last of these is accessible to us as we run the algorithm. In the quadratic case (Example 4.6), the gradient norm corresponds to the residual norm $\|Ax - b\|$, which is the practical optimality metric used to measure progress in iterative solvers for linear systems.

Now if f is strictly convex, then all three of these quantities are zero if and only if $x = x^\star$, i.e., x is optimal. But we are interested in how each can be used to quantitatively control the others. Such quantitative control can be phrased in terms of μ-strong convexity and L-smoothness.

Lemma 4.11 (Strongly convex optimality control)**.** *Let f be differentiable and μ-strongly convex on $\mathbb{R}^n$. Let $x^\star \in \mathbb{R}^n$ be the minimizer of f. Then for all $x \in \mathbb{R}^n$,*

$$\frac{\mu}{2}\|x - x^\star\|^2 \leq f(x) - f(x^\star) \leq \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

Proof. By convexity

$$f(x) \geq f(x^\star) + \nabla f(x^\star) \cdot (x - x^\star) + \frac{\mu}{2}\|x - x^\star\|^2.$$

But $\nabla f(x^\star) = 0$ by optimality (Theorem 2.72), so after rearranging we deduce the left inequality

$$\frac{\mu}{2}\|x - x^\star\|^2 \leq f(x) - f(x^\star).$$

Then it remains to show the right inequality. Again by convexity, we have

$$\begin{aligned} f(x^\star) &\geq f(x) + \nabla f(x) \cdot (x^\star - x) + \frac{\mu}{2}\|x - x^\star\|^2 \\ &= f_{x,\mu^{-1}}(x^\star) \\ &\geq \min_{z \in \mathbb{R}^n} \left\{ f_{x,\mu^{-1}}(z) \right\} \end{aligned}$$

where we recall the definition (4.5) of $f_{y,\eta}$. (Note the subtlety that although $x^\star$ is optimal for f, it is not necessarily optimal for the proxy $f_{x,\mu^{-1}}$.) Then Lemma 4.7 furnishes the optimal value of the minimization problem in the last expression, yielding the bound:

$$f(x^\star) \geq f(x) - \frac{1}{2\mu}\|\nabla f(x)\|^2.$$

After rearrangement, this establishes the right inequality, and the proof is complete. □

Lemma 4.12 (Smooth optimality control)**.** *Let f be convex and L-smooth on $\mathbb{R}^n$. Let $x^\star$ be a minimizer of f. Then for all $x \in \mathbb{R}^n$,*

$$\frac{1}{2L}\|\nabla f(x)\|^2 \leq f(x) - f(x^\star) \leq \frac{L}{2}\|x - x^\star\|^2.$$

Remark 4.13. Compare and contrast with Lemma 4.11. Together the two lemmas show that if f is L-smooth and μ-strongly convex, then we can use the 'residual' $\nabla f(x)$ of our iterates to construct both upper and lower bounds on the optimality gap and the distance to the optimizer.

Proof. As in the proof of Lemma 4.11, we apply L-smoothness (Theorem 3.3) two ways. First,

$$f(x) \leq f(x^\star) + \nabla f(x^\star) \cdot (x - x^\star) + \frac{L}{2}\|x - x^\star\|^2.$$

But $\nabla f(x^\star) = 0$ by optimality (Theorem 2.72), so after rearranging we deduce the right inequality

$$\frac{L}{2}\|x - x^\star\|^2 \geq f(x) - f(x^\star).$$

Then it remains to show the left inequality. By the descent lemma (Lemma 4.1), if we consider $y = x - \frac{1}{L}\nabla f(x)$, obtained by applying on iteration of gradient descent with step size $\frac{1}{L}$ to x, then we know that

$$f(y) \leq f(x) - \frac{1}{2L}\|\nabla f(x)\|^2.$$

But $f(x^\star) \leq f(y)$ by optimality, so after substitution and rearrangement we obtain the left inequality. □

4.5 Smooth and strongly convex case

We will now study the convergence of gradient descent in the case where f is L-smooth and μ-strongly convex.

Theorem 4.14 (Convergence of gradient descent: smooth and strongly convex case)**.** *Let f be L-smooth and μ-strongly convex on $\mathbb{R}^n$, and consider gradient descent (4.1) with step size $\eta = \frac{1}{L}$. Then for all $t = 0, 1, 2, \ldots$, we have*

$$f(x_t) - f(x^\star) \leq \left(1 - \frac{\mu}{L}\right)^t \left[f(x_0) - f(x^\star)\right].$$

Remark 4.15. This means that the optimality gap $f(x_t) - f(x^\star)$ decays exponentially as $t \to \infty$, where the base of the exponent is $1 - \frac{\mu}{L}$. Sometimes in optimization, we may informally define $\kappa := L/\mu \geq 1$ to be the **condition number** of the optimization problem, where we may understand L and μ to be the smallest and largest choices, respectively, for which f is both L-smooth and μ-strongly convex. The condition number can be viewed as controlling the difficulty of solving the optimization problem, in a sense that generalizes the way in which the condition number of a matrix controls the difficulty of solving a corresponding linear system via iterative methods. When the condition number is very large, this indicates the presence of different length scales that differ by orders of magnitude. We illustrate the connection below in Example 4.6, as well as in Figure 4.3.

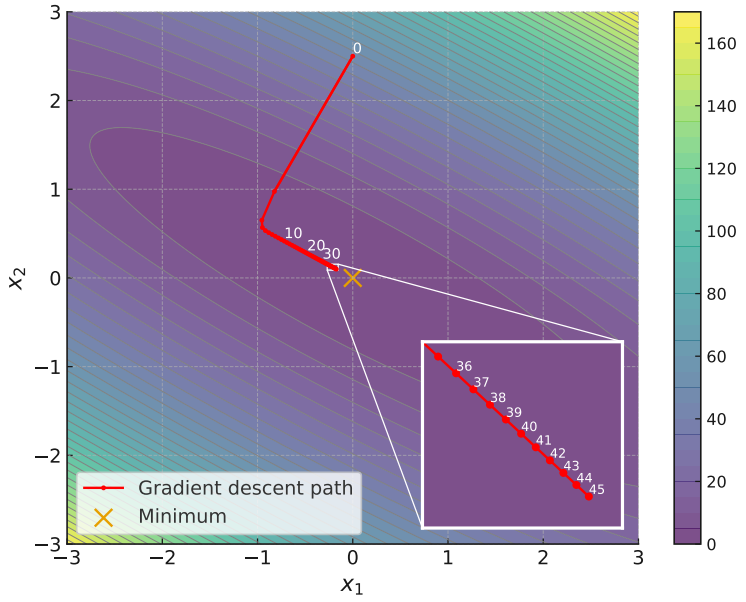

Figure 4.3: Illustration of gradient descent (4.1) on a poorly conditioned quadratic form. The numbers indicate the iteration count. To maintain stability in the sharp direction we must choose a step size $\eta \leq \frac{1}{L}$. We converge quickly in the sharp direction, but then in the flat direction, the step size is too small to make fast progress.

Remark 4.16. The decay factor in the theorem can be bounded using the general fact that $1+u \leq e^u$ for all $u \in \mathbb{R}$, which is in fact a direct consequence of the convexity of the exponential. Plugging in $u = -\kappa^{-1}$, this implies that

$$(1 - \kappa^{-1})^t \leq e^{-t/\kappa}.$$

This bound becomes asymptotically sharp when κ is large. Therefore to achieve an optimality gap within a target accuracy of $\varepsilon > 0$, it suffices to run gradient descent up to the T-th iterate where

$$T = \left\lceil \kappa \log\left(\frac{f(x_0) - f(x^\star)}{\varepsilon}\right) \right\rceil.$$

This means that to obtain any fixed accuracy $\varepsilon > 0$ in the objective, we must run gradient descent for a number of iterations that scales *linearly* with κ. Fortunately, the dependence on ε is only logarithmic due to the exponential convergence in the theorem.

Proof. Consider the descent lemma (Lemma 4.1) and subtract $f(x^\star)$ from both sides to deduce that

$$f(x_{t+1}) - f(x^\star) \leq [f(x_t) - f(x^\star)] - \frac{1}{2L}\|\nabla f(x_t)\|^2,$$

where we have substituted $\eta = \frac{1}{L}$. Then use Lemma 4.11 to lower-bound the gradient appearing on the right-hand side as

$$\|\nabla f(x_t)\|^2 \geq 2\mu\,[f(x_t) - f(x^\star)]\,.$$

Substituting and simplifying , we deduce

$$f(x_{t+1}) - f(x^\star) \le \left(1 - \frac{\mu}{L}\right) [f(x_t) - f(x^\star)] .$$

Since this holds for all $t = 0, 1, 2, \ldots$, the theorem follows by using this inequality inductively. □

Example 4.17 (Gradient descent on a positive definite quadratic form)**.** Recall the model quadratic objective studied in Example 4.6

$$f(x) = \frac{1}{2} x^\top A x - b \cdot x,$$

where A is positive definite. Let $\lambda_{\max}$ and $\lambda_{\min} > 0$ denote the largest and smallest eigenvalues, respectively. Thus in particular we know that (1) $\|A\|_2 = \lambda_{\max}$ and (2) $A - \lambda_{\min} I$ is PSD. Therefore Theorem 3.11 and Theorem 2.30 guarantee, respectively, that f is $\lambda_{\max}$-smooth and $\lambda_{\min}$-strongly concave. Then the condition number $\kappa = \lambda_{\max}/\lambda_{\min}$ in the optimization sense of Remark 4.15 is the same as the condition number of A as a matrix in the usual sense of numerical linear algebra (ratio of largest to smallest eigenvalue).

In Figure 4.3 we illustrate how poor conditioning slows down the progress of gradient descent.

4.6 Gradient descent as a contraction map

There is another perspective on gradient descent that allows us to more directly analyze the distance from the optimizer $\|x_t - x^\star\|$. The idea is to consider the map $\mathbb{R}^n \to \mathbb{R}^n$ that implements that the gradient descent update and show that it is a *contraction* in the following sense.

Definition 4.18 (Contraction maps)**.** For $\alpha \in [0, 1)$, the function $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is called an *α-contraction map* such if for all $x, y \in \mathbb{R}^n$, $\|\phi(x) - \phi(y)\| \le \alpha \|x - y\|$. We say that ϕ is *nonexpansive* if it merely holds that $\|\phi(x) - \phi(y)\| \le \|x - y\|$ for all $x, y \in \mathbb{R}^n$.

In the context of gradient descent with fixed step size η, our iteration map is defined by

$$\phi(x) := x - \eta \nabla f(x). \tag{4.6}$$

Note that x is a **fixed point** of ϕ, i.e., a point such that $\phi(x) = x$, if and only if $\nabla f(x) = 0$, which is equivalent to optimality for a smooth unconstrained convex objective.

Then we can view gradient descent as a **fixed-point iteration**, i.e., a sequence defined by

$$x_{t+1} = \phi(x_t), \tag{4.7}$$

which we hope converges to a fixed point of ϕ.

There is a classical theorem on the convergence of fixed point iterations, which holds over very general spaces. We state it only in the setting where we need it.

Theorem 4.19 (Banach fixed-point theorem)**.** *Suppose that $\phi : \mathbb{R}^n \to \mathbb{R}^n$ is an α-contraction map for $\alpha \in [0, 1)$. Then there exists a unique fixed point $x^\star$ of ϕ. Moreover, for arbitrary $x_0 \in \mathbb{R}^n$, define a sequence by the fixed-point iteration (4.7). Then for all $t = 0, 1, 2, \ldots$, it holds that*

$$\|x_t - x^\star\| \le \alpha^t \|x_0 - x^\star\|.$$

Remark 4.20. The inequality still holds with $\alpha = 1$ even if ϕ is merely nonexpansive, though the existence of a unique fixed point cannot be guaranteed in general in this case.

Proof. We will not prove the existence of a unique fixed point $x^\star$ since in our application setting of interest, i.e., unconstrained optimization of smooth and strongly convex functions, we already know that there exists a unique fixed point for (4.6). (The construction of $x^\star$ is also a bit of a detour; it involves showing that the fixed-point iteration sequence is Cauchy. However, showing uniqueness is a good exercise to check your understanding.)

Let us prove only the last statement. Observe that for any t,

$$\|x_{t+1} - x^\star\| = \|\phi(x_t) - \phi(x^\star)\| \leq \alpha \|x_t - x^\star\|,$$

where in the first equality we have used the definition (4.7) of x_{t+1} and the fact that $x^\star$ is a fixed point. The inequality uses the contraction map property. Since this holds for all $t = 0, 1, 2, \ldots$, the desired statement follows by using this inequality inductively. □

Next we establish that gradient descent defines a contraction map via (4.6) as long as the step size is sufficiently small.

Lemma 4.21 (Gradient descent as a contraction map)**.** *Let f be L-smooth and μ-strongly convex on $\mathbb{R}^n$, and consider gradient descent (4.1) with step size $\eta = \frac{1}{L}$. If $\mu > 0$, then the iteration map (4.6) is an α-contraction map for $\alpha = 1 - \frac{\mu}{L}$. If $\mu = 0$, i.e., f is merely assumed to be convex, the iteration map is nonexpansive.*

Remark 4.22. In the proof we will assume that f is twice differentiable everywhere. However, this is not a necessary assumption, so we omit it from the statement. One way to avoid the assumption is via Alexandrov's theorem (cf. Remark 2.29). It can also be avoided by considering a mollifying sequence approximation of f, since standard mollification preserves L-smoothness and μ-strong convexity. As far as I can tell, there is no direct alternative approach yielding the same sharp result. Please ignore these points if you like.

Proof. Let $x, y \in \mathbb{R}^n$ be arbitrary. We want to show that

$$\|\phi(y) - \phi(x)\| \leq \left(1 - \frac{\mu}{L}\right) \|y - x\|, \tag{4.8}$$

which will complete the proof.

First define $x_t = x + t(y - x)$ for $t \in [0, 1]$, and use the fundamental theorem of calculus to write

$$\phi(y) = \phi(x) + \int_0^1 \frac{d}{dt} \left[\phi(x_t)\right] \, dt. \tag{4.9}$$

Now by (4.6), we compute using standard manipulations and the chain rule

$$\begin{aligned}\frac{d}{dt}\left[\phi(x_t)\right] &= \frac{d}{dt}\left[x_t - \eta\nabla f(x_t)\right] \\ &= \frac{d}{dt}\left[x_t\right] - \eta\frac{d}{dt}\left[\nabla f(x_t)\right] \\ &= (y-x) - \eta\nabla^2 f(x_t)\,\frac{d}{dt}\left[x_t\right] \\ &= (y-x) - \eta\nabla^2 f(x_t)\,(y-x) \\ &= \left(I - \eta\nabla^2 f(x_t)\right)(y-x).\end{aligned}$$

Then we substitute into (4.9) and rearrange to obtain

$$\begin{aligned}\|\phi(y) - \phi(x)\| &= \left\|\int_0^1 \left(I - \eta\nabla^2 f(x_t)\right)(y-x)\,dt\right\| \\ &\le \int_0^1 \left\|\left(I - \eta\nabla^2 f(x_t)\right)(y-x)\right\|\,dt \\ &\le \int_0^1 \|I - \eta\nabla^2 f(x_t)\|\,\|y-x\|\,dt. \qquad (4.10)\end{aligned}$$

Now since f is μ-strongly convex and L-smooth, we know that the eigenvalues of $\nabla^2 f$ always lie in $[\mu, L]$, cf. Remark 3.12. Thus, recalling that $\eta = \frac{1}{L}$, the eigenvalues of $I - \eta\nabla^2 f(x_t)$ lie in $[0, 1-\mu/L]$. Then by Exercise 3.9,

$$\|I - \eta\nabla^2 f(x_t)\| \le 1 - \frac{\mu}{L},$$

and in turn it follows from (4.10) that (4.8) holds, as we desired to show. □

Corollary 4.23 (Iterate convergence of gradient descent: smooth and strongly convex case)**.** *Let f be L-smooth and μ-strongly convex on $\mathbb{R}^n$, and consider gradient descent (4.1) with step size $\eta = \frac{1}{L}$. Then for all $t = 0, 1, 2, \ldots$, it holds that*

$$\|x_t - x^\star\| \le \left(1 - \frac{\mu}{L}\right)^t \|x_0 - x^\star\|.$$

Remark 4.24. Similar comments as those made in Remark 4.16 apply here, but note that the relevant notion of error here is different (distance to optimizer vs. optimality gap).

Remark 4.25. Note that in the case $\mu = 0$, where f is merely convex, the result holds but does not imply convergence. However, we are still guaranteed that $\|x_t - x^\star\| \le \|x_0 - x^\star\|$ for all t, i.e., gradient descent with a sufficiently small step size cannot increase the initial distance from the optimizer. This point will be useful later on when we drop the strong convexity assumption.

Proof. The result follows directly from the Banach fixed point theorem together with Lemma 4.21. □

4.7 Smooth and convex case

Now we must leave behind the comfort of strong convexity! We will still be able to prove a convergence result for gradient descent on L-smooth objectives, but it has a completely different qualitative character.

Theorem 4.26 (Convergence of gradient descent: smooth convex case)**.** *Let f be convex and L-smooth on $\mathbb{R}^n$ and suppose that $x^\star$ is any minimizer. Consider gradient descent (4.1) with step size $\eta = \frac{1}{L}$. Then for all $t = 1, 2, \ldots$, we have*

$$f(x_t) - f(x^\star) \leq \frac{L\|x_0 - x^\star\|^2}{2t}.$$

Remark 4.27. Note the qualitative difference in the convergence rate of the optimality gap compared to the smooth and strongly convex case of Theorem 4.14. Here the rate is algebraic $O(L/t)$, as opposed to the exponential bound of $O(e^{-t/\kappa})$, where $\kappa = L/\mu$, appearing in Theorem 4.14.

Importantly, our analysis here in Theorem 4.26 *still applies* to μ-strongly convex f where $\mu > 0$. To reiterate, we should not think of it as only applying in the case $\mu = 0$. We should also think of it as an alternative bound which can be sharper than the bound of Theorem 4.14 in the case where μ is extremely small, hence the condition number κ is extremely large.

Remark 4.28. Let us examine this point more quantitatively in contrast with the discussion of Remark 4.16. Theorem 4.26 implies that to achieve an optimality gap within a target accuracy of $\varepsilon > 0$, it suffices to run gradient descent up to the T-th iterate where

$$T = \left\lceil \frac{L\|x_0 - x^\star\|^2}{2\varepsilon} \right\rceil.$$

This means that to obtain any fixed accuracy $\varepsilon > 0$ in the objective, we must run gradient descent for a number of iterations that scales *linearly* with L. This may be much better than the linear dependence on κ revealed in Remark 4.16. However, the catch with the new result is that the dependence on ε is now $O(\varepsilon^{-1})$ as opposed to $O(\log(1/\varepsilon))$, so getting a highly accurate estimate (i.e., within machine precision) of $f(x^\star)$ is essentially impossible.

Proof. This proof is pretty inspired and involves a trick that is not very intuitive *a priori*.

First we know from the convexity of f (via Theorem 2.20) that

$$f(x^\star) \geq f(x_t) + \nabla f(x) \cdot (x^\star - x_t).$$

But from L-smoothness we know (via Theorem 3.3) that

$$f(x_{t+1}) \leq f(x_t) + \nabla f(x_t) \cdot (x_{t+1} - x_t) + \frac{L}{2}\|x_{t+1} - x_t\|^2.$$

Subtract the first inequality from the second to deduce

$$f(x_{t+1}) - f(x^\star) \leq \nabla f(x_t) \cdot (x_{t+1} - x^\star) + \frac{L}{2}\|x_{t+1} - x_t\|^2.$$

Now we can solve for $\nabla f(x_t)$ in the gradient descent update rule (4.1) with step size $\eta = \frac{1}{L}$ to write

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

and substitute to obtain

$$f(x_{t+1}) - f(x^\star) \leq L\,(x_t - x_{t+1}) \cdot (x_{t+1} - x^\star) + \frac{L}{2}\|x_{t+1} - x_t\|^2. \tag{4.11}$$

Next, we will expand the dot product in the right-hand side using the polarization identity

$$u \cdot v = \frac{1}{2}\left(\|u+v\|^2 - \|u\|^2 - \|v\|^2\right)$$

which holds for general $u, v \in \mathbb{R}^n$. (This is equivalent to the expansion $\|u+v\|^2 = \|u\|^2 + \|v\|^2 + 2\,u \cdot v$.)

Plugging in $u = x_t - x_{t+1}$ and $v = x_{t+1} - x^\star$ and substituting into (4.11), we obtain

$$\begin{aligned} f(x_{t+1}) - f(x^\star) &\leq \frac{L}{2}\left(\|x_t - x^\star\|^2 - \|x_t - x_{t+1}\|^2 - \|x_{t+1} - x^\star\|^2\right) + \frac{L}{2}\|x_{t+1} - x_t\|^2 \\ &= \frac{L}{2}\|x_t - x^\star\|^2 - \frac{L}{2}\|x_{t+1} - x^\star\|^2, \end{aligned}$$

due to a cancellation of terms in the last step.

Now for any positive integer T, we can sum these inequalities over $t = 0, 1, \ldots, T-1$, taking advantage of telescoping to deduce

$$\begin{aligned} \sum_{t=0}^{T-1}[f(x_{t+1}) - f(x^\star)] &\leq \frac{L}{2}\|x_0 - x^\star\|^2 - \frac{L}{2}\|x_T - x^\star\|^2 \\ &\leq \frac{L}{2}\|x_0 - x^\star\|^2. \end{aligned}$$

But by the descent lemma, given our choice of step size $\eta = \frac{1}{L}$, we know that $f(x_{t+1}) \geq f(x_T)$ for all $t = 0, \ldots, T-1$. After plugging in and dividing by T, we deduce that

$$f(x_T) - f(x^\star) \leq \frac{L}{2T}\|x_0 - x^\star\|^2,$$

and since T was arbitrary this completes the proof. □

4.8 Backtracking line search

In practice we may not be given any upper bound for the strong smoothness parameter L, which is what we need to ensure that $\eta \leq \frac{1}{L}$. (Recall that we can just assume $\eta = \frac{1}{L}$ by relaxing this upper bound if necesary.)

To address this shortcoming, we can consider running gradient descent (4.2) where the step size η_t for each iteration is chosen via a search procedure.

Specifically, inspired by the descent lemma (Lemma 4.1) we want to choose η_t such that

$$f(x_{t+1}) \leq f(x_t) - \frac{\eta_t}{2}\|\nabla f(x_t)\|^2. \tag{4.12}$$

The descent lemma says that this can be guaranteed by taking η_t sufficiently small (i.e., $\eta_t \leq \frac{1}{L}$). However, it is really the descent property itself that powered our convergence theorems. As long as (4.12) holds at every iteration, Theorems 4.14 and 4.26 hold with $\sup_{t=0,1,2,\ldots}\{1/\eta_t\}$ in place of L.

Therefore we want an algorithm for choosing η_t ensuring that it is not too much smaller than necessary. Specifically, we want a choice guaranteeing $\eta_t \geq \tau/L$, where $\tau \in (0,1)$ is a fixed hyperparameter.

This can be achieved by **backtracking line search**, in which we try an initial guess $\eta_t \leftarrow \eta_t^{(0)}$ and keep reducing η_t by a factor of τ until (4.12) holds, then output η_t. Once η_t drops below $1/L$, we know that (4.12) will hold. Therefore the output step size satisfies $\eta_t \geq \min\{\eta_t^{(0)}, \tau/L\}$.

An aggressive initial guess (i.e., $\eta_t^{(0)} \gg 1/L$) will ensure that $\eta_t \geq \tau/L$. Note that the downside of an aggressive choice is that we must perform many evaluations of the objective for each line search. Specifically, if $\eta_t^{(0)} \leq 1/L$, we at worst we must perform $\lceil 1 + \log_{1/\tau}(\eta_t^{(0)} L) \rceil$ evaluations.

We assume that we always set $\eta_{t+1}^{(0)} \geq \eta_t$, i.e., choose each guess for the step size to be at least as aggressive as the last step size. Under this assumption we are guaranteed that $\eta_{t+1} \geq \min\{\eta_t, \tau/L\}$ for all t. In turn this ensures that $\eta_t \geq \min\{\eta_0^{(0)}, \tau/L\}$ for all t.

A reasonable heuristic is to take

$$\eta_{t+1}^{(0)} = \gamma\, \eta_t,$$

where $\gamma > 1$ is another hyperparameter. This way, we are constantly trying to nudge up the step size to be more aggressive, but the backtracking line search criterion keeps us in check.

We summarize our conclusions in the following remark.

Remark 4.29 (Convergence of gradient descent with backtracking line search). Performing gradient descent (4.2) in which the step sizes are chosen via backtracking line search as outlined above, Theorems 4.14 and 4.26 hold with $\tilde{L} := \max\{L/\tau, 1/\eta_0^{(0)}\}$ in the place of L. If we make sure that our initial guess is sufficiently aggressive, i.e., satisfying $\eta_0^{(0)} \leq 1/L$, then we have simply $\tilde{L} = L/\tau$.

Algorithm 2 `BacktrackGD`$(f, x, \eta^{(0)}, \tau)$, backtracking line search gradient descent step

Input: $f : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$, step size guess $\eta^0 > 0$, contraction parameter $\tau \in (0,1)$

Output: $\eta > 0$ and new point $x^+ = x - \eta \nabla f(x)$ satisfying $f(x^+) \leq f(x) - \frac{\eta}{2}\|\nabla f(x)\|^2$

```
Set η ← η_0, p ← f(x), v ← ∇f(x)
while f(x − ηv) > p − (η/2)‖v‖² do
    Set η ← τη
end while
return η, x − ηv
```

The pseudocode for a single step of gradient descent with backtracking line search is given in Algorithm 2. In terms of this algorithm, the update rule can be written

$$\eta_t, x_{t+1} := \texttt{BacktrackGD}(f, x_t, \eta_t^{(0)}, \tau), \quad \eta_{t+1}^{(0)} := \gamma\, \eta_t$$

where $\eta_t^{(0)}$ is the step size guess at the t-th iteration.

Reasonable choices for the hyperparameters are given by $\tau = 0.5$ and $\gamma = 1.2 \in (1, \tau^{-1})$, but the best choices may be problem-dependent. Moreover, if evaluation of ∇f is significantly more expensive than evaluation of f, it makes sense to do a more careful line search, i.e., choose τ closer to 1.

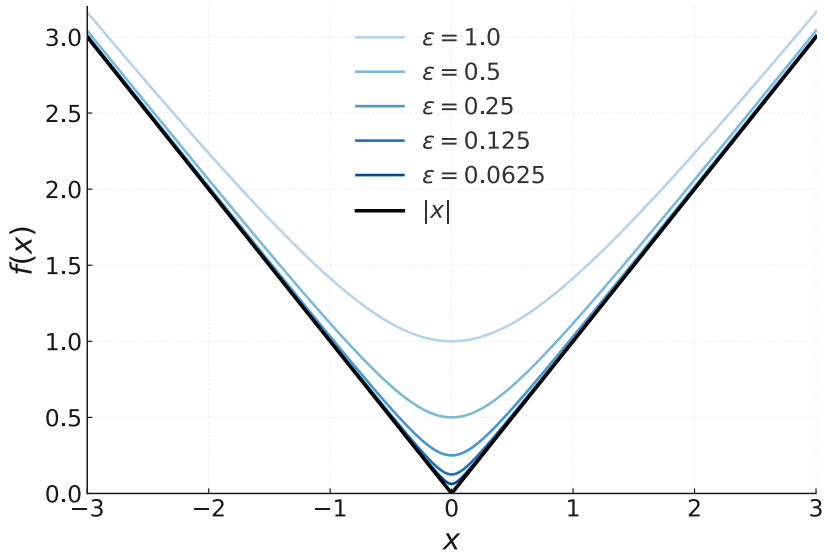

Figure 5.1: Illustration of sequence of smooth functions $f_\varepsilon(x) = \sqrt{x^2 + \varepsilon^2}$ approaching the non-smooth limiting function $f(x) = |x|$ as $\varepsilon \to 0$.

5 Subgradient descent

Finally, it is time to leave behind our last major luxury: L-smoothness. This means that we have to give up the descent lemma, and we cannot guarantee a decrease of the objective with a fixed step size. That sounds quite dismal, but there is a loophole that gives us hope: decaying step size. Before proceeding, first let us put our goals into context.

5.1 Motivation

Consider Figure 5.1, illustrating a sequence of smooth functions $f_\varepsilon(x) = \sqrt{x^2 + \varepsilon^2}$ approaching the non-smooth limiting function $f(x) = |x|$ as $\varepsilon \to 0$. In this section, we care about optimizing non-smooth functions like $f(x) = |x|$, but our analysis is also relevant for L-smooth functions whose gradients are bounded but with poor quantitative smoothness (i.e., functions for which L is extremely large).

Note that the sequence in Figure 5.1 converges to a limiting convex function which is not even differentiable. In order to come up with a theory that is robust in the non-smooth limit, we must define a generalization of the gradient called the *subgradient*.

5.2 Subgradients

Definition 5.1 (Subdifferentials and subgradients)**.** Let $f : C \to \mathbb{R}$ be convex on a convex domain $C \subset \mathbb{R}^n$. For any $x_0 \in C$, define the *subdifferential* (of f at x_0) as

$$\partial f(x_0) = \{v \in \mathbb{R}^n \,:\, f(x_0) + v \cdot (x - x_0) \leq f(x) \;\text{ for all } x \in \mathbb{R}^n\}.$$

We say that any $v \in \partial f(x_0)$ is a *subgradient* (of f at x_0).

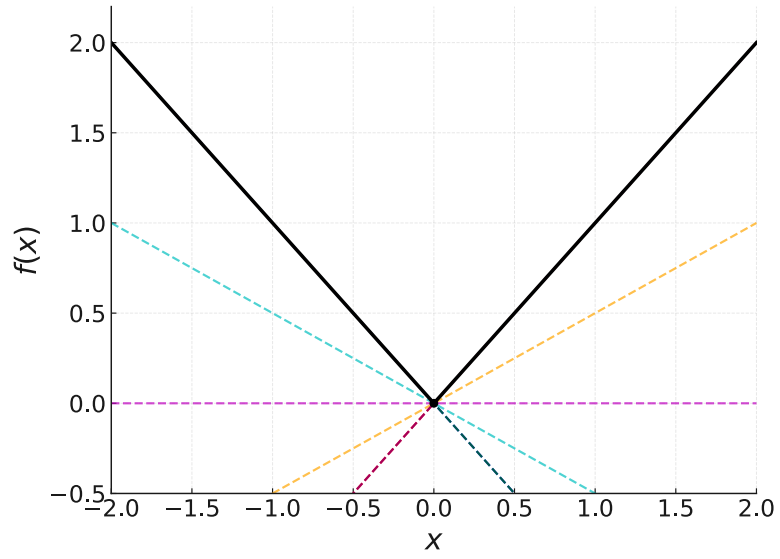

Figure 5.2: The slopes of the dotted lines all lie in the subdifferential $\partial f(0)$ of the function $f(x) = |x|$ at $x = 0$. The graph of f is plotted with a solid line.

The subdifferential at a point x consists of all vectors defining an affine-linear function touching the graph from below at x. In light of Theorem 2.20, we know that if f is differentiable at x, then $\nabla f(x) \in \partial f(x)$.

Example 5.2 (Subdifferential of $|x|$)**.** For $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = |x|$, we have that

$$\partial f(x) = \begin{cases} \{1\} & \text{if } x > 0, \\ \{-1\} & \text{if } x < 0, \\ [-1, 1] & \text{if } x = 0. \end{cases}$$

Exercise 5.3 (Subdifferentials are convex sets)**.** The subdifferential (of any convex function at any point in its domain) is a convex set.

The following theorem guarantees existence of subgradients in typical scenarios.

Theorem 5.4 (Subdifferentials are nonempty in the interior)**.** *Let $f : C \to \mathbb{R}$ be convex on a convex domain $C \subset \mathbb{R}^n$. For any x in the interior of C, $\partial f(x)$ is nonempty.*

We will defer the proof for the moment. First it is worthwhile to illustrate that at boundary points, even where f is defined and finite, subgradients may fail to exist.

Exercise 5.5. Find some $f : C \to \mathbb{R}$, convex on a convex domain $C \subset \mathbb{R}^n$, for which there exists $x \in C$ such that $\partial f(x)$ is empty. **Hint:** The preceding theorem says that the point must be a boundary point. Take $n = 1$, $C = [0, 1]$, and try to build a convex function that doesn't admit a subgradient at 0.

Next we illustrate an important and reassuring fact. The subdifferential consists only of the gradient wherever the function is differentiable.

Theorem 5.6 (Subdifferential is a singleton at differentiable points)**.** *Let $f : C \to \mathbb{R}$ be convex on a convex domain $C \subset \mathbb{R}^n$. If f is differentiable at some x in the interior of C, then $\partial f(x) = \{\nabla f(x)\}$.*

Proof. Suppose f is differentiable at x_0 in the interior of C. We automatically know $\nabla f(x_0) \in \partial f(x_0)$ by Theorem 2.20. But we need to rule out the possibility that there are *other* subgradients.

Suppose that $v \in \partial f(x_0)$. We want to show that $v = \nabla f(x_0)$. Now by the subgradient property,

$$f(x) \geq f(x_0) + v \cdot (x - x_0)$$

for all $x \in C$. But by differentiability, there exists ψ defined near the origin such that $\lim_{z \to 0} \psi(z) = 0$ and

$$f(x) = f(x_0) + \nabla f(x_0) \cdot (x - x_0) + \psi(x - x_0)\, \|x - x_0\|.$$

Then subtracting our equation from the subgradient inequality and rearranging, we find that

$$[v - \nabla f(x_0)] \cdot \frac{x - x_0}{\|x - x_0\|} \leq \psi(x - x_0)$$

for all x near x_0.

For any $z \in \mathbb{R}^n$ with $\|z\| = 1$, we can plug in $x = x_0 + tz$, where $t \neq 0$ is sufficiently small, to deduce that

$$[v - \nabla f(x_0)] \cdot z = \psi(tz).$$

Then taking the limit as $t \to 0$, we deduce that

$$[v - \nabla f(x_0)] \cdot z \leq 0$$

for all unit vectors z. Then plugging in

$$z = \frac{v - \nabla f(x_0)}{\|v - \nabla f(x_0)\|},$$

we deduce that $\|v - \nabla f(x_0)\| \leq 0$, which in turn implies that $\|v - \nabla f(x_0)\| = 0$, i.e., $v = \nabla f(x_0)$. This completes the proof. □

Now to prove Theorem 5.4, we need to make another definition.

Definition 5.7 (Epigraphs)**.** Let $f : C \to \mathbb{R}$ where $C \subset \mathbb{R}^n$. The *epigraph* of f, denoted $\text{epi}(f)$ is a subset of $\mathbb{R}^{n+1}$ defined as follows

$$\text{epi}(f) = \{(x, t) \,:\, x \in C,\, t \in \mathbb{R},\, t \geq f(x)\}.$$

The epigraph defines a connection between convex functions and convex sets.

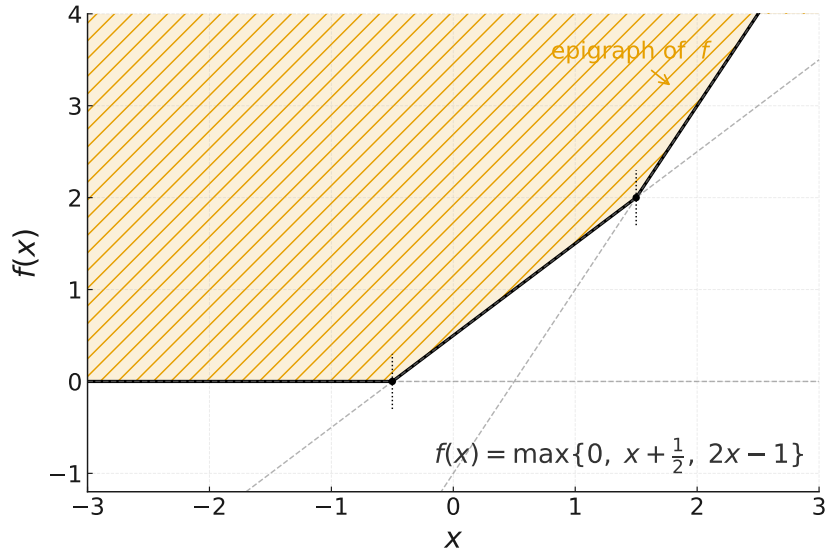

Figure 5.3: Depiction of the epigraph of a piecewise linear convex function, $f(x) = \max\left\{0, x + \frac{1}{2}, 2x - 1\right\}$.

Exercise 5.8 (Epigraphs of convex functions are convex sets)**.** Consider $f : C \to \mathbb{R}$ where $C \subset \mathbb{R}^n$ is a convex domain. Then $\text{epi}(f)$ is a convex set if and only if f is a convex function. **Hint:** To get started, first consider the reverse direction, assuming that f is convex. Then let (x, t) and (y, s) lie in $\text{epi}(f)$ and let $\theta \in [0, 1]$. What we want to show is that $([1-\theta]x + \theta y, [1-\theta]t, \theta s) \in \text{epi}(f)$. From there, try to unpack things further using the definition of the epigraph.

Finally, we proceed with the proof of Theorem 5.4.

Proof of Theorem 5.4. Let x_0 be in the interior of C. We want to show that $\partial f(x_0)$ is nonempty. We do this by constructing a subgradient.

Now the point $(x_0, f(x_0))$ lies in $\partial\text{epi}(f)$, the boundary of the epigraph of f (namely the 'graph'). Since $\text{epi}(f)$ is convex by Exercise 5.8, the supporting hyperplane theorem (Theorem 2.8) says that $\text{epi}(f)$ admits a supporting hyperplane at $(x_0, f(x_0))$. Concretely, there exists $(b, c) \in \mathbb{R}^{n+1}$ nonzero (in which $b \in \mathbb{R}^n$ and $c \in \mathbb{R}$) such that

$$(b, c) \cdot (x, t) \leq (b, c) \cdot (x_0, f(x_0)) \quad \text{for all } (x, t) \in \text{epi}(f). \tag{5.1}$$

To visualize the supporting hyperplane it should help to look at Figure 5.3. The vector (b, c) defines the direction normal to the hyperplane, pointing downward away from the epigraph.

We will use the fact that x_0 lies in the interior of C to rule out the possibility that $c = 0$, i.e., that the hyperplane is 'vertical.' Indeed, suppose for contradiction that $c = 0$. Then it follows from (5.1) that $b \cdot x \leq b \cdot x_0$ for all $x \in C$, or $b \cdot (x - x_0) \leq 0$ for all $x \in C$. But since x_0 is an interior point, we know that $x = x_0 + \varepsilon b \in C$ for $\varepsilon > 0$ sufficiently small. Then by plugging in this choice of x, we deduce that $\varepsilon \|b\|^2 \leq 0$, which in turn implies that $b = 0$. This is a contradiction, because the normal vector (b, c) must be nonzero.

Next we establish that in fact $c < 0$, which corresponds to the fact that (b, c) points away from the epigraph. Observe that $(x_0, t) \in \text{epi}(f)$ as long as $t \geq f(x_0)$. Hence substituting such (x_0, t)

for (x,t) in (5.1) implies, via some cancellation, that $c(f(x_0)-t)\geq 0$. Plugging in any $t<f(x_0)$, we see that we must have $c\leq 0$ for the inequality to hold. Since we already know that $c\neq 0$, we conclude that $c<0$ as claimed.

Then we can divide (5.1) by $-c>0$ and define $v:=-b/c$ to conclude that

$$(v,-1)\cdot(x,t)\leq(v,-1)\cdot(x_0,f(x_0)) \quad \text{for all } (x,t)\in \operatorname{epi}(f).$$

Equivalently, after expanding the dot products and rearranging:

$$f(x_0)+v\cdot(x-x_0)\leq t \quad \text{for all } (x,t)\in \operatorname{epi}(f).$$

In particular $(x,f(x))$ lies in $\operatorname{epi}(f)$ for all $x\in C$, so it follows that

$$f(x_0)+v\cdot(x-x_0)\leq t \quad \text{for all } x\in C,$$

meaning precisely that v is a subgradient. □

The following exercise shows how to furnish a subgradient at any point of a function defined as pointwise maxima of several 'ingredient' functions. This is a common way that non-smoothness arises in practice. The non-differentiable points are those where the graphs of the ingredient functions cross, cf. Figure 5.3.

Exercise 5.9 (Subdifferential of pointwise maximum)**.** Suppose $f_1,f_2:C\to\mathbb{R}$ are convex functions on a convex domain $C\subset\mathbb{R}^n$, and define $f:C\to\mathbb{R}$ as the pointwise maximum $f(x)=\max\{f_1(x),f_2(x)\}$ for $x\in C$. Show that for any $x\in C$, if $f_1(x)\geq f_2(x)$ and $v\in\partial f_1(x)$, then $v\in\partial f(x)$. Likewise show that if $f_2(x)\geq f_1(x)$ and $v\in\partial f_2(x)$, then $v\in\partial f(x)$.

In other words, at any point x, either f_1, f_2, or both 'activated' in the sense that it is the greatest. A subgradient of any activated function at x is also a subgradient of the pointwise maximum.

Remark 5.10. More generally, suppose that $f_i:C\to\mathbb{R}$, $i=1,\dots,k$, are convex functions on a convex domain $C\subset\mathbb{R}^n$, and define $f:C\to\mathbb{R}$ by $f(x)=\max_{i=1,\dots,k}\{f_i(x)\}$ for $x\in C$. For any x, suppose that i is a maximizing index in the expression for $f(x)$, i.e., that $f(x)=f_i(x)$. Then $\partial f_i(x)\subset\partial f(x)$. The proof of this fact is just a slight generalization of the exercise.

Remark 5.11. In fact, it is possible to say more and characterize the subdifferential $\partial f(x)$ completely as the *convex hull* (denoted here 'conv' and cf. Remark 2.4) of the union of the subdifferential of all functions that are 'activated' at x

$$\partial f(x)=\operatorname{conv}\left(\bigcup_{i:f(x)=f_i(x)}\partial f_i(x)\right).$$

In the case of $f(x)=|x|=\max\{x,-x\}$, this recovers the observation that

$$\partial f(0)=\operatorname{conv}(\{-1,1\})=[-1,1],$$

cf. Example 5.2. The preceding remark implies the '$\supset$' containment, but the reverse containment is harder to prove.

Subgradients also transform intuitively under suitable operations preserving convexity.

Remark 5.12. Suppose $f, g : C \to \mathbb{R}$ are convex and $\lambda \geq 0$, and define $h := \lambda f + g$. Suppose that g is differentiable at x in the interior of C. Then

$$\partial h(x) = \lambda \, \partial f(x) + \nabla g(x) = \{\lambda v + \nabla g(x) \, : \, v \in \partial f(x)\}.$$

We will omit the proof.

Finally, we offer a generalization of Theorem 2.72 (which says that if the optimal value is attained at a differentiable interior point, then it is a critical point) to the nondifferentiable case.

Theorem 5.13 (Generalized first-order optimality)**.** *Let $f : C \to \mathbb{R}$ be convex on a convex domain $C \subset \mathbb{R}^n$ and suppose that $x^\star$ is an optimizer of f over C. Then $0 \in \partial f(x^\star)$.*

Proof. For any $x \in C$, we have

$$f(x) \geq f(x^\star) = f(x^\star) + 0 \cdot (x - x^\star),$$

which means precisely that $0 \in \partial f(x^\star)$. □

5.3 Subgradient descent

Subgradient descent is an unconstrained optimization algorithm that generalizes gradient descent (4.2) to the case of non-differentiable f. Given an initial guess x_0, it defines:

$$x_{t+1} = x_t - \eta_t g_t, \quad \text{where } v_t \in \partial f(x_t), \quad t = 0, 1, 2, \ldots \tag{5.2}$$

Here η_t is a step size which may depend on the iteration t. Note that subgradient descent only requires us to furnish *some* subderivative v_t at each iterate x_t. Typically this is not much more of a challenge than finding an ordinary gradient, cf. Exercise 5.9. Let us expand further upon our philosophy in considering this algorithm.

Remark 5.14 (Philosophy of subgradient descent analysis). The point of considering subgradient descent is not necessarily that we will really have to compute subgradients at nondifferentiable points. In fact, convex functions are differentiable almost everywhere (due to Rademacher's theorem, which is outside the scope of the course), and generically subgradient descent will never see a non-differentiable point. (Caveat: if we determine η_t using an exact line search we may often see non-differentiable points.)

Moreover, as we noted in the motivating discussion of Section 5.1, we may even be interested in applying the analysis of subgradient descent to smooth objectives, where subgradient descent is just gradient descent! Specifically, it will be useful for analyzing objectives with poor quantitative smoothness (large L) that nonetheless admit bounds on the gradients and subgradients. Think of $f(x) = |x|$ as well as its smooth approximations depicted in Figure 5.1.

Philosophically, what is important here is that the subgradient property, which is in particular satisfied by a true gradient, is the crucial ingredient in the analysis. This means that the analysis extends robustly to the possibly non-differentiable case, where we can make statements of greater elegance and generality.

To get started, we illustrate the necessity of considering a decaying step size in the analysis of gradient descent.

Exercise 5.15. Consider the objective $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = |x|$. Show that for any fixed step size $\eta_t = \eta > 0$, it is possible to choose initial conditions such that subgradient descent (4.1) does not converge as $t \to \infty$. **Hint:** You should be able to see that for a generic initial condition, the algorithm eventually gets stuck in a loop that bounces back and forth across the origin.

Exercise 5.16. Consider the objective $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = |x|$. Show that if we choose the step size schedule $\eta_t = 1/\sqrt{t+1}$, subgradient descent (4.1) converges as $t \to \infty$, regardless of the initial condition. **Hint:** You should first show that the algorithm crosses the origin infinitely many times. (When we say 'crosses', we include the possibility that it lands on the origin.) If you cross the origin at step t, in light of the decaying step size, what can you say about $|x_s|$ for all future iterations $s \geq t$?

5.4 Convergence of subgradient descent: finite time horizon

In light of Exercise 5.15, we cannot expect a convergence result in the same style as Theorem 4.26 for gradient descent with a fixed step size. However, in light of Exercise 5.16, there seems to be hope if we allow a decaying step size.

We will ultimately prove such a result, but first we will consider a result that is easier to prove, though the logic of the statement is a bit funky.

Suppose we fix a time horizon T i.e., fix the number of iterations of gradient descent in advance. Can we choose our step size $\eta = \eta(T)$ as a function of T, so that the optimality gap of this algorithm after T iterations is guaranteed to be small? We will see that a choice of the form $\eta = O(1/\sqrt{T})$ emerges naturally from this question, and it guarantees an optimality gap of $O(1/\sqrt{T})$ at the end of the procedure.

This suggests that if we are willing to run gradient descent for a long time T, we should choose a smaller step size $\eta = \eta(T)$ to get a better final value. But Exercise 5.15 suggests that if we continue running the algorithm after the time horizon, we cannot expect any further improvement. Spiritually, this feels quite similar to saying that we can get convergence with a decaying step size. Indeed it is, but we will come to that later.

Definition 5.17 (Bounded subgradients)**.** We say that a convex function $f : C \to \mathbb{R}$ on a convex domain $C \subset \mathbb{R}^n$ has G-bounded subgradients if for any $x \in C$ and any $v \in \partial f(x)$, we have $\|g\| \leq G$.

Theorem 5.18 (Convergence of subgradient descent: fixed step size, finite time horizon)**.** *Let f be convex on $\mathbb{R}^n$ with G-bounded subgradients. Consider subgradient descent (5.2) with fixed step size $\eta_t = \eta > 0$. For any positive integer T and any $x \in \mathbb{R}^n$, we have that*

$$\frac{1}{T}\sum_{t=0}^{T-1}[f(x_t) - f(x)] \leq \frac{\|x_0 - x\|^2}{2\eta T} + \frac{\eta G^2}{2}. \tag{5.3}$$

For any fixed T and x, the bound on the right-hand side is optimized by the choice

$$\eta = \frac{\|x_0 - x\|}{G}\frac{1}{\sqrt{T}},$$

which yields

$$\frac{1}{T}\sum_{t=0}^{T-1}[f(x_t) - f(x)] \leq \frac{G\|x_0 - x\|}{\sqrt{T}}. \tag{5.4}$$

It follows that

$$\min_{t\in\{0,\dots,T-1\}}\{f(x_t)\} \leq f(x) + \frac{G\|x_0 - x\|}{\sqrt{T}}.$$

Remark 5.19. It may seem funny that we are not assuming that $x = x^\star$, i.e., that x is an optimizer, in the above statement. But in fact the theorem is true as written. A key point to note is that we only have the complementary lower bound $f(x_t) - f(x) \geq 0$ when x is an optimizer. Ultimately the case $x = x^\star$ is the one that really interests us. The more general formulation is saying that for any x not too far away from our initial guess x_0, our optimizer eventually finds some x_t that is 'at worst almost as good' as x. This connects to the notion of a *regret bound* in online learning, where the objective itself f_t depends on the iteration t. The proof generalizes easily to the online context.

Remark 5.20. It also follows from (5.4) that

$$f(\overline{x}_T) - f(x) \leq \frac{G\|x_0 - x\|}{\sqrt{T}},$$

where

$$\overline{x}_T := \frac{1}{T}\sum_{t=0}^{T-1} x_t$$

is the T-th **Cesàro mean** of the iterates. Indeed, the convexity of f directly implies that $f(\overline{x}_T) \leq \frac{1}{T}\sum_{t=0}^{T-1} f(x_t)$.

Proof. This proof, like that of Theorem 4.26, is pretty inspired and involves a trick that is not very intuitive *a priori*.

Fix $x \in \mathbb{R}^n$. From the subgradient property of v_t, we know that

$$f(x) \geq f(x_t) + v_t \cdot (x - x_t)$$

for all t. It is useful to solve the subgradient descent iteration (5.2) for v_t

$$v_t = \frac{1}{\eta}(x_t - x_{t+1}) \tag{5.5}$$

and substitute into our inequality, yielding the upper bound

$$f(x_t) - f(x) \leq \frac{1}{\eta}(x_t - x_{t+1}) \cdot (x - x_t).$$

We are hoping to get something out of the right-hand side that can yield a telescoping sum.

To this end, as in the proof of Theorem 4.26, we will expand the right-hand side using a polarization identity

$$a \cdot b = \frac{1}{2}\left(\|a\|^2 + \|b\|^2 - \|a - b\|^2\right)$$

which holds for general $a, b \in \mathbb{R}^n$. (This is equivalent to the expansion $\|a-b\|^2 = \|a\|^2+\|b\|^2-2\,a\cdot b$.)

Applying with $a = x_t - x_{t+1}$ and $b = x - x_t$ we see that

$$\begin{aligned} f(x_t) - f(x) &\leq \frac{1}{2\eta}\left[\|x_t - x_{t+1}\|^2 + \|x - x_t\|^2 - \|x - x_{t+1}\|^2\right] \\ &= \frac{\eta}{2}\|v_t\|^2 + \frac{1}{2\eta}\left[\|x - x_t\|^2 - \|x - x_{t+1}\|^2\right], \end{aligned} \tag{5.6}$$

where we have used (5.5) in the last line to simplify the first term. Note that the remaining two terms will telescope!

Then fixing a positive T and summing we obtain

$$\begin{aligned} \sum_{t=0}^{T-1}\left[f(x_t) - f(x)\right] &\leq \frac{\eta}{2}\sum_{t=0}^{T-1}\|v_t\|^2 + \frac{1}{2\eta}\|x - x_0\|^2 - \frac{1}{2\eta}\|x - x_T\|^2 \\ &\leq \frac{\eta T G^2}{2} + \frac{\|x - x_0\|^2}{2\eta}. \end{aligned}$$

Dividing the inequality by T, we complete the proof of (5.3). The remaining parts of the theorem follow directly. □

5.5 Convergence of subgradient descent: infinite time horizon

As our previous discussions suggest, if we want subgradient descent (5.2) to converge over an infinite time horizon in the sense that $\lim_{t\to\infty} f(x_t) = f(x^\star)$, we must consider a step size η_t. The preceding theorem spiritually suggests the right choice of step size is $O(1/\sqrt{t})$.

Theorem 5.21 (Convergence of subgradient descent: decaying step size, infinite time horizon)**.** *Let f be convex on $\mathbb{R}^n$ with G-bounded subgradients, and fix any $x \in \mathbb{R}^n$. Consider subgradient descent (5.2) with*

$$\eta_t = \frac{\|x_0 - x\|}{G}\sqrt{\frac{1}{t+1}}, \quad t = 0, 1, 2, \ldots \tag{5.7}$$

For any positive integer T and any $x \in \mathbb{R}^n$,

$$\min_{t\in\{0,1,\ldots,T-1\}}\{f(x_t)\} \leq f(x) + \frac{G\|x_0 - x\|}{\sqrt{T}}\left(1 + \frac{1}{4}\log T\right).$$

Remark 5.22. Plugging in $x = x^\star$, the theorem says that within $T-1$ steps, subgradient descent is guaranteed to hit an iterate achieving an optimality gap of $O(G \log T/\sqrt{T})$. Apparently the log factor cannot be removed without introducing additional complications / assumptions.

Remark 5.23. By analogy to Remark 5.20, a suitable statement can be made in terms of a *weighted* Cesàro mean, as shall become clear from the proof.

Proof. Following the same argument as in the proof of Theorem 5.18 up to (5.6), with η_t in place of η, we find that

$$f(x_t) - f(x) \leq \frac{\eta_t}{2}\|v_t\|^2 + \frac{1}{2\eta_t}\left[\|x - x_t\|^2 - \|x - x_{t+1}\|^2\right].$$

Then multiply both sides by η_t, use the bound $\|v_t\| \leq G$, and sum both sides over $t = 0, 1, \ldots, T-1$ (using telescoping) to conclude

$$\begin{aligned}\sum_{t=0}^{T-1} \eta_t \left[f(x_t) - f(x)\right] &\leq \frac{1}{2}\left(\sum_{t=0}^{T-1} \eta_t^2\right) G^2 + \frac{1}{2}\|x - x_0\|^2 - \|x - x_T\|^2 \\ &\leq \frac{1}{2}\left(\sum_{t=0}^{T-1} \eta_t^2\right) G^2 + \frac{1}{2}\|x - x_0\|^2.\end{aligned}$$

We divide both sides by $\sum_{t=0}^{T-1} \eta_t$ to interpret the left-hand side as a weighted average of the values $f(x_t) - f(x)$ over $t = 0, \ldots, T-1$, which must be lower-bounded by the minimum among these values.

We conclude that there exists $t' \in \{0, \ldots, T-1\}$ such that

$$\begin{aligned}f(x_{t'}) - f(x) &\leq \frac{1}{2}\frac{\sum_{t=0}^{T-1} \eta_t^2}{\sum_{t=0}^{T-1} \eta_t} G^2 + \frac{1}{2}\frac{1}{\sum_{t=0}^{T-1} \eta_t}\|x - x_0\|^2 \\ &= \frac{1}{2} G \|x_0 - x\| \; \frac{1 + \sum_{t=0}^{T-1} \frac{1}{t+1}}{\sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}}}, \end{aligned} \tag{5.8}$$

where in the last expression we have substituted the definition (5.7) of η_t and performed elementary simplifications.

First want to upper bound the sum in the numerator, which is precisely the T-th Harmonic sum. By comparison to the integral of $\frac{1}{u}\,du$, we can bound

$$\sum_{t=0}^{T-1} \frac{1}{t+1} = 1 + \sum_{j=2}^{T} \frac{1}{j} \leq 1 + \int_1^T \frac{1}{u}\,du = 1 + \log T.$$

Next we want to lower bound the sum in the denominator by comparison to an integral:

$$\sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}} = \sum_{j=1}^{T} \frac{1}{\sqrt{j}} \geq \int_1^{T+1} \frac{1}{\sqrt{u}}\,du = 2\sqrt{T+1} - 2.$$

Combining these bounds, we

$$\frac{1 + \sum_{t=0}^{T-1} \frac{1}{t+1}}{\sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}}} \leq \frac{2 + \log T}{2\sqrt{T+1} - 2} \sim \frac{\log T}{2\sqrt{T}}$$

where the asymptotic equality holds as $T \to \infty$.

In fact, it is possible to show a slightly better nonasymptotic bound

$$\frac{1 + \sum_{t=0}^{T-1} \frac{1}{t+1}}{\sum_{t=0}^{T-1} \frac{1}{\sqrt{t+1}}} \leq \frac{4 + \log T}{2\sqrt{T}} \sim \frac{\log T}{2\sqrt{T}},$$

which shares the same asymptotics. We omit the details required to prove this as they are not qualitatively important.

Now substituting into (5.8) yields

$$f(x_{t'}) - f(x) \leq G\|x_0 - x\| \frac{4 + \log T}{4\sqrt{T}} = \frac{G\|x_0 - x\|}{\sqrt{T}} \left(1 + \frac{1}{4} \log T\right).$$

This completes the proof. □

---

6 Proximal gradient method

Next we consider a substantial generalization of gradient descent called the proximal gradient method. In particular, this method unlocks a completely different approach to optimizing certain kinds of non-smooth functions, while retaining the same convergence theory as gradient descent and avoiding the $O(1/\sqrt{T})$ convergence rate of subgradient descent. However, the possibility of achieving this win depends very much on the structure of the non-smoothness. We will see that the proximal gradient method also enables a gradient-descent-like approach to certain *constrained* optimization problems. This perspective defines the projected gradient method.

6.1 Meta-algorithm

In this section, we consider an optimization problem where the objective F splits into two terms:

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad F(x), \quad \text{where} \quad F(x) = f(x) + g(x). \tag{6.1}$$

Here $f : \mathbb{R}^n \to \mathbb{R}$ is at least L-smooth, and $g : \mathbb{R}^n \to \mathbb{R}$ is convex but otherwise arbitrary for now (possibly non-smooth).

Given our current iterate x_t, the **proximal gradient method** defines the subsequent iterate as

$$x_{t+1} = \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ f(x_t) + \nabla f(x_t) \cdot (x - x_t) + g(x) + \frac{1}{2\eta}\|x - x_t\|^2 \right\}. \tag{6.2}$$

The idea is that we perform a first-order Taylor expansion of the smooth part but leave the non-smooth part exact within the argmin. By Lemma 4.9 (from Section 4.3 on the 'variational interpretation of gradient descent'), ordinary gradient descent is recovered in the case where $g \equiv 0$.

Note that we can still interpret this as a **majorization-minimization** algorithm in the sense of Section 4.3, provided that f is L-smooth and $\eta \leq \frac{1}{L}$, because in this case the objective within the argmin of (6.2) majorizes F.

The proximal gradient method is a **meta-algorithm** and leaves unspecified the details of implementing the exact minimization within (6.2). Indeed, the implementation of (6.2) is *not necessarily tractable* for any given choice of g. We are interested especially in choices of g for which the minimization within (6.2) *can be performed in closed form*, in which case (6.2) defines an implementable algorithm. However, the convergence theory for the abstract update (6.2) permits a fairly general convex g, as we shall elucidate below.

Ultimately, *the convergence theory will look almost exactly the same as that of ordinary gradient descent, phrased in terms of the smoothness and/or strong convexity of the f term alone.* Importantly, the smoothness of g does not affect the convergence rates.

6.2 The proximal mapping

The proximal gradient update (6.2) can be rephrased in terms of a general construction called the proximal mapping.

Definition 6.1 (Proximal mapping)**.** For a convex function $h : \mathbb{R}^n \to \mathbb{R}$, the **proximal mapping** of h, denoted $\operatorname{prox}_h : \mathbb{R}^n \to \mathbb{R}^n$, is defined for $y \in \mathbb{R}^n$ by

$$\operatorname{prox}_h(y) := \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \left\{ h(x) + \frac{1}{2}\|x - y\|^2 \right\}. \tag{6.3}$$

Remark 6.2. The argmin is well-defined because the map $x \mapsto h(x) + \frac{1}{2}\|x-y\|^2$ is automatically strongly convex, hence admits a unique minimizer (cf. Lemmas 2.62 and 2.70).

Definition 6.3 (Extending the proximal mapping, domain and properness)**.** More generally, we can extend the definition of prox_h to any convex h taking values in $\mathbb{R} \cup \{+\infty\}$, as long as h is l.s.c. For such a general h, it is useful to define the **domain** of h as

$$\operatorname{dom} h = \{x \in \mathbb{R}^n \,:\, h(x) < +\infty\},$$

which must necessarily be a convex set. Then it is equivalent to define

$$\operatorname{prox}_h(y) := \operatorname*{argmin}_{x \in \operatorname{dom} h} \left\{ h(x) + \frac{1}{2}\|x-y\|^2 \right\}.$$

The argmin is still well-defined by strong convexity, as long as $\operatorname{dom} h$ is nonempty, again by Lemmas 2.62 and 2.70 (the latter requiring the l.s.c. property).

Any convex $h : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ satisfying $\operatorname{dom} h \neq \emptyset$ is called **proper**. The preceding discussion ensures that the definition of prox_h extends to any proper, l.s.c. convex function h.

Remark 6.4. The projected gradient method can be understood in this extended framework, so it is convenient to generalize our perspective.

Our motivation for defining the proximal mapping is the following lemma, which also explains the origin of the name 'proximal gradient method.'

Lemma 6.5. *The proximal gradient method update rule (6.2) can be written equivalently as*

$$x_{t+1} = \operatorname{prox}_{\eta g}\left(x_t - \eta \nabla f(x_t)\right).$$

Remark 6.6. Thus the tractability of the proximal gradient method boils down to the tractability of evaluating $\operatorname{prox}_{\eta g}(y)$ for arbitrary $\eta > 0$ and $y \in \mathbb{R}^n$.

Proof. Consider completing the square in the objective minimized by the proximal gradient method (6.2), using the polarization identity $\|u+v\|^2 = \|u\|^2 + \|v\|^2 + 2\,u \cdot v$:

$$\begin{aligned} &f(x_t) + \nabla f(x_t) \cdot (x - x_t) + g(x) + \frac{1}{2\eta}\|x - x_t\|^2 \\ &\quad = f(x_t) + g(x) + \frac{1}{2\eta}\|(x - x_t) + \eta \nabla f(x_t)\|^2 - \frac{\eta}{2}\|\nabla f(x_t)\|^2 \\ &\quad = g(x) + \frac{1}{2\eta}\|x - [x_t - \eta \nabla f(x_t)]\,\|^2 + \text{const.}, \end{aligned}$$

where const. indicates terms that are independent of x. Then minimization over x of this objective is equivalent to minimization of

$$\eta g(x) + \frac{1}{2}\|x - [x_t - \eta \nabla f(x_t)]\,\|^2.$$

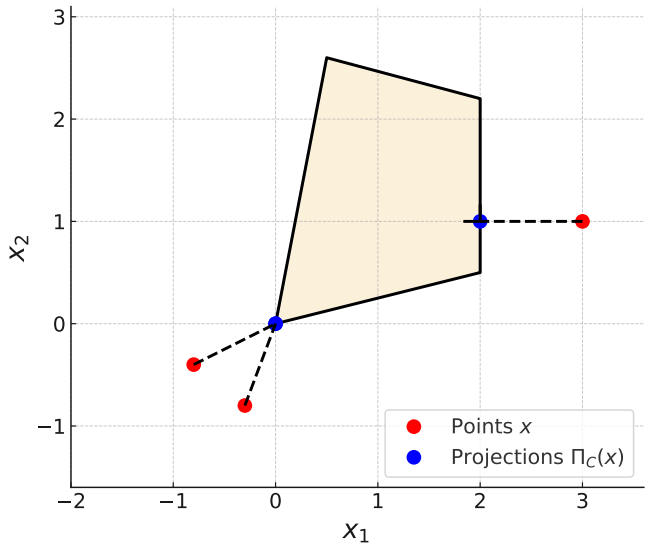

Figure 6.1: Depiction of projection onto a convex set. The concept generalizes the orthogonal projection onto a subspace.

The optimizer of this expression is precisely $\operatorname{prox}_{\eta g}(x_t - \eta \nabla f(x_t))$, by the definition of the proximal mapping. □

6.3 Projected gradient method

The projected gradient method is recovered from the proximal gradient method when we choose g to be an indicator function in the following sense.

Definition 6.7. For any set $C \subset \mathbb{R}^n$, define the **indicator function** $\iota_C : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ for C by

$$\iota_C(x) = \begin{cases} 0, & x \in C, \\ +\infty, & x \notin C. \end{cases}$$

Remark 6.8. If C is a nonempty, closed convex set, then ι_C is a proper, l.s.c. convex function in the sense of Definition 6.3.

Observe that under the choice $g = \iota_C$, the unconstrained problem (6.1) reduces to the general constrained optimization problem

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & x \in C. \end{aligned} \tag{6.4}$$

Thus the proximal gradient method framework furnishes an abstract algorithm even for solving constrained problems! It can be phrased more succinctly in terms of the projection onto C.

Definition 6.9 (Projection)**.** For a nonempty, closed convex set $C \subset \mathbb{R}^n$, define the *projection onto* C, which is a map $\Pi_C : \mathbb{R}^n \to C$ by

$$\Pi_C(y) := \operatorname*{argmin}_{x \in C} \|x - y\|.$$

Remark 6.10. Observe that for any $\eta > 0$, if $g = \iota_C$, then $\operatorname{prox}_{\eta g}(y) := \operatorname*{argmin}_{x \in C} \|x - y\|^2 = \Pi_C(y)$.

In light of Remark 6.10, the proximal gradient algorithm (6.2) in the case $g = \iota_C$ reduces to the update

$$x_{t+1} = \Pi_C \left(x_t - \nabla f(x_t) \right). \tag{6.5}$$

The algorithm defined by (6.5) is called the **projected gradient method**, for solving (6.4).

6.4 Concrete examples

Now we consider several examples of proximal mappings, starting in particular with several simple projections.

Exercise 6.11 (Projection onto ball)**.** For some fixed $x_0 \in \mathbb{R}^n$, consider the ball $C = \{x \in \mathbb{R}^n : \|x - x_0\| \leq R\}$. Determine a formula for Π_C.

Exercise 6.12 (Projection onto nonnegative vectors)**.** Let $C = \{x \in \mathbb{R}^n : x \geq 0\}$. Determine a formula for Π_C.

Exercise 6.13 (Projection onto box)**.** Fix $a_1, b_1, \ldots, a_n, b_n \in \mathbb{R}$. Let

$$C = \{x \in \mathbb{R}^n : x_1 \in [a_1, b_1], \ldots, x_n \in [a_n, b_n]\}.$$

Determine a formula for Π_C.

We will be able to determine formulas for several more kinds of projection later in the course after our discussion of duality.

Now we turn to an important examples of proximal mappings that are not projections.

Exercise 6.14. Define $g : \mathbb{R}^n \to \mathbb{R}$ by $g(x) = \frac{1}{2} x^\top A x - b \cdot x$. Then for $\lambda \geq 0$ and $x \in \mathbb{R}^n$, show that

$$\operatorname{prox}_{\lambda g}(x) = (I + \lambda A)^{-1}(x - \lambda b).$$

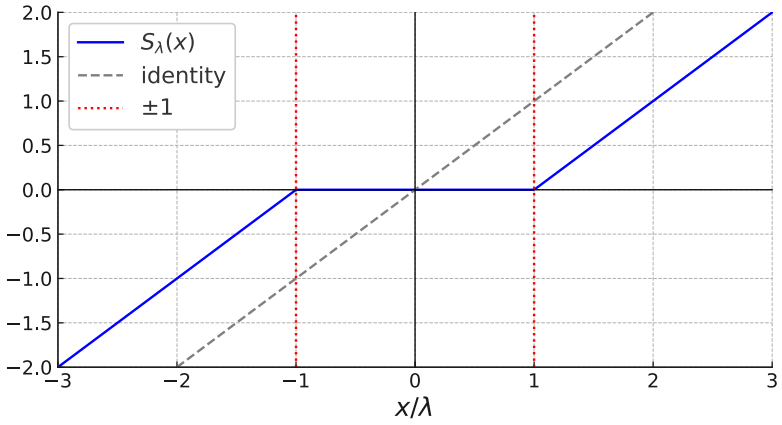

Figure 6.2: Depiction of the shrinkage operator $S_\lambda(x)$ in the scalar case. For $x \in \mathbb{R}^n$, the same operation is applied elementwise.

Example 6.15. Although g as in Exercise (6.14) is smooth, it may still be useful to consider within applications of the proximal gradient method. For example, consider a linear solve of the form $Ax = b$ where $A = A_1 + A_2$ and both terms are positive definite. Suppose moreover that $\|A_2\| \gg \|A_1\|$ but linear solves of the form $(I + \lambda A_2)y = c$ are easy to perform. This scenario arises frequently in numerical PDE, for example, where A_2 may be a discretization of the Laplacian operator.

Then applying the proximal gradient method to

$$F(x) = \frac{1}{2}x^\top Ax - b \cdot x = \underbrace{\frac{1}{2}x^\top A_1 x - b \cdot x}_{=: f(x)} + \underbrace{\frac{1}{2}x^\top A_2 x}_{=: g(x)}$$

yields the algorithm

$$x_{t+1} = (I + \eta A_2)^{-1} \left[x_t - \eta(A_1 x - b)\right].$$

Provided that we can efficiently perform the linear solves required by the update, this can be an effective algorithm, because the convergence rate will be controlled only by $\|A_1\|$. If A_1 is in addition well-conditioned, then we will even get a good exponential convergence rate in terms of its condition number.

Exercise 6.16 (Soft thresholding)**.** Define $g : \mathbb{R}^n \to \mathbb{R}$ by $g(x) = \|x\|_1$. Show that for $\alpha \geq 0$,

$$\left[\operatorname{prox}_{\alpha g}(x)\right]_i = \begin{cases} x_i - \alpha, & x_i \geq \alpha, \\ 0, & |x_i| \leq \alpha, \\ x_i + \alpha, & x_i \leq -\alpha, \end{cases}$$

for $i = 1, \ldots, n$.

We can more compactly write

$$\operatorname{prox}_{\alpha g}(y) = S_\alpha(y) := [y - \alpha \mathbf{1}_n]_+ - [-y - \alpha \mathbf{1}_n]_+,$$

where $[\,\cdot\,]_+$ denotes the elementwise $\max(0, \cdot)$ operation.

The function S_α is called the **soft thresholding** or **shrinkage-thresholding** operator, depicted in Figure 6.2.

Hint: The fact that $g(x)$ is *separable*, i.e., a sum of terms each involving only a single coordinate x_i, means that you can essentially reduce to the scalar case $n = 1$.

Example 6.17 (Iterative shrinkage-thresholding algorithm (ISTA))**.** Consider any objective of the form

$$F(x) = f(x) + \lambda \|x\|_1.$$

Then by Exercise 6.16, the proximal gradient method applied in this case is specified by the update

$$x_{t+1} = S_{\eta\lambda}\left(x_t - \eta \nabla f(x_t)\right).$$

This is called the **iterative shrinkage-thresholding algorithm (ISTA)**. Although F is non-smooth, the convergence rate of ISTA will be controlled only in terms of the smoothness (and possibly strong convexity) of f.

In the special case

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

motivated by L1-regularized least squares or LASSO (cf. Section 2.3.2), ISTA becomes more concretely:

$$x_{t+1} = S_{\eta\lambda}\left(x_t - \eta A^\top [Ax - b]\right).$$

Example 6.18 (Low-rank matrix completion)**.** Suppose I am give entries A_{ij} of an unknown matrix $A \in \mathbb{R}^{m\times n}$, only for $(i,j) \in \mathcal{S}$ (some subset). Can I estimate A fully just from the known entries, under the assumption that the rank of A is low? (The Netflix Prize problem can be phrased as a problem of this form.)

One relaxed approach to this problem takes the following form:

$$\underset{X\in\mathbb{R}^{m\times n}}{\text{minimize}} \left\{ \frac{1}{2} \sum_{(i,j)\in\mathcal{S}} (A_{ij} - X_{ij})^2 + \lambda \|X\|_* \right\}, \tag{6.6}$$

where $\lambda > 0$ is a regularization parameter and $\|X\|_* := \sum_{i=1}^{\min(m,n)} \sigma_i(X)$ denotes the **nuclear norm**. In this definition, $\sigma_i(X)$ is the i-th singular value of X.

In fact $\|\cdot\|_*$ defines a norm, hence a convex function. We will take this fact on faith here, though it is not obvious *a priori*. As a regularizer, the nuclear norm encourages sparsity of the singular values, i.e., low-rankness, in much the same ways as L1 regularization encourages sparsity. Like the 1-norm, the nuclear norm is nonsmooth.

We can suitably extend our previous developments by identifying $\mathbb{R}^{m\times n}$ with $\mathbb{R}^{mn}$ and then consider the proximal gradient method with nonsmooth term $g(X) := \|X\|_*$.

Without loss of generality assume that $m \geq n$ (considering the transposed problem if not). Then for $Y \in \mathbb{R}^{m\times n}$, let $Y = U\Sigma V^\top$ be an SVD, where $U \in \mathbb{R}^{m\times m}$ and $V \in \mathbb{R}^{n\times n}$ are orthogonal matrices and

$$\Sigma = \begin{pmatrix} \operatorname{diag}[\sigma_1, \ldots, \sigma_n] \\ \mathbf{0}_{(m-n)\times 0} \end{pmatrix}$$

is built from the singular values $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$.

It is possible to show that for any $\alpha > 0$,

$$\operatorname{prox}_{\alpha\|\cdot\|_*}(Y) = U \begin{pmatrix} \operatorname{diag}[S_\alpha(\sigma_1), \ldots, S_\alpha(\sigma_n)] \\ \mathbf{0}_{(m-n)\times 0} \end{pmatrix} V^\top,$$

where $S_\lambda : \mathbb{R} \to \mathbb{R}$ denotes the scalar soft thresholding operator defined in Exercise 6.16. In words, *the proximal operator can be computed by taking the SVD and replacing the singular values with soft-thresholded singular values.*

Then by setting

$$f(X) := \frac{1}{2} \sum_{(i,j)\in\mathcal{S}} (A_{ij} - X_{ij})^2 = \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{n} M_{ij}(A_{ij} - X_{ij})^2,$$

where we define a 'mask' matrix entrywise by:

$$M_{ij} := \begin{cases} 1, & (i,j) \in \mathcal{S}, \\ 0, & (i,j) \notin \mathcal{S}, \end{cases}$$

we can apply the proximal gradient method to the objective $F(X) = f(X) + g(X)$ from (6.6).

We can compute $\nabla f(X) = M \odot (X - A)$, where $\odot$ denotes the Hadamard or entrywise product, yielding the update

$$X_{t+1} = \operatorname{prox}_{\eta\lambda\|\cdot\|_*} \left(X_t - \eta\, M \odot (X - A) \right),$$

in which the prox map can be computed using the SVD.

6.5 Smooth and strongly convex case

Now we delve into the convergence theory, starting with the smooth and strongly convex case. Implicitly, when we talk about proximal gradient descent, we will assume throughout that the objective is $F = f + g$ where g satisfies the conditions of Definition 6.3 such that the proximal operator $\text{prox}_{\eta g}$ is well-defined.

Theorem 6.19 (Convergence of proximal gradient method: smooth and strongly convex case)**.** *Let f be L-smooth and μ-strongly convex on $\mathbb{R}^n$, and consider the proximal gradient method (6.2) with step size $\eta = \frac{1}{L}$. Then for all $t = 0, 1, 2, \ldots$, we have*

$$F(x_t) - F(x^\star) \leq \left(1 - \frac{\mu}{L}\right)^t \left[F(x_0) - F(x^\star)\right].$$

Remark 6.20. This is a generalization of Theorem 4.14, which is recovered in the case $g \equiv 0$.

Proof. It is convenient to fix the notation $x = x_t$ and $x^+ = x_{t+1}$ for some arbitrary t. It suffices to show that

$$F(x^+) - F(x^\star) \leq (1 - \mu/L)\left[F(x) - F(x^\star)\right].$$

By analogy to (4.5), define $F_{\eta,x} : \mathbb{R}^n \to \mathbb{R}$ by

$$F_{\eta,x}(y) := f(x) + \nabla f(x) \cdot (y - x) + \frac{1}{2\eta}\|x - y\|^2 + g(y), \quad y \in \mathbb{R}^n.$$

By the L-smoothness of f (by way of Theorem 3.3), since $\eta = \frac{1}{L}$ we know that

$$F(y) \leq F_{\eta,x}(y), \quad \text{for all } y \in \mathbb{R}^n.$$

(We can say that $F_{\eta,x}$ *majorizes* F in the sense of Section 4.3.)

Now by the update rule (6.2), $x^+ = \operatorname{argmin}_{y \in \mathbb{R}^n} F_{\eta,x}(y)$. Therefore, for any $y \in \mathbb{R}^n$, we have

$$F(x^+) \leq F_{\eta,x}(x^+) \leq F_{\eta,x}(y).$$

Then to get an upper bound on $F(x^+)$, we have the freedom to plug in any value of y. Let us consider $x_\theta := x + \theta(x^\star - x)$, or $x_\theta = (1 - \theta)x + \theta x^\star$, where $\theta \in [0, 1]$. We will optimize the choice of θ later. For now, compute

$$\begin{aligned} F(x^+) \leq F_{\eta,x}(x_\theta) &= f(x) + \nabla f(x) \cdot (x_\theta - x) + \frac{1}{2\eta}\|x_\theta - x\|^2 + g(x_\theta) \\ &= f(x) + \theta\, \nabla f(x) \cdot (x^\star - x) + \frac{\theta^2}{2\eta}\|x^\star - x\|^2 + g(x_\theta) \\ &\leq f(x) + \theta\, \nabla f(x) \cdot (x^\star - x) + \frac{\theta^2}{2\eta}\|x^\star - x\|^2 + (1 - \theta)\, g(x) + \theta\, g(x^\star), \end{aligned} \tag{6.7}$$

where in the first step we plugged in $x_\theta - x = \theta(x^\star - x)$, and in the last step we used the convexity of g.

Next, by strong convexity observe that

$$f(x^\star) \geq f(x) + \nabla f(x) \cdot (x^\star - x) + \frac{\mu}{2}\|x^\star - x\|^2,$$

hence after rearranging:

$$\nabla f(x) \cdot (x^\star - x) \leq f(x^\star) - f(x) - \frac{\mu}{2}\|x^\star - x\|^2.$$

We can plug into (6.7) to obtain

$$F(x^+) \leq (1-\theta)f(x) + \theta\, f(x^\star) + \frac{\theta}{2}\left(\frac{\theta}{\eta} - \mu\right)\|x^\star - x\|^2 + (1-\theta)\, g(x) + \theta\, g(x^\star).$$

Now we can finally fix $\theta := \eta\mu = \mu/L \leq 1$ (*how do we know* ≤ 1*?*) to cancel the third term and obtain

$$\begin{aligned} F(x^+) &\leq (1-\theta)f(x) + \theta\, f(x^\star) + (1-\theta)\, g(x) + \theta\, g(x^\star) \\ &= (1-\theta)F(x) + \theta\, F(x^\star). \end{aligned}$$

In turn

$$F(x^+) - F(x^\star) \leq (1-\theta)F(x) + (\theta - 1)\, F(x^\star) = (1-\theta)\left[F(x) - F(x^\star)\right].$$

Since $\theta = \mu/L$, this completes the proof. □

6.6 Contraction mapping perspective

Now we will explore an alternative pathway to convergence in this case that builds on Section 4.6 and directly establishes convergence of the distance to the optimizer $\|x_t - x^\star\|$. First we show a fundamental fact about proximal operators.

Lemma 6.21 (Proximal operator is nonexpansive)**.** *For any convex h as in Definition 6.3,* prox_h *is nonexpansive.*

Proof. Let $x, y \in \mathbb{R}^n$. Let $\tilde{x} := \mathrm{prox}_h(x)$ and $\tilde{y} := \mathrm{prox}_h(y)$. It suffices to show that $\|\tilde{x} - \tilde{y}\| \leq \|x - y\|$.

By the optimality conditions defining the proximal operator, $x - \tilde{x} \in \partial g(\tilde{x})$ and $y - \tilde{y} \in \partial g(\tilde{y})$. Therefore

$$\begin{aligned} g(\tilde{y}) &\geq g(\tilde{x}) + (x - \tilde{x}) \cdot (\tilde{y} - \tilde{x}), \\ g(\tilde{x}) &\geq g(\tilde{y}) + (y - \tilde{y}) \cdot (\tilde{x} - \tilde{y}). \end{aligned}$$

Then add both inequalities and cancel to obtain

$$\begin{aligned} 0 &\geq (x - \tilde{x}) \cdot (\tilde{y} - \tilde{x}) + (y - \tilde{y}) \cdot (\tilde{x} - \tilde{y}) \\ &= -(x - \tilde{x}) \cdot (\tilde{x} - \tilde{y}) + (y - \tilde{y}) \cdot (\tilde{x} - \tilde{y}) \\ &= ([y - x] + [\tilde{x} - \tilde{y}]) \cdot (\tilde{x} - \tilde{y}) \\ &= \|\tilde{x} - \tilde{y}\|^2 - (\tilde{x} - \tilde{y}) \cdot (x - y), \end{aligned}$$

hence

$$\|\tilde{x} - \tilde{y}\|^2 \leq (\tilde{x} - \tilde{y}) \cdot (x - y) \leq \|\tilde{x} - \tilde{y}\|\, \|x - y\|,$$

where the last inequality follows from Cauchy-Schwarz. Cancelling terms completes the proof. □

Now recall the definition (4.6) of the 'iteration map' of gradient descent with step size $\eta > 0$:

$$\phi(x) = x - \eta \nabla f(x). \tag{6.8}$$

We already showed (Lemma 4.21) that ϕ is an α-contraction map for $\alpha = 1 - \mu/L.$, provided that f is L-smooth and μ-strongly convex.

Now the iteration map of proximal gradient descent is simply the composition $\text{prox}_{\eta g} \circ \phi$. Then the following result follows immediately.

Corollary 6.22 (Iterate convergence of proximal gradient method: smooth and strongly convex case)**.** *Let f be L-smooth and μ-strongly convex on $\mathbb{R}^n$, and consider the proximal gradient method (6.2) with step size $\eta = \frac{1}{L}$. Then for all $t = 0, 1, 2, \ldots$, it holds that*

$$\|x_t - x^\star\| \leq \left(1 - \frac{\mu}{L}\right)^t \|x_0 - x^\star\|.$$

Remark 6.23. This is a generalization of Corollary 4.23, which is recovered in the case $g \equiv 0$.

Proof. By the Banach fixed-point theorem, it suffices to show that $\text{prox}_{\eta g} \circ \phi$ is an α-contraction map, where $\alpha = 1 - \mu/L$. Let $x, y \in \mathbb{R}^n$ be arbitrary. Then

$$\|\text{prox}_{\eta g} \circ \phi(x) = \text{prox}_{\eta g} \circ \phi(y)\| \leq \|\phi(x) - \phi(y)\| \leq \alpha \|x - y\|,$$

by the fact that $\text{prox}_{\eta g}$ is nonexpansive and the fact that ϕ is an α-contraction map (Lemma 4.21). This completes the proof. □

6.7 Smooth and convex case

Next we give up strong convexity and attempt to generalize Theorem 4.26 for ordinary gradient descent.

Before proceeding with the convergence analysis, we make a useful definition.

Definition 6.24. Given suitable g and $\eta > 0$, define $G_\eta : \mathbb{R}^n \to \mathbb{R}^n$ by

$$G_\eta(x) = \frac{1}{\eta}\left[x - \text{prox}_{\eta g}(x - \eta \nabla f(x))\right].$$

Remark 6.25. With this definition, the proximal gradient descent can be written as

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

and we can view G_η as generalizing the role of ∇f in the ordinary gradient descent update.

Now we proceed with a lemma which generalizes the descent lemma.

Lemma 6.26 (Generalized descent lemma)**.** *If f is L-smooth on $\mathbb{R}^n$ and $\eta \leq \frac{1}{L}$, then the iterates of the proximal gradient method (6.2) satisfy*

$$f(x_{t+1}) \leq f(x_t) - \eta \nabla f(x_t) \cdot G_\eta(x_t) + \frac{\eta}{2}\|G_\eta(x_t)\|^2$$

for all $t = 0, 1, 2, \dots$.

Remark 6.27. For now we consider a fixed step size, but this lemma will suggest the correct stopping condition for backtracking line search.

Proof. By the definition of G_η (Definition 6.24) and the L-smoothness of f, we have

$$\begin{aligned} f(x_{t+1}) &= f(x_t - \eta G_\eta(x)) \\ &\leq f(x_t) - \eta \nabla f(x_t) \cdot G_\eta(x_t) + \frac{\eta^2 L}{2}\|G_\eta(x_t)\|^2, \end{aligned}$$

which implies what we want to show since $\eta \leq \frac{1}{L}$. □

Lemma 6.28. *If f is convex and L-smooth on $\mathbb{R}^n$ and $\eta \leq \frac{1}{L}$, then the iterates of the proximal gradient method (6.2) satisfy*

$$F(x_{t+1}) \leq F(z) + G_\eta(x_t) \cdot (x_t - z) - \frac{\eta}{2}\|G_\eta(x_t)\|^2$$

for all $z \in \mathbb{R}^n$ and all $t = 0, 1, 2, \dots$.

Remark 6.29. In particular, taking $z = x_t$ we recover the inequality

$$F(x_{t+1}) \leq F(x_t) - \frac{\eta}{2}\|G_\eta(x_t)\|^2,$$

which also generalizes the ordinary descent lemma and directly implies that the objective $F(x_t)$ is nonincreasing over the trajectory.

Proof $^{(\star)}$. In this proof it is useful to set $x = x_t$ and $x^+ = x_{t+1}$. In turn we have $G_\eta(x) = \frac{1}{\eta}(x - x^+)$.

Now recall the definition (6.2) of $x^+ = x_{t+1}$:

$$x^+ = \operatorname*{argmin}_{w \in \mathbb{R}^n} \left\{ f(x) + \nabla f(x) \cdot (w - x) + g(w) + \frac{1}{2\eta}\|w - x\|^2 \right\}.$$

Now by the (subgradient) first-order optimality condition of this optimization problem, we know that

$$0 \in \nabla f(x) + \partial g(x^+) + \frac{1}{\eta}(x^+ - x).$$

After rearranging we see that

$$v := G_\eta(x) - \nabla f(x) \in \partial g(x^+).$$

Now fix an arbitrary $z \in \mathbb{R}^n$, then expand and bound

$$\begin{aligned} F(x^+) &= f(x^+) + g(x^+) \\ &\overset{\text{(i)}}{\leq} f(x) - \eta \nabla f(x) \cdot G_\eta(x) + \frac{\eta}{2}\|G_\eta(x)\|^2 + g(x^+) \\ &\overset{\text{(ii)}}{\leq} [f(z) + \nabla f(x) \cdot (x - z)] - \eta \nabla f(x) \cdot G_\eta(x) + \frac{\eta}{2}\|G_\eta(x)\|^2 \\ &\qquad + \big[g(z) + v \cdot (x^+ - z)\big] \\ &= F(z) + [\nabla f(x) \cdot (x - z) + v \cdot (x - z) - \eta\, v \cdot G_\eta(x)] \\ &\qquad - \eta \nabla f(x) \cdot G_\eta(x) + \frac{\eta}{2}\|G_\eta(x)\|^2. \end{aligned}$$

Here (i) follows from Lemma 6.26, while (ii) follows from the convexity of f and the subgradient property of $v \in \partial g(x^+)$.

Now expand the bracketed term in the last expression:

$$\begin{aligned} &\nabla f(x) \cdot (x - z) + v \cdot (x - z) - \eta\, v \cdot G_\eta(x) \\ &\quad = [\nabla f(x) + v] \cdot (x - z) - \eta\, v \cdot G_\eta(x) \\ &\quad = G_\eta(x) \cdot (x - z) - \eta\, \|G_\eta(x)\|^2 + \eta\, \nabla f(x_t) \cdot G_\eta(x), \end{aligned}$$

where we have used the definition $v = G_\eta(x) - \nabla f(x)$ in the last line. Substituting back into our bound, we see

$$F(x^+) \leq F(z) + G_\eta(x) \cdot (x - z) - \frac{\eta}{2}\|G_\eta(x)\|^2,$$

which completes the proof. □

Finally we are ready to prove our convergence theorem.

Theorem 6.30 (Convergence of proximal gradient method: smooth convex case)**.** *Let f be convex and L-smooth on $\mathbb{R}^n$ and suppose that $x^\star$ is any minimizer. Consider gradient descent (6.2) with step size $\eta = \frac{1}{L}$. Then for all $t = 1, 2, \ldots$, we have*

$$F(x_t) - F(x^\star) \leq \frac{L\|x_0 - x^\star\|^2}{2t}.$$

Remark 6.31. This is a generalization of Theorem 4.26, which is recovered in the case $g \equiv 0$.

Proof $^{(\star)}$. Apply Lemma 6.28 with $z = x^\star$ to deduce:

$$\begin{aligned} F(x_{t+1}) - F(x^\star) &\leq G_\eta(x_t) \cdot (x_t - x^\star) - \frac{\eta}{2}\|G_\eta(x_t)\|^2 \\ &= -\eta \left(\frac{1}{2}\|G_\eta(x_t)\|^2 - G_\eta(x_t) \cdot \left[\eta^{-1}(x_t - x^\star)\right] \right) \\ &\overset{(\star)}{=} -\eta \left[\frac{1}{2}\|G_\eta(x_t) - \eta^{-1}(x_t - x^\star)\|^2 - \frac{\eta^{-2}}{2}\|x_t - x^\star\|^2 \right] \\ &= -\frac{\eta}{2}\|\eta^{-1}(x_t - x_{t+1}) - \eta^{-1}(x_t - x^\star)\|^2 + \frac{1}{\eta}\|x_t - x^\star\|^2 \\ &= \frac{1}{2\eta}\left(-\|x_{t+1} - x^\star\|^2 + \|x_t - x^\star\|^2 \right), \end{aligned}$$

where we have completed the square in $(\star)$.

Then perform the sum from $t = 0$ to arbitrary T and telescope to bound

$$\begin{aligned}\sum_{t=0}^{T} [F(x_{t+1}) - F(x^\star)] &\leq \frac{1}{2\eta}\|x_0 - x^\star\|^2 - \frac{1}{2\eta}\|x_T - x^\star\|^2 \\ &\leq \frac{1}{2\eta}\|x_0 - x^\star\|^2.\end{aligned}$$

Since $F(x_t)$ is nonincreasing (Remark 6.29), we have $F(x_{t+1}) \leq F(x_T)$ on the left hand side for all t in the sum, hence

$$F(x_T) - F(x^\star) \leq \frac{1}{2T\eta}\|x_0 - x^\star\|^2,$$

which concludes the proof. □

6.8 Backtracking line search

We hope to generalize the discussion of Section 6.29 to derive a backtracking line search algorithm for the proximal gradient descent method with variable step size η_t:

$$x_{t+1} = \operatorname{prox}_{\eta_t g}\left(x_t - \eta_t \nabla f(x_t)\right). \tag{6.9}$$

The key is offered by Lemma 6.26, which suggests the following criterion:

$$f(x_{t+1}) \leq f(x_t) - \eta_t \nabla f(x_t) \cdot G_{\eta_t}(x_t) + \frac{\eta_t}{2}\|G_{\eta_t}(x_t)\|^2, \tag{6.10}$$

where we recall G_η from Definition 6.24.

We know that this holds for $\eta_t \leq \frac{1}{L}$. But regardless—when this bound holds, the convergence analysis carries through almost exactly the same as above.

Therefore we can follow the discussion of Section 6.29, replacing our simpler criterion $f(x_{t+1}) \leq f(x_t) - \frac{\eta_t}{2}\|\nabla f(x_t)\|^2$ for gradient descent with its generalization (6.10). Specifically, we try an initial guess $\eta_t \leftarrow \eta_t^{(0)}$ and keep reducing η_t by a factor of τ until (6.10) holds, then output η_t. Once η_t drops below $1/L$, we know that (6.10) will hold. Therefore the output step size satisfies $\eta_t \geq \min\{\eta_t^{(0)}, \tau/L\}$.

Note that each 'try' within a line search requires one evaluation of f *and* one evaluation of the proximal operator. However, only a single evaluation of ∇f is required at the beginning of the line search. Hence if ∇f is a bottleneck relative to evaluating f and the proximal operator, then it makes sense to do a more careful line search, i.e., choose τ closer to 1.

As before, we suggest the simple heuristic for choosing the guess in terms of the last step size:

$$\eta_{t+1}^{(0)} = \gamma\, \eta_t,$$

where $\gamma \in (1, \tau^{-1})$ is a hyperparameter. This guarantees in particular that $\eta_t \geq \min\{\eta_0^{(0)}, \tau/L\}$ for all t.

Remark 6.32 (Convergence of proximal gradient descent with backtracking line search). Performing gradient descent (6.9) in which the step sizes are chosen via backtracking line search as outlined above, Theorems 6.19 and 6.30 hold with $\tilde{L} := \max\{L/\tau, 1/\eta_0^{(0)}\}$ in the place of L. If we make sure that our initial guess is sufficiently aggressive, i.e., satisfying $\eta_0^{(0)} \leq 1/L$, then we have simply $\tilde{L} = L/\tau$.

7 Acceleration

Our analysis of gradient descent (and more generally of the proximal gradient method) for convex L-smooth f yields the famous $O(1/t)$ convergence rate for the optimality gap, cf. Theorem 6.30.

In fact, at essentially no additional cost, it is possible to modify the proximal gradient method to obtain a convergence rate of $O(1/t^2)$. Moreover it is even possible to show that one can hope for no better using a single gradient query per iteration, though we will not prove this claim here. The resulting **acceleration** of gradient descent is called **(Nesterov's) accelerated gradient descent**. The accelerated variant of the proximal gradient method more generally is called the **accelerated proximal gradient method**.

In this section, we consider a general objective of the form $F = f + g$, where g is an arbitrary convex function (nice enough in the sense of Definition 6.3 such that the suitable proximal mapping is well-defined) and f is convex and L-smooth. Accelerated gradient descent will be recovered in the case $g \equiv 0$.

In the special case where $g \equiv \iota_C$ for a closed nonempty convex set C, our general algorithm recovers a suitable **accelerated projected gradient method** as a special case. Finally, in the special case where $g(x) = \lambda\|x\|_1$, we recover a famous algorithm called **FISTA** (*fast* iterative shrinkage-thresholding algorithm), which is the accelerated variant of ISTA.

In fact, acceleration techniques can be applied in the L-smooth and μ-strongly convex case as well, bringing the convergence rate from $O(e^{-t/\kappa})$ down to $O(e^{-t/\sqrt{\kappa}})$ in terms of the condition number $\kappa := L/\mu$. For solving positive definite systems via optimization of $f(x) = \frac{1}{2}x^\top Ax - b \cdot x$, this yields a convergence rate that is qualitatively the same as the conjugate gradient method! However, acceleration is less widely considered in this context because the suitable variant of the algorithm requires *a priori* knowledge of κ (or equivalently an upper bound on κ), which is in general not accessible in practice. Thus we will not consider this variant, though we comment that *restarted* accelerated gradient techniques have been considered in the literature to address this shortcoming.

In the following we will use **(A)PGM** as an abbreviation for the (accelerated) proximal gradient method, though we try to use this abbreviation sparingly to avoid confusion with the projected gradient special case.

7.1 Definition and basic intuition

APGM is defined by the following update rule:

$$\begin{cases} x_{t+1} = \operatorname{prox}_{\eta_t g}\left(y_t - \eta_t \nabla f(y_t)\right) \\ y_{t+1} = x_{t+1} + \gamma_t\left(x_{t+1} - x_t\right) \end{cases} \tag{7.1}$$

defining a sequence $x_t, y_t \in \mathbb{R}^n$, $t = 0, 1, 2, \ldots$. Given an initial guess x_0, we always set

$$y_0 = x_0 \tag{7.2}$$

to get started. Moreover γ_t, $t = 0, 1, 2, \ldots$, is a sequence of scalars that we concretely define as

$$\gamma_t := \frac{t}{t+3}. \tag{7.3}$$

In practice the step size sequence η_t, $t = 0, 1, 2, \ldots$, can be chosen constant $\eta_t := \eta \leq 1/L$ or via a backtracking line search procedure on which we elaborate below.

We will see how this mysterious choice arises, but to gain intuition, first just think of $\gamma_t > 0$ as arbitrary. The intuition of the APGM update (6.30) is to consider the PGM update $y_t \to x_{t+1}$

and then, instead of accepting this update, go even further forward in the search direction $x_{t+1} - x_t$ to define y_{t+1}. The coefficient $\gamma_t > 0$ dictates the extent by which we extrapolate our update forward. Note that $\gamma_t = 0$ would simply recover the PGM update, while a larger value of $\gamma_t > 0$ corresponds to a more 'aggressive' extrapolation. We can also interpret the γ_t term as a 'momentum' contribution.

Lemma 7.1. *APGM (7.1) is equivalently defined by the update rule*

$$\begin{cases} x_{t+1} = \operatorname{prox}_{\eta_t g}\left(y_t - \eta_t \nabla f(y_t)\right) \\ v_{t+1} = x_t + \frac{1}{\theta_t}(x_{t+1} - x_t) \\ y_{t+1} = (1 - \theta_{t+1})x_{t+1} + \theta_{t+1} v_{t+1} \end{cases} \tag{7.4}$$

for $x_t, v_t, y_t \in \mathbb{R}^n$, $t = 0, 1, 2, \ldots$ where

$$\theta_t := \frac{2}{t+2}, \tag{7.5}$$

in terms of which

$$\gamma_t = \theta_{t+1}\left(\frac{1}{\theta_t} - 1\right). \tag{7.6}$$

Remark 7.2. This equivalent formulation of APGM expresses the next iterate y_{t+1} as a convex combination of two guesses: x_{t+1} (the update from y_t suggested by PGM) and v_{t+1} (an extrapolation of the update $x_t \to x_{t+1}$). Henceforth we will take (7.4) as a definition specifying v_t in addition to x_t, y_t, $t = 0, 1, 2, \ldots$. Note that we are free to specify $v_0 = x_0$ (hence also $v_0 = y_0$) without altering the output of (7.4), and we adopt this convention going forward.

Remark 7.3. This lemma partially explains the mysterious formula for γ_t, but it remains to see why we choose $\theta_t = 2/(t+2)$. Ultimately, we will see that the key properties satisfied by this choice are that $\theta_t \in [0, 1]$ for all $t = 0, 1, 2, \ldots$ and that

$$\frac{1 - \theta_t}{\theta_t^2} \leq \frac{1}{\theta_{t-1}^2} \tag{7.7}$$

for all $t = 1, 2, \ldots$. This can be verified directly for our choice, but other choices of sequence satisfying these properties are possible. In particular, the sequence θ_t can be defined recursively to ensure that the inequality holds with equality for all t.

Proof. Simply expand the expression in the last line of the update rule (7.4) and verify that it recovers the last line of (7.1):

$$\begin{aligned} &(1 - \theta_{t+1})x_{t+1} + \theta_{t+1} v_{t+1} \\ &\quad = x_{t+1} - \theta_{t+1} x_{t+1} + \theta_{t+1}\left[x_t + \frac{1}{\theta_t}(x_{t+1} - x_t)\right] \\ &\quad = x_{t+1} + \theta_{t+1}\left[(x_t - x_{t+1}) + \frac{1}{\theta_t}(x_{t+1} - x_t)\right] \\ &\quad = x_{t+1} + \theta_{t+1}\left(\frac{1}{\theta_t} - 1\right)(x_{t+1} - x_t). \end{aligned}$$

Hence it remains only to show that $\theta_{t+1}\left(\frac{1}{\theta_t}-1\right)=\gamma_t=\frac{t}{t+3}$ by the definition (7.3).

To see this, substitute (7.5) and compute

$$\theta_{t+1}\left(\frac{1}{\theta_t}-1\right)=\frac{2}{t+3}\left(\frac{t+2}{2}-1\right)=\frac{2}{t+3}\left(\frac{t}{2}\right)=\frac{t}{t+3},$$

as desired. □

Lemma 7.4. *The iterates in APGM (7.4) satisfy*

$$v_{t+1}=v_t-\frac{\eta_t}{\theta_t}G_{\eta_t}(y_t).$$

Proof. Recall that G_η was defined (cf. Definition 6.24) precisely such that $y_t-\eta_t G_{\eta_t}(y_t)$ is the PGM update from y_t, i.e.,

$$x_{t+1}=y_t-\eta_t G_{\eta_t}(y_t).$$

Then rewrite y_t in terms of x_t and v_t, following (7.4):

$$x_{t+1}=(1-\theta_t)x_t+\theta_t v_t-\eta_t G_{\eta_t}(y_t).$$

Note that this even holds for $t=0$ by our convention $v_0=x_0$ (Remark 7.2).

Then substituting into the second line of (7.4), we obtain

$$\begin{aligned}
v_{t+1}&=x_t+\frac{1}{\theta_t}\left[x_{t+1}-x_t\right]\\
&=x_t+\frac{1}{\theta_t}\left[(1-\theta_t)x_t+\theta_t v_t-\eta_t G_{\eta_t}(y_t)-x_t\right]\\
&=x_t+\frac{1}{\theta_t}\left[-\theta_t x_t+\theta_t v_t-\eta_t G_{\eta_t}(y_t)\right]\\
&=v_t-\frac{\eta_t}{\theta_t}G_{\eta_t}(y_t),
\end{aligned}$$

as was to be shown. □

7.2 Convergence theorem

Now we state the celebrated $O(1/t^2)$ convergence theorem.

Theorem 7.5 (Convergence of accelerated proximal gradient method: smooth convex case)**.** *Let f be convex and L-smooth on $\mathbb{R}^n$ and suppose that $x^\star$ is any minimizer. Consider APGM (7.1) with step size $\eta=\frac{1}{L}$. Then for all $t=1,2,\ldots$, we have*

$$F(x_t)-F(x^\star)\le\frac{2L\|x_0-x^\star\|^2}{(t+1)^2}.$$

To prove the theorem, we use the following lemma to establish the progress made in one iteration.

Lemma 7.6. *Let f be convex and L-smooth on $\mathbb{R}^n$, and consider APGM (7.1) with step size $\eta \le \frac{1}{L}$. Then for all $t = 0, 1, 2, \ldots$, we have*

$$F(x_{t+1}) - F(x^\star) \le (1-\theta_t)\left[F(x_t) - F(x^\star)\right] + \frac{\theta_t^2}{2\eta}\left[\|v_t - x^\star\|^2 - \|v_{t+1} - x^\star\|^2\right]$$

Proof. Within the scope of the proof, we view t as fixed and set the notations $x = x_t$, $y = y_t$, $v = v_t$, $x^+ = x_{t+1}$, $y^+ = y_{t+1}$, $v^+ = v_{t+1}$, $\eta = \eta_t$, $\theta = \theta_t$.

Now since x^+ is obtained by applying on iteration of PGM to y, we can apply Lemma 6.28 (with y in place of x_t) to deduce that

$$F(x^+) \le F(z) + G_\eta(y) \cdot (y - z) - \frac{\eta}{2}\|G_\eta(y)\|^2$$

for any $z \in \mathbb{R}^n$.

Setting $z = x$ and $z = x^\star$, we obtain two inequalities:

$$F(x^+) \le F(x) + G_\eta(y) \cdot (y - x) - \frac{\eta}{2}\|G_\eta(y)\|^2$$

$$F(x^+) \le F(x^\star) + G_\eta(y) \cdot (y - x^\star) - \frac{\eta}{2}\|G_\eta(y)\|^2.$$

Then take a convex combination of these inequalities:

$$F(x^+) \le (1-\theta)F(x) + \theta F(x^\star) + G_\eta(y) \cdot (y - (1-\theta)x - \theta x^\star) - \frac{\eta}{2}\|G_\eta(y)\|^2.$$

Now by the third line of (7.4), we have that $y = (1-\theta)x + \theta v$, hence by substitution, we obtain:

$$F(x^+) \le (1-\theta)F(x) + \theta F(x^\star) + \theta G_\eta(y) \cdot (v - x^\star) - \frac{\eta}{2}\|G_\eta(y)\|^2.$$

Then we can complete the square involving the last two terms to obtain

$$\begin{aligned} F(x^+) &= (1-\theta)F(x) + \theta F(x^\star) + \frac{\theta^2}{2\eta}\left[\|v - x^\star\|^2 - \|v - x^\star - (\eta/\theta)G_\eta(y)\|^2\right] \\ &= (1-\theta)F(x) + \theta F(x^\star) + \frac{\theta^2}{2\eta}\left[\|v - x^\star\|^2 - \|v^+ - x^\star\|^2\right], \end{aligned}$$

where in the last line we used Lemma 7.4, which says here that $v^+ = v - (\eta/\theta)\, G_\eta(y)$.

Then by subtracting $F(x^\star)$ from both sides we conclude the proof. □

Proof of Theorem 7.5. For simplicity of notation, define the optimality gap $\Delta_t := F(x_t) - F(x^\star)$ for all t. Then Lemma 7.6 says, after dividing by θ_t^2, that

$$\frac{\Delta_{t+1}}{\theta_t^2} \le \frac{1-\theta_t}{\theta_t^2}\Delta_t + \frac{1}{2\eta}\left[\|v_t - x^\star\|^2 - \|v_{t+1} - x^\star\|^2\right] \tag{7.8}$$

for all t.

Now by (7.7) (cf. Remark 7.3) we know that

$$\frac{1-\theta_t}{\theta_t^2} \le \frac{1}{\theta_{t-1}^2},$$

hence by substituting this inequality into (7.8) we obtain

$$\frac{\Delta_{t+1}}{\theta_t^2} \leq \frac{\Delta_t}{\theta_{t-1}^2} + \frac{1}{2\eta}\left[\|v_t - x^\star\|^2 - \|v_{t+1} - x^\star\|^2\right]$$

for all $t = 0, 1, 2, \ldots$.

But then, setting $t \leftarrow t-1$ in this inequality, we also know

$$\frac{\Delta_t}{\theta_{t-1}^2} \leq \frac{\Delta_{t-1}}{\theta_{t-2}^2} + \frac{1}{2\eta}\left[\|v_{t-1} - x^\star\|^2 - \|v_t - x^\star\|^2\right],$$

which we can substitute into the preceding inequality, and so on, yielding, via telescoping:

$$\begin{aligned}\frac{\Delta_{t+1}}{\theta_t^2} &\leq \frac{1-\theta_0}{\theta_0^2}\Delta_0 + \frac{1}{2\eta}\left[\|v_0 - x^\star\|^2 - \|v_{t+1} - x^\star\|^2\right] \\ &\leq \frac{L}{2}\|x_0 - x^\star\|^2,\end{aligned}$$

where we have used in the last line the facts that $\theta_0 = 1$, $v_0 = x_0$, and $\eta = \frac{1}{L}$.

Then unpacking and using the fact that $\theta_t = 2/(t+2)$, we conclude that

$$F(x_{t+1}) - F(x^\star) \leq \frac{2L}{(t+2)^2}\|x_0 - x^\star\|^2.$$

Since this holds for arbitrary t, we have concluded the proof. □

7.3 Backtracking line search

Similarly to our discussion of backtracking line search for PGM (Section 6.8), we observe that the key inequality that we really require of our step size η_t is that the following inequality holds:

$$f(x_{t+1}) \leq f(y_t) - \eta_t \nabla f(y_t) \cdot G_\eta(y_t) + \frac{\eta_t}{2}\|G_{\eta_t}(y_t)\|^2.$$

Contrast with (6.10) and indeed note that y_t (not x_t) is the point from which we compute the PGM update in iteration t of APGM, yielding x_{t+1}.

The rest of the discussion carries over *mutatis mutandi*, using this inequality as the stopping criterion for backtracking line search.

Part III

Newton's method and its variants

We have seen that for L-smooth and μ-strongly convex objectives, gradient descent (4.1) with step size $\eta = 1/L$ converges exponentially fast in the sense that

$$\|x_t - x^\star\| \leq \alpha^t \|x_0 - x^\star\|, \quad t = 0, 1, \ldots,$$

where

$$\alpha := 1 - \mu/L.$$

More sharply, we showed in the proof of Corollary 4.23 that the distance to the optimizer contracts by the fixed ratio $\alpha \in [0, 1)$ in every iteration

$$\|x_{t+1} - x^\star\| \leq \alpha \|x_t - x^\star\| \quad t = 0, 1, \ldots.$$

We will actually call this type of convergence *linear* convergence due to its position in a more general framework that we explain below. Within this nomenclature, Newton's method achieves *quadratic* convergence. Intuitively speaking, linear convergence means that each additional digit of accuracy costs about the same as the last one. For quadratically (or more generally *superlinearly*) converging methods, each digit is cheaper than the last. This qualitative difference has profound implications. However, each iteration of Newton's method costs more than an iteration of gradient descent because it requires us to solve a linear system. *Quasi-Newton* methods have been proposed to alleviate the additional cost while retaining some of the advantages of Newton's method.

8 Orders of convergence

Before introducing and discussing the convergence of Newton's method, we take a brief digression to define the notion of the order of convergence of an iterative method.

Definition 8.1. Consider an iterative method that, given some initial guess x_0, yields a sequence $\{x_t\}_{t=0}^{\infty}$ satisfying

$$\|x_{t+1} - x^\star\| \leq M\|x_t - x^\star\|^q, \quad t = 0, 1, 2, \ldots \tag{8.1}$$

for some $x^\star$, a positive integer q, and $M > 0$. The **order of convergence** of the method is q, and the corresponding **rate of convergence** is M. The case $q = 1$ is called **linear convergence.** The case $q = 2$ is called **quadratic convergence**. More generally, the case $q > 1$ is called **superlinear convergence**.

Remark 8.2. Note that in the case $q = 1$, we need $M < 1$ for the inequality (8.1) to actually imply anything quantitatively about convergence, no matter how close the initial guess is to $x^\star$. Hence linear convergence as defined above is not interesting unless $M < 1$.

However, if $q > 1$, we *do not* need $M < 1$ for the definition to be interesting. As long as the initial guess is close enough to $x_\star$ (or x_t is eventually close to $x_\star$ for t sufficiently large), the inequality (8.1) itself will guarantee convergence.

In practice an iterative method may converge slowly at first, until it reaches a zone of superlinear convergence. If one is interested only in studying the asymptotic convergence behavior, it is therefore reasonable for theoretical purposes to assume that the initial guess is sufficiently close to the true solution. Then the following theorem explains why the inequality (8.1) implies extremely fast *local* convergence when $q > 1$. Note with caution that (8.1) *does not* in general imply anything about *global* convergence.

Theorem 8.3. *Suppose that an iterative method, with notation adopted from Definition 8.1, satisfies (8.1) with order $q > 1$ and rate $M > 0$. If $\|x_0 - x^\star\| \leq \frac{1}{M^{1/(q-1)}}$, then the method converges, and moreover there exist $\alpha, C > 0$ such that*

$$\|x_t - x^\star\| \leq Ce^{-\alpha q^t}.$$

Remark 8.4. In fact we can take $C = M^{-1/(q-1)}$, and $\alpha = -\log\left[M^{1/(q-1)}\|x^{(0)} - x^{(\star)}\|\right]$.

Remark 8.5. Notice the remarkable implication: the *logarithm* of the error goes to $-\infty$ exponentially fast as $t \to \infty$. Thus we conclude that with respect to an error tolerance $\varepsilon > 0$, the number of iterations required to reach the tolerance scales as $O(\log\log(1/\varepsilon))$, provided that we start in the local superlinear convergence regime.

Proof. Define $E_t := \|x_t - x^\star\|$ and plug in (8.1) recursively to deduce

$$\begin{aligned}
E_1 &\leq ME_0^q, \\
E_2 &\leq ME_1^q \leq M^{1+q}E_0^{q^2}, \\
E_3 &\leq ME_2^q \leq M^{1+q+q^2}E_0^{q^3}, \\
E_4 &\leq \cdots,
\end{aligned}$$

from which we conclude that

$$E_t \leq M^{\sum_{j=0}^{t-1} q^j} E_0^{q^t}$$

as can be verified by induction.

Taking logarithms, defining $e_t := \log E_t$, and summing the geometric series $\sum_{j=0}^{t-1} q^j = \frac{1}{q-1}[q^t - 1]$, we obtain

$$\begin{aligned}
e_t &\leq \frac{1}{q-1}[q^t - 1]\log M + q^t \log E_0 \\
&= [q^t - 1]\log\left(M^{1/(q-1)}\right) + q^t \log E_0 \\
&= -\alpha q^t + \log C,
\end{aligned}$$

where we define C and α following Remark 8.4.

We have convergence (i.e., $e_t \to -\infty$) if $\alpha > 0$, which holds when $\|x_0 - x^\star\| \leq \frac{1}{M^{1/(q-1)}}$. Thus the theorem is proved by re-exponentiating to get the desired bound. □

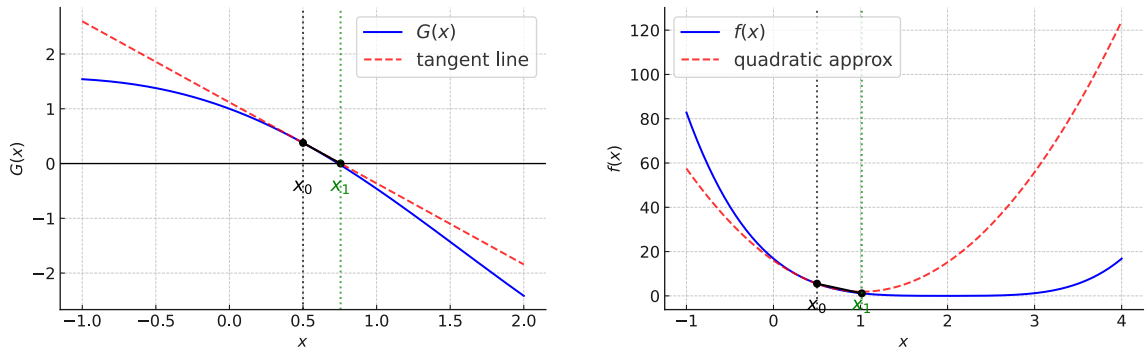

Figure 9.1: Two perspectives on the update in Newton's method. **Left:** for solving $G(x) = 0$, we solve the linearized equations. **Right:** for optimizing $f(x)$, we exactly minimize a quadratic approximation. (Note that the left and right panels do not correspond to one another here.)

9 Newton's method

Newton's method defines an approach for solving general systems of nonlinear equations with equal number of equations and unknowns:

$$G(x) = 0, \quad G : \mathbb{R}^n \to \mathbb{R}^n. \tag{9.1}$$

We can always write G in terms of *coordinate functions* $\mathbb{R}^n \to \mathbb{R}$:

$$G(x) = \begin{pmatrix} g_1(x) \\ \vdots \\ g_n(x) \end{pmatrix}, \quad g_1, \ldots, g_n : \mathbb{R}^n \to \mathbb{R}.$$

In the context of optimization of an unconstrained objective $f(x)$, we are interested in the choice

$$G(x) := \nabla f(x),$$

so that (9.1) defines the equations for a critical point. In the setting of differentiable strictly convex optimization, we know that solving these equations is equivalent to finding a unique optimizer. However, Newton's method can still be applied more broadly to nonconvex problems, and we will see that there is a local convergence theory guaranteeing convergence to critical points for a sufficiently close initial guess.

9.1 Derivation

Newton's method is based on the following iterative loop:

1. Linearize the equations near our current guess.
2. Solve linearized equations to obtain new guess.

More concretely, suppose that x_t is our current guess. Near x_t we can expand $G(x)$ to first order using multivariate Taylor expansion:

$$G(x) \approx G(x_t) + DG(x_t)\,(x - x_t), \tag{9.2}$$

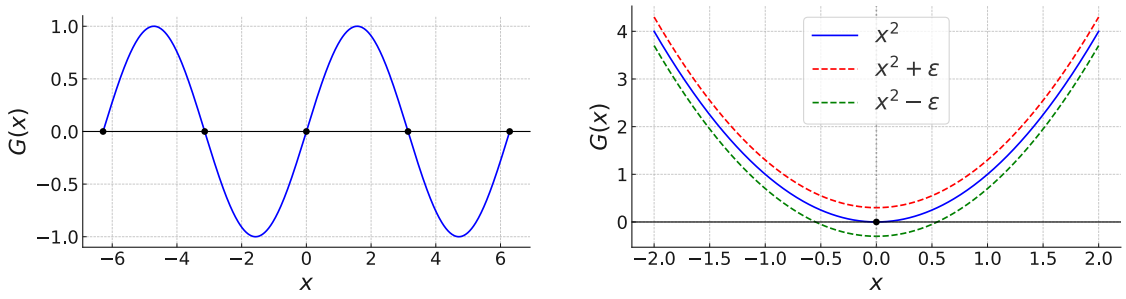

Figure 9.2: **Left:** A function $G(x)$ with many degenerate roots $x^\star$ satisfying $G'(x^\star) \neq 0$. **Right:** A function with a single degenerate root $x^\star$, for which $G'(x^\star) = 0$. The root can either vanish or bifurcate upon a small perturbation of G.

where

$$DG(x) := \left(\frac{\partial g_i}{\partial x_j}\right)_{i,j=1}^{n}$$

is the usual Jacobian matrix.

Then we replace G with the approximation (9.2) in the nonlinear equations $G(x) = 0$ to obtain the linearized equations

$$0 = G(x_t) + DG(x_t)\,(x - x_t),$$

which we can solve for x to obtain

$$x = x_t - DG(x_t)^{-1} G(x_t).$$

We set the solution of these equations to be the next guess $x^{(k+1)}$, thereby fixing the following iteration

$$x_{t+1} = x_t - DG(x_t)^{-1} G(x_t), \tag{9.3}$$

which is called ***Newton's method***. In practice we want to avoid literally inverting the matrix $DG(x_t)$, so we instead think of the process of computing $y = DG(x_t)^{-1} G(x_t)$ as solving the following linear system for y:

$$DG(x_t)\, y = G(x_t),$$

in which x_t is fixed.

In the optimization setting where $G(x) := \nabla f(x)$, we have $DG(x) = \nabla^2 f(x)$, so Newton's method reads as

$$x_{t+1} = x_t - \nabla^2 f(x_t)^{-1} \nabla f(x_t).$$

An equivalent derivation is to approximate f locally about x_t with a quadratic using a second-order Taylor series expansion, then solve for a critical point of this quadratic approximation.

Intuitively, since $DG(x_t)^{-1}$ arises in the method and we hope that $x_t \to x^\star$ as $t \to \infty$, we expect that there should be problems if $DG(x^\star)$ is not invertible. Our convergence theory will rule out this possibility by assumption, but let us dwell on the intuition.

When $DG(x^\star)$ is invertible, this means that the root $x^\star$ satisfying $G(x^\star) = 0$ is nondegenerate in the sense that a small perturbation of G will yield a small perturbation of the root location. By contrast, a degenerate root may vanish or bifurcate upon an infinitesimally small perturbation of G. See Figure 9.2 for an illustration in dimension $n = 1$, where the nondegeneracy condition simplifies to saying that $G'(x^\star) \neq 0$.

Moreover, note that if $DG(x^\star)$ is invertible, then $DG(x_t)$ will be invertible for x_t sufficiently close to $x^\star$, so the update (9.3) is well-defined in the local convergence regime.

A model pseudocode for the optimization context is provided in Algorithm 3. However, we warn that only local convergence is guaranteed. Below we will consider variants that are safer for global (and possibly nonconvex) unconstrained optimization.

Algorithm 3 Newton's method for optimization (*only local convergence guaranteed!*)

Input: $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$, $\nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{n \times n}$, initial guess $x_0 \in \mathbb{R}^n$, maximum number of iterations T, residual tolerance ε

Output: Estimates for optimal x and $f(x)$, convergence flag

```
Set x ← x_0 and flag ← false
for t = 1, ..., T do
    Set b ← ∇f(x), A ← ∇²f(x)
    Solve Ay = b for y ∈ ℝ^n
    Set x ← x − y
    if ‖b‖ ≤ ε then
        flag ← true
        break
    end if
end for
return x, f(x), flag
```

9.2 Local convergence theory

Now we study the local convergence of Newton's method. We must assume first of all that there exists some $x^\star$ satisfying $G(x^\star) = 0$.

We start with a formal argument before trying to make it rigorous. Define the error vector $e_t := x_t - x^\star$ and subtract both sides from the update rule (9.3) to deduce:

$$e_{t+1} = e_t - DG(x_t)^{-1} G(x_t). \tag{9.4}$$

Now, assuming sufficient smoothness (e.g., G twice continuously differentiable), consider the Taylor expansion

$$0 = G(x^\star) = G(x_t) + DG(x_t)\,(x_t - x^\star) + O(\|x_t - x^\star\|^2).$$

Then recall that $G(x^\star) = 0$, rearrange, and use the definition of e_t to write

$$G(x_t) = -DG(x_t)\, e_t + O(\|e_t\|^2).$$

Finally, substitute into (9.4) to deduce

$$e_{t+1} = O(\|e_t\|^2),$$

which is precisely indicative of *quadratic* convergence.

Our implicit assumption above that G is C^2 can be relaxed somewhat. We really only need DG to be Lipschitz in a certain sense, as we state presently.

Theorem 9.1 (Quadratic convergence of Newton's method)**.** *Suppose that $G : \mathbb{R}^n \to \mathbb{R}$ and $x^\star \in \mathbb{R}^n$ satisfies $G(x^\star) = 0$.*

Assume that the Jacobian $DG : \mathbb{R}^n \to \mathbb{R}^{n\times n}$ is B-Lipschitz for some $B \geq 0$ in the sense that

$$\|DG(x) - DG(y)\| \leq B\|x - y\| \tag{9.5}$$

for all $x, y \in \mathbb{R}^n$. (The vector norm can be arbitrary, and the matrix norm on the left-hand side is the corresponding operator norm.)

Moreover assume that there exists $\mu > 0$ such that the Jacobian satisfies

$$\|DG(x)^{-1}\| \leq \mu^{-1} \tag{9.6}$$

for all $x \in \mathbb{R}^n$.

Then the iterates of Newton's method (9.3) with arbitrary initial condition x_0 satisfy

$$\|x_{t+1} - x^\star\| \leq \frac{B}{2\mu}\|x_t - x^\star\|^2, \tag{9.7}$$

i.e., Newton's method is quadratically convergent with rate $B/(2\mu)$, in the sense of Definition 8.1.

Remark. Note that the nondegeneracy assumption, i.e., that $DG(x^\star)$ is invertible, is built into (9.6), which phrases this assumption more quantitatively. In the context of optimization where $G = \nabla f$, we know that $DG(x) = \nabla^2 f(x)$ is always invertible as long as f is strictly convex. More quantitatively, (9.6) follows precisely from μ-strong convexity in the case where $\|\cdot\| = \|\cdot\|_2$, since in this case $\|DG(x)^{-1}\|_2$ is the largest eigenvalue of the inverse of $\nabla^2 f(x)$, and μ-strong convexity means that all eigenvalues of $\nabla^2 f(x)$ are at least as large as μ.

Remark. If we want to show local convergence of Newton's method, the global quantitative assumptions in terms of B and μ can be relaxed to local assumptions on a neighborhood of $x^\star$. In that case, we must assume that the initial condition x_0 is sufficiently close to $x^\star$ in order for both (1) (9.7) to hold and (2) (9.7) to guarantee that $\|x_{t+1} - x^\star\| \leq \|x_t - x^\star\|$, ensuring that we never leave the neighborhood. As long as DG is locally B-Lipschitz (in particular, if G is C^2) and $DG(x^\star) \neq 0$, a suitable local convergence theorem can be stated and proved. For simplicity, we opt for the cleaner global statement.

Proof. Consider (9.4) as our starting point, where we recall $e_t := x_t - x^\star$.

Then fix t, define

$$x(\theta) := x^\star + \theta(x_t - x^\star) = x^\star + \theta e_t \tag{9.8}$$

for $\theta \in [0, 1]$, and then use the fundamental theorem of calculus to write

$$G(x_t) = G(x_t) - G(x^\star) = \int_0^1 \frac{d}{d\theta} G(x(\theta))\, d\theta = \int_0^1 DG(x(\theta))\, e_t\, d\theta,$$

where we have used the facts that $G(x^\star) = 0$ and $\frac{d}{d\theta}x(\theta) = e_t$.

Now substitute into (9.4) to deduce

$$\begin{aligned} e_{t+1} &= e_t - DG(x_t)^{-1} \int_0^1 DG(x(\theta))\, e_t \, d\theta \\ &= DG(x_t)^{-1} \left[DG(x_t)\, e_t - \int_0^1 DG(x(\theta))\, e_t \, d\theta \right] \\ &= DG(x_t)^{-1} \left[\int_0^1 \left[DG(x_t) - DG(x(\theta)) \right] e_t \, d\theta \right]. \end{aligned}$$

Then take the norm of both sides, using the submultiplicativity of the operator norm and the triangle inequality to deduce

$$\begin{aligned} \|e_{t+1}\| &\le \|DG(x_t)^{-1}\| \int_0^1 \|DG(x_t) - DG(x(\theta))\| \, \|e_t\| \, d\theta \\ &\le \frac{B}{\mu} \|e_t\| \int_0^1 \|x_t - x(\theta)\| \, d\theta, \end{aligned}$$

where in the last step we used (9.5) and (9.6).

Finally, compute from the definition (9.5):

$$x_t - x(\theta) = (1-\theta)(x_t - x^\star) = (1-\theta) e_t$$

from which it follows that

$$\|e_{t+1}\| \le \frac{B}{\mu} \|e_t\|^2 \int_0^1 (1-\theta) \, d\theta = \frac{B}{2\mu} \|e_t\|^2.$$

This concludes the proof. □

9.3 Trust-region Newton

We would like to use Newton's method as a global unconstrained optimizer, possibly for nonconvex objectives, but there are two dangers:

1. Uncontrolled behavior outside the local convergence domain.
2. Hessian may not be positive definite, meaning that $-\nabla^2 f(x_t)^{-1} \nabla f(x_t)$ may not even define a descent direction!

There is a very safe way to proceed using a **trust region**, which is a concept of more general use.

For a given base point $y \in \mathbb{R}^n$, define the quadratic expansion about y as:

$$Q_y(x) := f(y) + \nabla f(y) \cdot (x - y) + \frac{1}{2} (x-y)^\top \nabla^2 f(y) (x-y). \tag{9.9}$$

Now at iteration t, instead of optimizing Q_{x_t} (which may not even admit a global minimizer if $\nabla^2 f(y)$ is indefinite!), we optimize it over a so-called trust region, taken to be a ball of radius $\Delta_t > 0$ about x_t:

$$x_{t+1} = \operatorname*{argmin}_{x \in \mathbb{R}^n \,:\, \|x - x_t\| \le \Delta_t} \left\{ Q_{x_t}(x) \right\}. \tag{9.10}$$

The update (9.10) defines a method that we call **trust-region Newton** for short.

There are now two difficulties to address:

1. How to solve the optimization problem in (9.10)?
2. How to choose the trust region radius Δ_t?

Therefore (9.10) does not yet define a complete algorithm, and in fact we will modify it to reject updates that fail to decrease the objective due to an overly ambitious choice of Δ_t.

9.3.1 Solving the trust region subproblem

We address the first point first. First we reformulate (9.10) in terms of $s = x - x_t$ as the equivalent minimization problem

$$s_t := \operatorname*{argmin}_{s \in \mathbb{R}^n \,:\, \|s\| \le \Delta_t} \left\{ \nabla f(x_t) \cdot s + \frac{1}{2} s^\top \nabla^2 f(x_t) s \right\}, \tag{9.11}$$

in terms of which we recover

$$x_{t+1} = x_t + s_t.$$

For notational simplicity, it makes sense to consider the more abstract problem

$$s^\star(b, A, \Delta) := \operatorname*{argmin}_{s \in \mathbb{R}^n \,:\, \|s\| \le \Delta} \{\phi_{b,A}(s)\}, \quad \text{where } \phi_{b,A}(s) := b \cdot s + \frac{1}{2} s^\top A s, \tag{9.12}$$

defined in terms of arbitrary $b \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ symmetric, and $\Delta > 0$.

Theorem 9.2. *Let $b \in \mathbb{R}^n$, $A \in \mathbb{R}^{n \times n}$ symmetric, and $\Delta > 0$. Let $\lambda_{\min}$ denote the minimal eigenvalue of A. Then a solution $s^\star = s^\star(b, A, \Delta)$ of (9.12) can be recovered as follows.*

- **Case 1:** A is positive definite ($\lambda_{\min} > 0$). *If $\|A^{-1}b\| \le \Delta$, we take $s^\star = -A^{-1}b$. (This is the case where the trust region constraint is inactive.) Otherwise, there exists a unique $\lambda^\star > 0$ such that $\|(A + \lambda^\star I)^{-1} b\| = \Delta$, in terms of which we take $s^\star = -(A + \lambda^\star I)^{-1} b$.*
- **Case 2:** A is not positive definite and $b \notin \mathrm{col}(A - \lambda_{\min} I) = \mathrm{null}(A - \lambda_{\min} I)^\perp$ (which holds generically). *Then there exists a unique $\lambda^\star > -\lambda_{\min}$ such that $\|(A + \lambda^\star I)^{-1} b\| = \Delta$, in terms of which we take $s^\star = -(A + \lambda^\star I)^{-1} b$.*
- **Case 3:** A is not positive definite and $b \in \mathrm{null}(A - \lambda_{\min} I)^\perp$. *Then if $\|(A - \lambda_{\min})^+ b\| \le \Delta$, we take $s^\star = -(A - \lambda_{\min} I)^+ b + v$, where $v \in \mathrm{null}(A - \lambda_{\min} I)$ is arbitrary with $\|v\|^2 = \Delta^2 - \|(A - \lambda_{\min} I)^+ b\|^2$. Otherwise, there exists a unique $\lambda^\star > -\lambda_{\min}$ such that $\|(A + \lambda^\star I)^{-1} b\| = \Delta$, in terms of which we take $s^\star = -(A + \lambda^\star I)^{-1} b$.*

Remark 9.3. Here $(A - \lambda_{\min} I)^+$ denotes the Moore-Penrose pseudoinverse. The last case will not occur generically, but if either of the other two cases returns $\lambda \approx -\lambda_{\min}$, for numerical robustness it can make sense to snap $\lambda \to -\lambda_{\min}$ and then use the pseudoinverse. We have not yet discussed an algorithmic strategy for computing λ, which we will consider below.

We will prove the theorem later using the tools of Lagrange duality. However, for now we will give arguments that explain the computation of λ and why it is uniquely defined as claimed in the theorem.

First we simplify the problem by diagonalization. Note that large-scale trust-region Newton solvers will tend to avoid such a step and instead frame the algorithm only in terms of linear

solves, but for pedagogical clarity we view diagonalization as a preprocessing step. The downstream computations following this step will be significantly cheaper.

To wit, write $A = U\Lambda U^\top$ where U is an orthogonal matrix and $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$ with diagonal entries ordered non-decreasingly, so $\lambda_{\min} = \lambda_1$.

Then we can rewrite the objective

$$\begin{aligned}\phi_{b,A}(s) &= b^\top s + \frac{1}{2} s^\top A s \\ &= b^\top U U^\top s + (U^\top s)^\top \Lambda (U^\top s) \\ &= (U^\top b) \cdot (U^\top s) + (U^\top s)^\top \Lambda (U^\top s).\end{aligned}$$

Hence when optimizing over s of unit norm, we can change variables to $z = U^\top s$, which automatically satisfies $\|z\| = \|s\|$. We conclude that we can recover $s^\star = s^\star(b, A, \Delta)$ as

$$s^\star = U z^\star, \quad \text{where} \;\; z^\star := s^\star(c, \Lambda, \Delta), \quad c := U^\top b,$$

hence by diagonalization we have reduced the general problem (9.12) to the case where the matrix is diagonal. Note that we are abusing notation slightly by using $s^\star$ both to denote an optimal value as well as the function (9.12) of the problem parameters.

Then we can apply Theorem (9.2), replacing $b \leftarrow c$ and $A \leftarrow \Lambda$. Let us consider the cases individually.

- **Case 1:** A is positive definite ($\lambda_{\min} > 0$). If $\|\Lambda^{-1} c\| > \Delta$, we want to find $\lambda > 0$ such that

  $$\psi(\lambda) := \|(\Lambda + \lambda I)^{-1} c\|^2 = \sum_{i=1}^n \frac{c_i^2}{(\lambda + \lambda_i)^2} = \Delta^2.$$

  Note that since $\lambda_i > 0$ for all $i = 1, \ldots, n$, the objective $\psi(\lambda)$ is strictly decreasing with $\lim_{\lambda \to +\infty} \psi(\lambda) = 0$. And by assumption, $\psi(0) > \Delta^2$. Thus there exists a unique $\lambda^\star > 0$ satisfying $\psi(\lambda^\star) = \Delta^2$.

- **Case 2:** A is not positive definite and $c \notin \mathrm{col}(\Lambda - \lambda_{\min} I)$. Note that this means that there exists some j for which $\lambda_j = \lambda_{\min}$ and $c_j \neq 0$. This implies that $\psi(\lambda) \to +\infty$ as $\lambda \to -\lambda_{\min}$ from above. Moreover, $\psi(\lambda)$ is strictly decreasing on $(\lambda_{\min}, +\infty)$. Hence there exists a unique $\lambda^\star > -\lambda_{\min}$ satisfying $\psi(\lambda^\star) = \Delta^2$.

- **Case 3:** A is not positive definite and $c \in \mathrm{col}(\Lambda - \lambda_{\min} I)$. In this case we know $c_j = 0$ for all j such that $\lambda_j = \lambda_{\min}$. Therefore it is not necessarily the case that $\psi(\lambda) \to +\infty$ as $\lambda \to -\lambda_{\min}$ from above. However, if $\|(\Lambda - \lambda_{\min})^+ c\| > \Delta$, this means that $\psi(\lambda_{\min}) > \Delta^2$, and still there exists a unique $\lambda^\star > -\lambda_{\min}$ satisfying $\psi(\lambda^\star) = \Delta^2$.

Finally, we must discuss how to compute $\lambda^\star$ in each of the above cases, when it is relevant to do so. Note that this is a one-dimensional root-finding problem, and in each case we have a lower bound for the search interval. If we can compute an upper bound on the search interval, then we can proceed using the **bisection method** (see Remark 9.4 below). More ambitiously, we could use Newton's method for this root-finding problem! But we would still have to use the interval bounds as a safeguard, and we omit discussion of this possibility. (Relative to the upfront cost of a diagonalization, the cost of the bisection method is negligible. However, in more sophisticated approaches, a better root-finder will require fewer queries to a linear solver.)

To compute an upper bound on the search interval , observe that since $\lambda_i \geq \lambda_{\min}$, for $\lambda > -\lambda_{\min}$ we can upper-bound

$$\psi(\lambda) \leq \sum_{i=1}^n \frac{c_i^2}{(\lambda + \lambda_{\min})^2} = \frac{\|c\|^2}{(\lambda + \lambda_{\min})^2}.$$

Hence by setting $\psi(\lambda) = \Delta^2$ and solving for λ, we conclude that when

$$\lambda \geq \frac{\|c\|}{\Delta} - \lambda_{\min},$$

we have that $\psi(\lambda) \leq \Delta^2$.

Therefore in each case where we have to search for λ, we can search using the bisection method over

$$\lambda \in \left[-\lambda_{\min}, \frac{\|c\|}{\Delta} - \lambda_{\min}\right]$$

to find λ such that

$$\psi(\lambda) - \Delta^2 = 0.$$

Remark 9.4 (Bisection method). In general, given a decreasing function $h(\lambda)$ that we know possesses a root in the interval $[a, b]$, the bisection method is a simple approach to finding this root. Define the midpoint $c = (a + b)/2$ and test the function there. If $h(c) < 0$, we know that the root lies in $[a, c]$, and we replace $b \leftarrow c$. Otherwise $h(c) > 0$, we know that the root lies in $[b, c]$, and we replace $a \leftarrow c$. Either way, we now know that the root lies in $[a, b]$, and we can repeat the procedure. (If $h(c) = 0$ exactly, we can terminate.) The bounding interval shrinks by a factor of 2 with each test, hence for any $\varepsilon > 0$ after $\lceil \log_2(\frac{b-a}{\varepsilon}) \rceil$ iterations (where a, b denote the original endpoints), our interval of uncertainty has length ε. Then the final midpoint must be of distance at most $\varepsilon/2$ from the exact root.

9.3.2 Choosing the trust region radius

To complete our discussion of trust-region Newton, it remains to discuss the choice of the trust region radius Δ_t.

After computing s_t as defined in (9.11), following the procedure outlined in the preceding section, we compute the following key quantity:

$$\rho_t := \frac{f(x_t) - f(x_t + s_t)}{Q_{x_t}(x_t) - Q_{x_t}(x_t + s_t)} = \frac{f(x_t + s_t) - f(x_t)}{\nabla f(x_t) \cdot s_t + \frac{1}{2} s_t^\top \nabla^2 f(x_t)\, s_t} = \frac{\text{“actual reduction”}}{\text{“predicted reduction” } (\geq 0)}, \tag{9.13}$$

in which we recall the definition of Q_{x_t} (9.9).

If $\rho_t \gtrsim 1$ (concretely, if $\rho_t > \eta_2$ for a hyperparameter $1 \approx \eta_2 > 0$), this means that we have reduced the objective at least about as much as we expected based on the quadratic proxy. Our trust region is safe, and we might try increasing the radius—by a factor of $\gamma_\uparrow \in (0, 1)$—if the boundary constraint was activated, i.e., if $\|s_t\| = \Delta_t$. If $\rho_t < 0$, then the objective increased, which is bad! In this case we should reject the update and shrink the radius, by a factor of $\gamma_\downarrow \in (0, 1)$. However, even if ρ_t is positive and small (i.e., $\rho_t < \eta_0$, for a hyperparameter $\eta_0 \in (0, \eta_2)$), we conclude that our proxy is not accurate on the trust region, and we still reject the update and shrink the radius. For intermediate values $\rho \in [\eta_0, \eta_1)$, where $\eta_1 \in (\eta_0, \eta_2)$, we accept the update but still shrink the radius.

Reasonable choices for the hyperparameters are given by $\eta_0 = 0.1$, $\eta_1 = 0.25$, $\eta_2 = 0.75$, $\gamma_\downarrow = 0.25$, $\gamma_\uparrow = 2$. A complete pseudocode is given in Algorithm 4.

Algorithm 4 Trust-region Newton

Input: $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f : \mathbb{R}^n \to \mathbb{R}^n$, $\nabla^2 f : \mathbb{R}^n \to \mathbb{R}^{n \times n}$, initial guess $x_0 \in \mathbb{R}^n$, initial radius $\Delta_0 > 0$, maximum radius $\Delta_{\max} > 0$, threshold parameters $\eta_2 > \eta_1 > \eta_0 > 0$, radius contraction parameter $\gamma_\downarrow \in (0, 1)$, radius expansion parameter $\gamma_\uparrow > 1$, maximum number of iterations T, residual tolerance ε

Output: Estimates for optimal x and $f(x)$, convergence flag

Set $x \leftarrow x_0$, $\Delta \leftarrow \Delta_0$, and `flag` $\leftarrow$ `false`
for $t = 1, \ldots, T$ **do**
  Set $b \leftarrow \nabla f(x)$, $A \leftarrow \nabla^2 f(x)$ ▷ Store gradient and Hessian
  Compute $s = s^\star(b, A, \Delta)$ ▷ Solve trust region subproblem (9.12)
  Set $\rho = [f(x + s) - f(x)] \,/\, [b \cdot s + \frac{1}{2} s^\top A s]$ ▷ Compute reduction ratio, cf. (9.13)
  if $\rho > \eta_0$ **then**
    Set $x \leftarrow x + s$ ▷ Accept the update
  end if
  if $\rho < \eta_1$ **then**
    Set $\Delta \leftarrow \gamma_\downarrow \Delta$ ▷ Shrink the trust region radius
  else if $\rho > \eta_2$ and $\|s\| = \Delta$ **then**
    Set $\Delta \leftarrow \min\{\gamma_\uparrow \Delta, \Delta_{\max}\}$ ▷ Expand the trust region radius
  end if
  if $\|b\| \le \varepsilon$ **then**
    `flag` $\leftarrow$ `true`
    break
  end if
end for
return x, $f(x)$, `flag`

10 Quasi-Newton methods

Next we turn to the topic of **quasi-Newton methods**, which attempt to match the performance of Newton's method while avoiding the $O(n^3)$ cost of a generic Hessian solve. Specifically, we will focus on the **BFGS** (Broyden–Fletcher–Goldfarb–Shanno) **algorithm**, which attempts to maintain an approximation M_t of the inverse Hessian matrix that is updated online at each iteration t. (Often, the inverse Hessian approximation is denoted H_t in the literature, but this notation is arguably confusing.) Interestingly, in the BFGS algorithm, M_t is positive semidefinite by construction even when the underlying optimization problem is nonconvex! This will ensure that $p_t := -M_t \nabla f(x_t)$ always defines a valid descent direction. However, this key property depends on a line search condition that we must discuss first.

The total cost per iteration of BFGS is $O(n^2)$. There is a famous extension called limited-memory BFGS (**L-BFGS**) which achieves $O(n)$ cost per iteration but involves additional approximation heuristics. We will not discuss L-BFGS in this course.

10.1 Interlude: line search

Within each iteration t of an optimization algorithm, line search starts with a current guess $x_t \in \mathbb{R}^n$ and a search direction $p_t \in \mathbb{R}^n$. The procedure for determining p_t depends on the optimization algorithm, but once a choice is made, the line search seeks to determine $\alpha_t \geq 0$, in terms of which we update $x_{t+1} = x_t + \alpha_t p_t$.

Typically we demand that p_t defines a descent direction at x_t in the following sense.

Definition 10.1 (Descent direction)**.** Given a differentiable objective $f : \mathbb{R}^n \to \mathbb{R}$, we say that $p \in \mathbb{R}^n$ is a **descent direction** at $x \in \mathbb{R}^n$ if

$$p \cdot \nabla f(x) \leq 0.$$

We may say p is a strict descent direction if the inequality holds strictly.

Remark 10.2. Note that it is equivalent to require that $\frac{d}{d\alpha}\big|_{\alpha=0} f(x + \alpha p) \leq 0$.

Observe that the negative gradient $p = -\nabla f(x)$ always defines a descent direction at x. More generally, any $p = -A\nabla f(x)$ where A is positive semidefinite defines a descent direction because

$$p \cdot \nabla f(x) = -\nabla f(x)^\top A \nabla f(x) \leq 0.$$

Therefore if f is strictly convex and twice differentiable, the descent direction

$$p_t := -\nabla^2 f(x_t)^{-1} \nabla f(x_t)$$

can be used to define a suitable "Newton's method with line search." However, for a general objective, this choice does not define a descent direction! This is why we introduced trust-region Newton. Still, Newton's method with line search can be useful for convex optimization and in particular for interior point methods.

Our line search will seek to achieve the following conditions.

Definition 10.3 (Wolfe conditions)**.** Given a descent direction $p \in \mathbb{R}^n$ at $x \in \mathbb{R}^n$ for a continuously differentiable objective $f : \mathbb{R}^n \to \mathbb{R}$, we say that $\alpha \geq 0$ satisfies the **weak Wolfe conditions** (with hyperparameters $0 < c_1 < c_2 < 1$) if

$$\begin{aligned} f(x + \alpha p) &\leq f(x) + c_1 \alpha\, p \cdot \nabla f(x) && \text{(Armijo condition)} \\ -p \cdot \nabla f(x + \alpha p) &\leq -c_2\, p \cdot \nabla f(x) && \text{(Weak curvature condition)} \end{aligned}$$

If the weak curvature condition is replaced with the following stronger condition

$$|p \cdot \nabla f(x + \alpha p)| \leq -c_2\, p \cdot \nabla f(x) \qquad \text{(Strong curvature condition)}$$

we say that α satisfies the **strong Wolfe conditions**.

Remark 10.4. A typical choice of hyperparameters c_1, c_2 is given by $c_1 = 10^{-4}$, $c_2 = 0.9$. We will see that we need $c_1 < c_2$ in order to ensure that there exists $\alpha \geq 0$ satisfying the Wolfe conditions.

Remark 10.5. Note that the Armijo condition recovers our backtracking line search condition for gradient descent if we choose $p = -\nabla f(x)$ and $c_1 = \frac{1}{2}$. See Figure 10.1 for an illustration.

To motivate the curvature condition, consider the objective $\phi : \mathbb{R} \to \mathbb{R}$ defined by restricting f to the line defined by a search direction p at x:

$$\phi(\alpha) := f(x + \alpha p). \tag{10.1}$$

We will *not* ask for α that *exactly* minimizes ϕ, but note that there should exist such a point, which must satisfy $\phi'(\alpha) = 0$, i.e., $p \cdot \nabla f(x + \alpha p) = 0$. Therefore the curvature condition seeks to find α that is closer to satisfying this first-order optimality condition than the base point $\alpha = 0$. More concretely, it asks for

$$-\phi'(\alpha) \leq -c_2 \phi'(0) \quad \text{(Weak)}, \qquad |\phi'(\alpha)| \leq c_2 |\phi'(0)| \quad \text{(Strong)}$$

Theorem 10.6 (Wolfe interval existence)**.** *Suppose $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and bounded below. Suppose that $p \in \mathbb{R}^n$ is a descent direction at $x \in \mathbb{R}^n$ such that $\nabla f(x) \neq 0$. Then there exists $\alpha > 0$ satisfying the strong Wolfe conditions with hyperparameters $0 < c_1 < c_2 < 1$.*

Remark 10.7. We will even see that the conditions hold with strict inequality, hence by continuity there exists an interval on which they are satisfied.

Proof. Define $\psi : \mathbb{R} \to \mathbb{R}$ via

$$\begin{aligned} \psi(\alpha) &:= f(x + \alpha p) - [f(x) + c_1 \alpha\, p \cdot \nabla f(x)] \\ &= \phi(\alpha) - [\phi(0) + c_1 \phi'(0) \alpha]\,, \end{aligned} \tag{10.2}$$

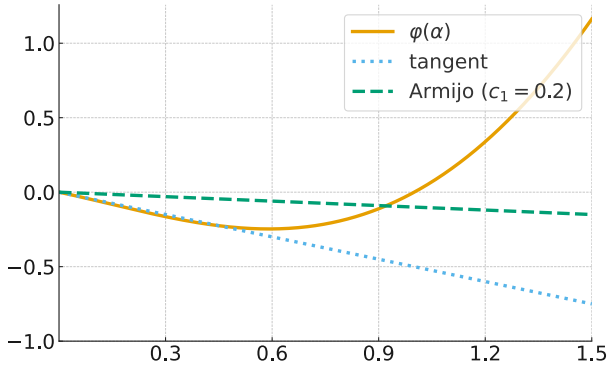

Figure 10.1: Depiction of the Armijo condition, which is satisfied where the curve of $\phi(\alpha) = f(x+\alpha p)$ lies below the dotted green line which is the graph of $\phi(0) + c_1\phi'(0)\alpha$.

where $\phi(\alpha) = f(x + \alpha p)$ as in (10.1). Note that ψ corresponds to the difference between the gold curve and the dotted green line in Figure 10.1. Moreover, $\psi < 0$ precisely where the Armijo condition holds strictly.

Now $\psi'(0) = (1 - c_1)\phi'(0)$, and $\phi'(0) = p \cdot \nabla f(x) < 0$ since p is a descent direction. Therefore since $c_1 \in (0, 1)$, we have that $\psi'(0)$. In turn there exists some $\delta > 0$ such that $\psi(\alpha) < 0$ for $\alpha \in (0, \delta)$.

Meanwhile we know that $\psi(\alpha) \to +\infty$ as $\alpha \to +\infty$ because f, and hence also ϕ, is bounded below and $\phi'(0) < 0$.

Therefore by the intermediate value theorem there exists $\tilde{\alpha} > 0$ such that $\psi(\tilde{\alpha}) = 0$ and more concretely we can consider the smallest such $\tilde{\alpha}$, defined by

$$\tilde{\alpha} := \inf\{\alpha > 0 : \psi(\alpha) = 0\}.$$

Then since $\psi < 0$ on $(0, \delta)$, we must indeed have $\tilde{\alpha} \geq \delta > 0$ and moreover by continuity $\psi(\tilde{\alpha}) = 0$, as claimed. Then by construction we know not only that $\psi(\tilde{\alpha}) = 0$ but also that $\psi < 0$ on $(0, \tilde{\alpha})$.

By Rolle's theorem, since $\psi(0) = \psi(\tilde{\alpha}) = 0$, we know that there exists some $\alpha^\star \in (0, \tilde{\alpha})$ such that $\psi'(\alpha^\star) = 0$, i.e., such that

$$-\phi'(\alpha) = \underbrace{-c_1\phi'(0)}_{>0} < -c_2\phi'(0),$$

(where we have used the fact that $c_1 < c_2$), meaning that the weak curvature condition holds strictly. Note that the middle expression is positive, so the strong curvature condition holds strictly as well.

Since $\alpha^\star \in (0, \tilde{\alpha})$, we know that $\psi(\alpha^\star) < 0$, i.e., the Armijo condition holds strictly. □

It remains to explain how to actually find α satisfying the Wolfe conditions. We will only concern ourselves with the weak case, which is all that is required by BFGS. The strong Wolfe conditions can be satisfied using the more complicated "bracket and zoom" algorithm (cf. Nocedal and Wright).

In our construction we will maintain the notations ψ (10.2) from the last proof. Recall that the Armijo condition for α is equivalent to $\psi(\alpha) \leq 0$.

We will maintain an interval (a, b) guaranteed to contain a weak Wolfe point, iteratively bisecting it while preserving this property. In particular, we will always **ensure that** a **satisfies Armijo**

and b fails Armijo. (Note that the considerations are more subtle than in the case of the standard bisection method for root finding.)

Indeed, note that if these properties hold, then in particular $\psi(b) > 0$ and $\psi(a) \leq 0$, so $\psi(b) - \psi(a) > 0$, and by the mean value theorem we know that there exists $\alpha \in (a, b)$ such that $\psi'(\alpha) > 0$, i.e., $\phi'(\alpha) \geq c_1\phi'(0)$, which implies that $-\phi'(\alpha) \leq -c_2\phi'(0)$, i.e., that α satisfies weak curvature.

Then the algorithm proceeds in two phases.

- Starting from an initial guess for α, keep inflating α by a factor of 2 until it fails Armijo.
  - Until α fails Armijo, we may as well check whether α passes weak curvature, in which case we can simply terminate and **return** α.
  - If we failed on the first try, simply set $a = 0$, $b = \alpha$. Then a passes Armijo trivially and b fails by construction.
  - Otherwise, set $a = \alpha/2$, $b = \alpha$, since we know by construction that $\alpha/2$ passes Armijo.
- Repeat the following. (Note that by construction, a will always pass Armijo, while b will always fail Armijo.)
  - Set $\alpha = (a + b)/2$ and check whether α fails Armijo, in which case we set $b = \alpha$ and **break**.
  - Otherwise α passes Armijo, so check whether it passes weak curvature. If so we can terminate and **return** α.
  - Otherwise, set $a = \alpha$.

A more formal pseudocode is given in Algorithm 5. We comment that a reasonable initial guess for α may be given by 1, if p is a Newton or quasi-Newton step. Note that each check of the Armijo and curvature conditions at some point α requires an evaluation of both $f(x + \alpha p)$ and $\nabla f(x + \alpha p)$.

10.2 BFGS

In the BFGS algorithm, we recursively update a matrix M_t, which can be viewed as an approximation of the inverse Hessian (at least asymptotically as $t \to \infty$ and x_t converges to a local minimizer). Soft convergence results (beyond the scope of this course) guarantee superlinear convergence under fairly generic conditions, though quantitative bounds are difficult to obtain.

In terms of M_t, at iteration t we define a search direction

$$p_t := -M_t \nabla f(x_t).$$

Remarkably, M_t will always be positive semidefinite even for a nonconvex objective f. This property can be maintained recursively provided that we choose the step size α_t using a weak Wolfe line search (cf. Algorithm 5). Typically we initialize $M_0 = I$ or $M_0 = \gamma I$ where $\gamma > 0$, depending on whether we have any *a priori* knowledge about the scale of the Hessian, so M_0 is positive definite in the base case. Under the inductive hypothesis that M_t is positive semidefinite, p_t defines a descent direction, so the line search is well-posed. Then in terms of α_t, we define the update

$$x_{t+1} = x_t + \alpha_t p_t.$$

To complete the specification of the BFGS algorithm, we must explain how to update M_t iteratively, i.e., how to determine M_{t+1} after x_{t+1} is computed.

The motivation for the update rule comes from the Taylor series expansion of the gradient,

$$\nabla f(x_t) \approx \nabla f(x_{t+1}) + \nabla^2 f(x_{t+1})\,(x_t - x_{t+1}),$$

Algorithm 5 Weak Wolfe line search

Input: $f : \mathbb{R}^n \to \mathbb{R}$ continuously differentiable, $x \in \mathbb{R}^n$ not a critical point, descent direction $p \in \mathbb{R}^n$, initial step size guess $\alpha^{(0)} > 0$, $0 < c_1 < c_2 < 1$

Output: $\alpha > 0$ satisfying weak Wolfe conditions

```
Set a ← 0, α ← α^(0)
while α satisfies Armijo(c_1) do
    if α satisfies WeakCurvature(c_2) then
        return α
    end if
    Set a ← α
    Set α ← 2α
end while
Set b ← α
while true do
    Set α ← (a + b)/2
    if α fails Armijo(c_1) then
        Set b ← α
    else if α fails WeakCurvature(c_2) then
        Set a ← α
    else
        return α
    end if
end while
```

which holds accurately if $x_{t+1} \approx x_t$, i.e., asymptotically as we make smaller updates.

Defining

$$s_t := x_{t+1} - x_t = \alpha_t p_t, \quad y_t := \nabla f(x_{t+1}) - \nabla f(x_t),$$

we can rearrange to obtain

$$y_t \approx \nabla^2 f(x_{t+1})\, s_t.$$

In the one-dimensional case $n = 1$, there is a unique choice of scalar B_{t+1} such that

$$y_t = B_{t+1}\, s_t,$$

and this B_{t+1} is the slope of the secant line for the graph of f' connecting the points $(x_t, f'(x_t))$ and $(x_{t+1}, f'(x_{t+1}))$.

Defining $M_{t+1} = (B_{t+1})^{-1}$, we can equivalently ask for

$$M_{t+1} y_t = s_t, \tag{10.3}$$

which is satisfied in the scalar case given s_t and y_t by taking M_{t+1} to be the inverse of the secant slope. When $x_{t+1} \approx x_t$, we have that $M_{t+1} \approx 1/f''(x_{t+1})$.

More generally, for any dimension $n \geq 1$, (10.3) is called the **secant equation**. Given s_t and y_t, the set of matrices M_{t+1} satisfying (10.3) consists of infinitely many choices. In the BFGS algorithm, M_{t+1} can be viewed as the *minimal* update of M_t subject to the secant equation, in the following sense

$$M_{t+1} = \operatorname*{argmin}_{M \in \mathbb{R}^{n\times n} \text{ symmetric}} \left\{ \|M - M_t\| \,:\, M y_t = s_t \right\},$$

where $\|\cdot\|$ denotes a particular scaled norm that we will not discuss further (cf. Nocedal and Wright). Over many iterations, we expect that M_t becomes a close approximation of $[\nabla^2 f(x_t)]^{-1}$.

Instead of considering the variational characterization, we will directly consider the formula for the update, which is itself intuitive:

$$M_{t+1} := \left(I - \frac{s_t y_t^\top}{\delta_t}\right) M_t \left(I - \frac{y_t s_t^\top}{\delta_t}\right) + \frac{s_t s_t^\top}{\delta_t}, \quad \delta_t := s_t^\top y_t. \tag{10.4}$$

We can directly verify from the construction that $M_{t+1} y_t = s_t$ (*Exercise.*) Moreover, M_{t+1} is symmetric by construction, provided that M_t is symmetric.

Finally, observe that if $\delta_t > 0$, then M_{t+1} is in fact positive semidefinite. Indeed, in this case the term $s_t s_t^\top / \delta_t$ is automatically positive semidefinite. Moreover, the first term in (10.4) is of the form $P_t^\top M_t P_t$, hence always positive semidefinite. (*Why?*)

Then to complete the story, we now show that indeed $\delta_t > 0$, provided that $\alpha_t > 0$ was chosen to satisfy the weak Wolfe conditions. Note that $\delta_t = s_t \cdot y_t$ and $s_t = x_{t+1} - x_t = \alpha_t p_t$, hence it is equivalent to show that $p_t \cdot y_t > 0$, i.e., that $p_t \cdot \nabla f(x_{t+1}) - p_t \cdot \nabla f(x_t) > 0$. In turn we want to show that

$$p_t \cdot \nabla f(x_t + \alpha_t p_t) > p_t \cdot \nabla f(x_t).$$

But the weak curvature condition (Definition 10.3) implies the stronger condition that $p \cdot \nabla f(x + \alpha p) \geq c_2\, p \cdot \nabla f(x)$, where $c_2 \in (0, 1)$. This completes the argument.

Finally, observe (*exercise*) that we can expand the right-hand side of (10.4) to deduce

$$M_{t+1} = M_t + \left(1 + \frac{y_t^\top M_t y_t}{\delta_t}\right) \frac{s_t s_t^\top}{\delta_t} - \frac{s_t (M_t y_t)^\top + (M_t y_t) s_t^\top}{\delta_t}.$$

Importantly, note that the cost of evaluating the right-hand side directly is only $O(n^2)$.

A full pseudocode of the BFGS algorithm is offered in Algorithm 6. For simplicity we omit stopping criteria based on a maximum number of iterations or a residual tolerance, but the user should specify such criteria by analogy to previous algorithms. We also hard-code the initial condition $M_0 = I$, but this choice might be modified.

Algorithm 6 BFGS algorithm

Input: $f : \mathbb{R}^n \to \mathbb{R}$ continuously differentiable, $x_0 \in \mathbb{R}^n$, Wolfe parameters $0 < c_1 < c_2 < 1$

Set $M \leftarrow I$
Set $v \leftarrow \nabla f(x_0)$
for $t = 0, 1, 2, \ldots$ **do**
  Set $p \leftarrow -Mv$
  Choose $\alpha > 0$ satisfying weak Wolfe conditions (using, e.g., Alg. 5 with initial guess $\alpha = 1$)
  Set $s \leftarrow \alpha p$, then $x \leftarrow x + s$
  Set $v^+ \leftarrow \nabla f(x)$, then $y \leftarrow v^+ - v$
  Set $\delta \leftarrow s \cdot y$
  Set $M \leftarrow M + \left(1 + \frac{y^\top My}{\delta}\right) \frac{ss^\top}{\delta} - \frac{s(My)^\top + (My)s^\top}{\delta}$
  Set $v \leftarrow v^+$
end for

11 Overview of interior point methods

In this section we will explore how Newton-type methods can be used to solve inequality-constrained optimization problems. Consider such a problem in general:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m. \end{aligned} \tag{11.1}$$

Note that a problem of this form might be produced by eliminating some equality constraints in some original problem of interest, cf. Section 2.4.4.

We will assume that our problem now admits an **interior point**, i.e., there exists $x \in \mathbb{R}^n$ satisfying $g_i(x) < 0$ for all $i = 1, \ldots, m$. Letting C denote the feasible set

$$C := \{x \in \mathbb{R}^n \,:\, g_i(x) \leq 0, \quad i = 1, \ldots, m\}$$

with interior

$$\text{int}(C) = \{x \in \mathbb{R}^n \,:\, g_i(x) < 0, \quad i = 1, \ldots, m\},$$

it is equivalent to assume that there exists $x \in \text{int}(C)$.

11.1 Blueprint

To solve the problem (11.1), we will introduce a **barrier function**

$$\Phi(x) := -\sum_{i=1}^{m} \log(-g_i(x)) \tag{11.2}$$

defined in terms of the constraints. See Figure 2.17 for an earlier illustration. Note that the barrier function is only defined on $\text{int}(C)$ and tends to $+\infty$ as we approach the boundary of C from the interior.

Lemma 11.1. *The barrier function (11.2) is convex on* $\text{int}(C)$ *if the constraint functions* g_i *are convex.*

Proof. The proof follows from Exercise 2.46, since the function $u \mapsto -u\log(-u)$ is convex and increasing on $(-\infty, 0)$. □

Then we use the barrier to define, for any $\mu > 0$, the **barrier-augmented objective**

$$f_\mu(x) = f(x) + \mu\Phi(x),$$

which in turn determines the **barrier subproblem**

$$\underset{x\in\mathrm{int}(C)}{\text{minimize}} \quad \{f_\mu(x)\}\,. \tag{11.3}$$

Corollary 11.2. *If (11.1) is a convex optimization problem, then the barrier subproblem (11.3) is a convex optimization problem.*

Proof. This follows directly from the last lemma. □

The idea of interior point methods is to recover an optimal solution $x^\star$ for the original problem (11.1) as a limit $x^\star = \lim_{\mu\to 0^+} x_\mu$, where x_μ is the optimal solution of the barrier subproblem (11.3). The map $\mu \mapsto x_\mu$ is called the **central path** and can tend toward a boundary point in the limit $\mu \to 0^+$ where the barrier is relaxed to zero.

To realize this idea in practice, we must choose a sequence of barrier parameters $\mu = \mu_j$, $j = 0, 1, 2, \ldots$, for which to solve the barrier subproblems (11.3) as well as an optimization algorithm to be applied for each j. We will choose $\mu_{j+1} = \sigma\mu_j$ where $\sigma \in (0, 1)$, e.g., $\sigma = 0.2$, so that the barrier parameter decays exponentially. Importantly, the solution of the j-th subproblem can be used as an initial condition for the $(j + 1)$-th subproblem, which can ensure that a Newton-type solver for all problems $j \geq 1$ is guaranteed to start in the local quadratic convergence regime and converge within only a few iterations.

A delicate convergence theory of interior point methods was pioneered by Nesterov and Nemirovskii, via analysis of Newton's method under the technical condition *self-concordance*, which applies for example in the case of convex optimization problems with quadratic objectives and linear inequality constraints.

We will not delve into the theory but instead just provide a practical recipe using concepts we have already reviewed. See Algorithm 7 for a practical pseudocode.

Algorithm 7 Interior point method

Input: Interior point $x_0 \in \mathrm{int}(C)$ for problem (11.1), convergence tolerance $\varepsilon > 0$ (e.g., $\varepsilon = 10^{-6}$), hyperparameters $\sigma \in (0, 1)$ and $\eta \in (0, 1)$ (e.g., $\sigma = 0.2$ and $\eta = 0.1$)

Set $x \leftarrow x_0$, $\mu \leftarrow \sigma^{-1}$

while $\mu > \varepsilon$ **do**

Set $\mu \leftarrow \sigma\mu$

Use x as an initial guess to solve barrier subproblem (11.3) approximately, furnishing $x \in \mathrm{int}(C)$ with $\|\nabla f_\mu(x)\| \leq \eta\varepsilon$

end while

Note that Algorithm 7 leaves two questions open. (1) How to solve the barrier subproblems given an interior point? (2) How to produce an interior point at the very beginning to initialize the interior point method? We address these questions now in order.

11.2 Solving the barrier subproblems

In the convex case, where f_μ is convex, typically Newton's method is used with suitable modifications. Specifically, given an initial guess in $\text{int}(C)$, we want to ensure that the method never leaves $\text{int}(C)$.

Since the Hessian is positive definite by strict convexity of the barrier, given a fixed barrier parameter $\mu > 0$ and some iterate $x \in \text{int}(C)$, we can use Newton's method to define a search direction $p = -\nabla^2 f_\mu(x)^{-1} \nabla f_\mu(x)$ and then use **backtracking line search** to find $\alpha > 0$ ensuring that

$$x + \alpha p \in \text{int}(C), \quad f_\mu(x + \alpha p) \leq f_\mu(x) + c\alpha\, p \cdot \nabla f_\mu(x),$$

i.e., that we do not leave the interior of the feasible set and that Armijo's condition holds for the objective f_μ. Since we do not impose a curvature condition, it is typical to choose $c = 0.1$ or 0.01. (Note that we can check membership $y \in \text{int}(C)$ for arbitrary y by simply evaluating $g_i(y)$ for all $i = 1, \ldots, m$.) As the initial guess for α within backtracking line search, we should choose $\alpha = 1$ so that we default to exact Newton updates when no backtracking is required.

In the nonconvex case, the philosophy of interior point methods is still quite relevant, but there is less of a framework of standard theory and practice. Nonetheless, a perfectly reasonable strategy is available to use using the ideas we have already discussed in the course. For the barrier subproblems, we can apply a trust-region Newton method such as Algorithm 4, with the suitable modification that the trust region radius must shrink if the proposed update $x + s$ lies outside of $\text{int}(C)$.

11.3 Finding an interior point

Now it remains only to discuss how to produce an interior point $x \in \text{int}(C)$, which is a nontrivial task. The process for producing such a point is often known as 'Phase I' of an interior point method. The remaining part, which we have already described, is then called Phase II.

To produce an interior point, we actually formulate an auxiliary **Phase-I problem** that we solve with an interior point method! However, the Phase-I problem admits an obvious interior point.

Indeed, the Phase-I problem is written:

$$\begin{aligned} \underset{x \in \mathbb{R}^n,\, t \in \mathbb{R}}{\text{minimize}} \quad & t \\ \text{subject to} \quad & g_i(x) \leq t, \quad i = 1, \ldots, m. \end{aligned} \tag{11.4}$$

Note that we can produce an interior point $(x, t) \in \mathbb{R}^{n+1}$ for the Phase-I problem (11.4) by taking ***any*** $x \in \mathbb{R}^n$ and letting $t > \max_{i=1,\ldots,m}\{g_i(x)\}$. Concretely, we could set $x = 0$ and $t = 1 + \max_{i=1,\ldots,m}\{g_i(0)\}$, though better initial guesses might be available. Then (11.4) can be solved using an interior point method following Algorithm 7.

Note that if the *original* problem (11.1) admits an interior point $x \in \text{int}(C)$, this means precisely that $t := \max_{i=1,\ldots,m}\{g_i(x)\} < 0$, so the optimal value of the Phase-I problem must be strictly negative.

Conversely, if we can find any feasible point (x, t) for the Phase-I problem (11.4) satisfying $t < 0$, then x is an interior point for the original problem (11.1).

Therefore, the optimizer of the Phase-I problem furnishes an interior point for the original problem. However, once our algorithm for solving the Phase-I problem furnishes an iterate (x, t) such that $t < 0$, we can terminate Phase I and proceed directly to Phase II, using x as our initial interior point for the original problem.

Part IV
Duality and primal-dual methods

In the following we consider the following rather general constrained optimization problem:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g_i(x) \leq 0, \quad i = 1, \ldots, m, \\ & h_j(x) = 0, \quad j = 1, \ldots, p. \end{aligned} \tag{11.5}$$

To express this problem more compactly, we may view the g_i as coordinate functions for a vector-valued $g : \mathbb{R}^n \to \mathbb{R}^m$ and h_j as the coordinate functions for a vector-valued $h : \mathbb{R}^n \to \mathbb{R}^p$, and then write:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0, \\ & h(x) = 0. \end{aligned}$$

We are often interested in the case where the problem is convex (i.e., f and all the g_i are convex, and all the h_j are affine-linear), but some of the following developments are relevant even for nonconvex problems.

In the context of duality, we will refer to (11.5) as the **primal problem**, in order to distinguish it from the associated dual problem which we will define later.

Sometimes it is useful to view a constrained problem as an equivalent unconstrained problem

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad \tilde{f}(x),$$

in which we extend the objective function to take the value $+\infty$ outside of the feasible set:

$$\tilde{f}(x) = \begin{cases} f(x), & \text{if } x \text{ is feasible, i.e., } g(x) \leq 0,\ h(x) = 0 \\ +\infty, & \text{otherwise.} \end{cases}$$

We will always assume that the primal problem is feasible, i.e., that there exists $x \in \mathbb{R}^n$ such that $\tilde{f}(x) < +\infty$ and let $p^\star < +\infty$ denote the optimal value.

Throughout the following we will carry forward all the aforementioned notation without comment.

12 Lagrangian duality

12.1 The Lagrangian

Definition 12.1 (Lagrangian)**.** We associate to (11.5) a function called the **Lagrangian** $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m_+ \times \mathbb{R}^p \to \mathbb{R}$, defined by

$$\begin{aligned}\mathcal{L}(x;\lambda,\nu) &:= f(x) + \sum_{i=1}^m \lambda_i g_i(x) + \sum_{j=1}^p \nu_j h_j(x) \\ &= f(x) + \lambda \cdot g(x) + \nu \cdot h(x).\end{aligned}$$

The arguments λ, ν are called **dual variables** (or **Lagrange multipliers**), by contrast with the **primal variables** x, and we use a semicolon to emphasize the distinction. Note that the domain constrains the dual variable λ to lie in $\mathbb{R}^m_+$, i.e., to satisfy $\lambda \geq 0$. If there are no inequality constraints (respectively, no equality constraints), then the λ argument (respectively, the ν argument) is omitted.

Remark 12.2. Note with caution that equivalent formulations of the same problem may induce different Lagrangians (and downstream, different dual problems).

The key motivation for the Lagrangian is the following fundamental lemma.

Lemma 12.3. The Lagrangian satisfies

$$\sup_{\lambda \in \mathbb{R}^m_+, \nu \in \mathbb{R}^p} \mathcal{L}(x;\lambda,\nu) = \tilde{f}(x) = \begin{cases} f(x), & \text{if } x \text{ is feasible} \\ +\infty, & \text{otherwise.} \end{cases}$$

Proof. Fix $x \in \mathbb{R}^n$. Note that if $h_j(x) \neq 0$ for some j, then we can make $\mathcal{L}(x;\lambda,\nu)$ arbitrarily large by taking $\nu_j = \operatorname{sign}(h_j(x))\,\alpha$ and sending $\alpha \to +\infty$ (setting all remaining dual variables to zero, for concreteness).

Meanwhile, if $g_i(x) > 0$ for some i, then we can make $\mathcal{L}(x;\lambda,\nu)$ arbitrarily large by sending $\lambda_i \to +\infty$ (setting all remaining dual variables to zero, for concreteness). This completes the proof. □

12.2 The dual problem

The preceding lemma establishes that we can write the primal optimal value as

$$p^\star = \inf_{x \in \mathbb{R}^n} \left\{ \sup_{\lambda \in \mathbb{R}^m_+, \nu \in \mathbb{R}^p} \mathcal{L}(x;\lambda,\nu) \right\} = \inf_{x \in \mathbb{R}^n} \left\{ \tilde{f}(x) \right\}. \tag{12.1}$$

Motivated by a desired to exchange the infimum and supremum, we make the following definition.

Definition 12.4 (Dual problem)**.** Define

$$d^\star := \sup_{\lambda \in \mathbb{R}^m_+,\, \nu \in \mathbb{R}^p} \left\{ \inf_{x \in \mathbb{R}^n} \mathcal{L}(x; \lambda, \nu) \right\} = \sup_{\lambda \in \mathbb{R}^m_+,\, \nu \in \mathbb{R}^p} \{q(\lambda, \nu)\}, \tag{12.2}$$

where in turn

$$q(\lambda, \nu) := \inf_{x \in \mathbb{R}^n} \mathcal{L}(x; \lambda, \nu).$$

The *maximization* problem defining $d^\star$ is called the **dual problem** associated to (11.5), in which q is the **dual objective**.

Remark 12.5. We will see that it is very much possible that $q(\lambda, \nu) = -\infty$ for some $(\lambda, \nu) \in \mathbb{R}^m_+ \times \mathbb{R}^p$. In this sense the dual problem can often be viewed as a constrained problem. The details must be worked out case by case.

Even if q attains the value $-\infty$, in fact q is ***always concave*** on $\mathbb{R}^m_+ \times \mathbb{R}^p$. The reason is that $-q(\lambda, \nu) = \sup_{x \in \mathbb{R}^n} \{-\mathcal{L}(x; \lambda, \nu)\}$ can be viewed as pointwise supremum of linear (hence convex) functions. It follows that $-q$ is convex by a slight generalization of Exercise 2.14.

Before discussing conditions under which the infimum and supremum can be exchanged, we work out an example illustrating how to derive a dual problem.

Example 12.6 (Dual linear program)**.** Consider a linear program of the form

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & c\cdot x \\ \text{subject to} \quad & x \geq 0, \\ & Ax = b. \end{aligned} \tag{12.3}$$

This is one of several ***standard forms*** for a linear program. We will explain in the remark below how to reduce any linear program to this form. However, note with caution that different equivalent forms for the primal linear program may yield different dual problems!

Before defining the Lagrangian, we must interpret the constraints carefully by defining $g(x) = -x$ and $h(x) = b - Ax$. Note that defining $h(x) = Ax - b$ defines an equivalent primal problem but will yield a slightly different dual problem (in which the associated dual variables are negated).

Then the Lagrangian is

$$\begin{aligned} \mathcal{L}(x;\lambda,\nu) &= c\cdot x - \lambda\cdot x + \nu\cdot(b - Ax) \\ &= (c - \lambda - A^\top\nu)\cdot x + b\cdot\nu. \end{aligned}$$

Note that

$$\inf_{x\in\mathbb{R}^n}\left\{(c-\lambda-A^\top\nu)\cdot x\right\} = \begin{cases} 0, & c-\lambda-A^\top\nu = 0, \\ -\infty, & \text{otherwise}, \end{cases}$$

from which it follows that

$$q(\lambda,\nu) = \inf_{x\in\mathbb{R}^n}\mathcal{L}(x;\lambda,\nu) = \begin{cases} b\cdot\nu, & c-\lambda-A^\top\nu = 0, \\ -\infty, & \text{otherwise}. \end{cases}$$

Hence the dual problem $\sup_{\lambda\in\mathbb{R}^m_+,\,\nu\in\mathbb{R}^p}\{q(\lambda,\nu)\}$ can be viewed as a constrained maximization problem:

$$\begin{aligned} \underset{\lambda\in\mathbb{R}^n,\,\nu\in\mathbb{R}^p}{\text{maximize}} \quad & b\cdot\nu \\ \text{subject to} \quad & \lambda \geq 0, \\ & A^\top\nu = c - \lambda. \end{aligned}$$

We can eliminate the λ variable to obtain the equivalent problem

$$\begin{aligned} \underset{\nu\in\mathbb{R}^p}{\text{maximize}} \quad & b\cdot\nu \\ \text{subject to} \quad & A^\top\nu \leq c. \end{aligned}$$

Often it is this simplified version that we refer to as the 'dual problem.'

Remark 12.7 (Reducing to standard form LP). Consider a more general linear progam

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & c\cdot x \\ \text{subject to} \quad & Cx \le d, \\ & Ax = b. \end{aligned} \tag{12.4}$$

Note that the letters c, A, b here *do not* correspond to the appropriate c, A, b that we derive after reducing from (12.4) to an equivalent problem of the form (12.3).

To reduce to the standard form, we note that we can always rewrite $x = y - z$ in terms of auxiliary nonnegative variables $y, z \ge 0$. Then we can eliminate x by substituting $y - z$ in place of x. Moreover, we introduce a nonnegative ***slack variable*** $s \ge 0$ satisfying $s = d - Cx = d - Cy + Cz$. Thus we define an equivalent problem:

$$\begin{aligned} \underset{y,z\in\mathbb{R}^n,\, s\in\mathbb{R}^m}{\text{minimize}} \quad & (c, -c, 0)\cdot(y, z, s) \\ \text{subject to} \quad & (y, z, s) \ge 0, \\ & s + Cy - Cz = d, \\ & Ay - Az = b. \end{aligned}$$

This can be viewed as a problem of the form (12.3) over the optimization variable $(y, z, s) \in \mathbb{R}^{2n+m}$ since the inequality constraints are simply entrywise nonnegativity constraints, and the remaining constraints are affine-linear equality constraints.

12.3 Weak and strong duality

We hope that the infimum and supremum can be exchanged in (12.1) and (12.2) so that $p^\star = d^\star$. Often (arguably, generically for convex problems) this is the case, but we *always* have $p^\star \ge d^\star$. This inequality is called **weak duality**. We say that the problem satisfies **strong duality** if $p^\star = d^\star$.

Lemma 12.8 (Weak duality)**.** $p^\star \ge d^\star$.

Proof. For simplicity, let $x^\star$ be a primal optimizer, so

$$p^\star = f(x^\star) = \tilde{f}(x^\star) = \sup_{\lambda\in\mathbb{R}^m_+,\, \nu\in\mathbb{R}^p} \mathcal{L}(x^\star; \lambda, \nu). \tag{12.5}$$

(More generally, if the primal minimum is not attained, we can consider an infimizing sequence.)

Now for all $\lambda \in \mathbb{R}^m_+$ and $\nu \in \mathbb{R}^p$, we have

$$\mathcal{L}(x^\star, \lambda, \nu) \ge \inf_{x\in\mathbb{R}^n} \mathcal{L}(x, \lambda, \nu),$$

and substituting this inequality into (12.5) we deduce

$$p^\star \ge \sup_{\lambda\in\mathbb{R}^m_+,\, \nu\in\mathbb{R}^p} \ge \inf_{x\in\mathbb{R}^n} \mathcal{L}(x, \lambda, \nu) = d^\star,$$

as was to be shown. □

You should think of strong duality as usually holding for convex problems, but it is difficult to formulate necessary and sufficient conditions. Nonetheless, the following sufficient conditions are sufficiently powerful to cover many cases of interest.

Theorem 12.9 (Sion's minimax theorem)**.** *Suppose that the primal problem is convex. If the primal feasible set is compact, then strong duality holds, and moreover the primal minimum is attained.*

Remark 12.10. This is really just a special case of a much more general statement of Sion's minimax theorem, which concerns convex-concave minimax problems.

Theorem 12.11 (Slater's condition)**.** *Suppose that the primal problem is convex (so that in particular all equality constraints are linear) and that there exists a feasible point at which all the inequality constraints all hold strictly, i.e., a feasible point x such that $g_i(x) < 0$ for all $i = 1, \dots, m$. Then strong duality holds, and moreover the dual maximum is attained.*

Remark 12.12. This is also a special case of a slightly more general result called Slater's condition. Roughly speaking, the condition amounts to saying that after eliminating equality constraints, the feasible set has a nonempty interior.

The proofs of both theorems are quite involved and beyond the scope of this course.

We will now focus on the case where strong duality holds and there exists a primal-dual optimizer, in the following sense.

Definition 12.13 (Primal-dual optimizer)**.** We say that $(x^\star; \lambda^\star, \nu^\star)$ is a primal-dual optimizer if $x^\star$ and $(\lambda^\star, \nu^\star)$ are respective optimizers for the primal and dual problems and moreover strong duality holds, i.e.,

$$p^\star = f(x^\star) = q(\lambda^\star, \nu^\star) = d^\star.$$

Note that we may also write in this case:

$$x^\star = \operatorname*{argmin}_{x \in \mathbb{R}^n} \tilde{f}(x), \qquad (\lambda^\star, \nu^\star) = \operatorname*{argmax}_{\lambda \in \mathbb{R}^m_+, \, \nu \in \mathbb{R}^p} q(\lambda, \nu).$$

Lemma 12.14 (Effective unconstrained objective)**.** *Suppose that $(x^\star; \lambda^\star, \nu^\star)$ is a primal-dual optimizer. Then*

$$f(x^\star) = \mathcal{L}(x^\star; \lambda^\star, \nu^\star) = q(\lambda^\star, \nu^\star),$$

and moreover

$$x^\star = \operatorname*{argmin}_{x \in \mathbb{R}^n} \mathcal{L}(x; \lambda^\star, \nu^\star).$$

Finally,

$$\lambda_i^\star g_i(x^\star) = 0$$

for all $i = 1, \dots, m$.

Remark 12.15. This result conveys an important intuition, indicating that given the dual optimizer $(\lambda^\star, \nu^\star)$, the function $x \mapsto \mathcal{L}(x; \lambda^\star, \nu^\star)$ defines an *effective unconstrained objective* with the same optimizer as the original *constrained objective*.

Proof. Since $x^\star$ is feasible (i.e., $g(x) \leq 0$, $h(x) = 0$) and $\lambda^\star \geq 0$, it follows that

$$\mathcal{L}(x^\star; \lambda^\star, \nu^\star) = f(x^\star) + \underbrace{\lambda^\star \cdot g(x^\star)}_{\leq 0} + \underbrace{\nu^\star \cdot h(x^\star)}_{=0} \leq f(x^\star) = p^\star. \tag{12.6}$$

Meanwhile,

$$\mathcal{L}(x^\star; \lambda^\star, \nu^\star) \geq \inf_{x \in \mathbb{R}^n} \mathcal{L}(x; \lambda^\star, \nu^\star) = q(\lambda^\star, \nu^\star) = d^\star.$$

Therefore,

$$p^\star = f(x^\star) \geq \mathcal{L}(x^\star; \lambda^\star, \nu^\star) \geq q(\lambda^\star, \nu^\star) = d^\star = p^\star,$$

where the last equality holds by strong duality. It follows that all inequalities hold with equality. This establishes the first statement.

Then since

$$\mathcal{L}(x^\star; \lambda^\star, \nu^\star) = q(\lambda^\star, \nu^\star) = \inf_{x \in \mathbb{R}^n} \mathcal{L}(x; \lambda^\star, \nu^\star),$$

we have precisely that $x^\star$ is a minimizer of $x \mapsto \mathcal{L}(x; \lambda^\star, \nu^\star)$, i.e., the second statement is also proved.

Finally, note that since the inequality in (12.6) holds with equality, in turn it must be the case that $\lambda^\star \cdot g(x^\star) = 0$. Since $\lambda^\star \geq 0$ and $g(x^\star) \leq 0$, the third statement follows. □

---

13 The KKT conditions

The **KKT (Karush-Kuhn-Tucker) conditions** can be viewed as a generalization of the first-order optimality conditions to the context of *constrained* optimization problems. They may seem overwhelming when presented in a vacuum, but they follow logically from what we have already shown in the case where strong duality holds, as we expect for most convex optimization problems.

More generally, the KKT conditions can be justified via a geometric intuition, though the interpretation of the Lagrange multipliers as optimal dual variables does not hold up when strong duality fails.

Theorem 13.1 (KKT conditions under strong duality)**.** *Suppose that $(x^\star; \lambda^\star, \nu^\star)$ is a primal-dual optimizer and that f, g, h are all differentiable. Then the following conditions hold:*

- **Primal feasibility:** $g(x^\star) \leq 0$ and $h(x^\star) = 0$.
- **Dual feasibility:** $\lambda^\star \geq 0$.
- **Complementarity:** $\lambda_i^\star g_i(x^\star) = 0$ for all $i = 1, \ldots, m$.
- **Stationarity:** $\nabla_x \mathcal{L}(x^\star; \lambda^\star, \nu^\star) = 0$, i.e.,

$$\begin{aligned} 0 &= \nabla f(x^\star) + \sum_{i=1}^m \lambda_i^\star \nabla g_i(x^\star) + \sum_{j=1}^p \nu_j^\star \nabla h_j(x^\star) \\ &= \nabla f(x^\star) + Dg(x^\star)^\top \lambda^\star + Dh(x^\star)^\top \nu^\star. \end{aligned} \tag{13.1}$$

Proof. Primal and dual feasibility are automatic. Complementarity follows from Lemma 12.14. Stationarity also follows from Lemma 12.14, via the first-order optimality condition for minimization of $x \mapsto \mathcal{L}(x; \lambda^\star, \nu^\star)$. □

13.1 Geometric perspective

The KKT conditions admit a geometric interpretation that is more robust than strong duality itself.

Consider first the case of a single scalar equality constraint $h(x) = 0$ and no inequality constraints. Here stationarity says that

$$\nabla f(x^\star) = -\nu^\star \nabla h(x^\star),$$

which means precisely that the objective gradient points normal to the constraint hypersurface $\{x \in \mathbb{R}^n : h(x) = 0\}$ at $x^\star$, provided that $\nabla h(x^\star) \neq 0$. Indeed, if there were any tangent component, we could decrease the objective by moving in a tangent direction on the hypersurface.

More generally, consider the case of several equality constraints, where stationarity says that

$$\nabla f(x^\star) = -\sum_{j=1}^p \nu_j^\star \nabla h_j(x^\star).$$

Provided that $\nabla h_j(x^\star)$, $j = 1, \ldots, p$, are linearly independent, they span the subspace normal to the feasible set at $x^\star$, hence again stationarity is equivalent to saying that the objective gradient points normal to the feasible set at $x^\star$.

When we involve inequality constraints, if the inequality constraints are *inactive* at $x^\star$, then complementarity says that the corresponding Lagrange multipliers are zero. But the *active* constraints

induce additional directions in the 'normal cone' to the feasible set—corresponding to nonnegative linear combinations of the gradients $\nabla g_j(x^\star)$—in which it is permissible for the objective gradient to point without inducing a descent direction pointing toward the relative interior of the feasible set.

Theorem 13.2 (KKT conditions in general)**.** *Suppose that $x^\star$ is a primal optimizer and that the collection of all $\nabla g_i(x^\star)$ for i such that $g_i(x^\star) = 0$, together with all $\nabla h_j(x^\star)$ for $j = 1, \dots, p$, form a linearly independent set. Then there exist $\lambda^\star \in \mathbb{R}^m$ and $\nu^\star \in \mathbb{R}^p$ such that the same KKT conditions as in Theorem 13.1 hold.*

Remark 13.3. Note with caution that when strong duality fails, the $\lambda^\star$ and $\nu^\star$ appearing in the statement of this theorem may not be interpretable as dual optimizers. Moreover, $x^\star$ may not be interpretable as an optimizer of $x \mapsto \mathcal{L}(x; \lambda^\star, \nu^\star)$.

13.2 Applications

Next we consider several constrained optimization problems that can be solved exactly using the KKT conditions.

Example 13.4 (Eigenvalue problem as optimization problem)**.** Consider optimizing a quadratic form defined by a symmetric matrix $A \in \mathbb{R}^{n \times n}$ over the unit sphere:

$$\begin{aligned} \underset{x \in \mathbb{R}^n}{\text{minimize}} \quad & x^\top A x \\ \text{subject to} \quad & \|x\|_2 = 1. \end{aligned}$$

We can rewrite the constraint as $h(x) = 0$ where $h(x) = 1 - \|x\|_2^2$ and then form the Lagrangian

$$\mathcal{L}(x; \nu) = x^\top A x + \nu(1 - \|x\|^2).$$

The KKT conditions says that for an optimizer x, there exists $\nu \in \mathbb{R}$ such that

$$Ax = \nu x, \quad \|x\|_2 = 1,$$

i.e., x is a normalized eigenvector.

For any such x, the objective takes the value $x^\top A x = \nu x^\top x = \nu$, i.e., the coresponding eigenvalue. Therefore an *optimal* $x^\star$ is given by a normalized eigenvector corresponding to the *lowest* eigenvalue.

Remark 13.5. The preceding example establishes the simplest case of the *Courant-Fischer minimax principle*. This principle can be understood more generally via duality, but further discussion is outside of our scope.

Exercise 13.6 (Constrained least squares)**.** Consider the constrained least squares problem:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\|Ax-b\|_2^2 \\ \text{subject to} \quad & Cx = d. \end{aligned}$$

Show that a primal-dual optimizer $(x^\star, \nu^\star)$ can be recovered by solving the linear system:

$$\begin{pmatrix} A^\top A & C^\top \\ C & 0 \end{pmatrix} \begin{pmatrix} x^\star \\ \nu^\star \end{pmatrix} = \begin{pmatrix} A^\top b \\ d \end{pmatrix}.$$

Exercise 13.7 (Projection onto an affine subspace)**.** Consider a general affine subspace $S := \{x \in \mathbb{R}^n \,:\, Ax = b\}$ where A, b are arbitrary of compatible shapes. Find an explicit formula for the projection $\Pi_S(y)$ by considering the constrained least squares problem

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & \frac{1}{2}\|x-y\|_2^2 \\ \text{subject to} \quad & Ax = b. \end{aligned}$$

In the special case $A = a^\top \in \mathbb{R}^{1\times n}$, $b \in \mathbb{R}$, so that $S = \{x \in \mathbb{R}^n \,:\, a \cdot x = b\}$ is a hyperplane, conclude that

$$\Pi_S(y) = y + \frac{b - a\cdot y}{\|a\|^2} a.$$

Exercise 13.8 (Projection onto a half-space)**.** Consider a half-space $S := \{x \in \mathbb{R}^n \,:\, a\cdot x \le b\}$. Use the KKT conditions to show that

$$\Pi_S(y) = \begin{cases} y + \frac{b-a\cdot y}{\|a\|^2} a, & \text{if } \; a\cdot y > b, \\ y, & \text{if } \; a\cdot y \le b. \end{cases}$$

13.3 Revisiting the trust region subproblem

Recall the 'trust region subproblem,' whose solution we asserted in Theorem 9.2:

$$\begin{aligned} \underset{s\in\mathbb{R}^n}{\text{minimize}} \quad & f(s) := b\cdot s + \frac{1}{2} s^\top A s \\ \text{subject to} \quad & \frac{1}{2}\left[\|s\|_2^2 - \Delta^2\right] \le 0. \end{aligned}$$

Note that we have rewritten the constraint in a particular way to make the dual problem look nicer.

The KKT conditions suggest that the optimizer must take the form $x^\star = -(A + \lambda^\star I)^{-1} b$ for some $\lambda^\star$. However, it is difficult to understand from the KKT conditions alone why $\lambda^\star$ should be chosen in the way prescribed by the theorem (in particular, always such that $A + \lambda^\star I$ is positive semidefinite).

Thus while the KKT conditions offer a key intuition, we need a different approach for a rigorous proof. Our approach will solve the dual problem, yielding $\lambda^\star$ matching the prescription from the theorem statement. This will allow us to define $s^\star = -(A + \lambda^\star I)^{-1} b$ which we can verify is indeed primal-optimal via weak duality.

Proof of Theorem 9.2. We will ignore Case 3 for brevity. First focus on Case 2, where $\lambda_{\min} \leq 0$ and $b \notin \operatorname{col}(A - \lambda_{\min} I)$. The discussion after Theorem 9.2 established that in this case, the function $\psi(\lambda) = \|(A + \lambda I)^{-1} b\|^2$ is decreasing with $\psi(\lambda) \to +\infty$ as $\lambda \to -\lambda_{\min}$ from above. It is likewise not difficult to see via diagonalization that $b^\top (A + \lambda I)^{-1} b \to +\infty$ as $\lambda \to -\lambda_{\min}$ from above.

Since the problem is nonconvex, we cannot guarantee strong duality by any abstract condition. We will verify strong duality directly in the process of the proof, in which we solve the dual problem for $\lambda^\star$, establishing that the dual optimizer $\lambda^\star$ does in fact match the choice of $\lambda^\star$ indicated in the theorem statement. Then we will use $\lambda^\star$ to construct the primal-feasible point $s = -(A + \lambda^\star I)^{-1} b$, which we verify to be optimal by checking that $f(s) = d^\star$. This is sufficient because then automatically $p^\star \leq f(s) = d^\star \leq p^\star$, where the last inequality follows from weak duality, and in turn we know that $f(s) = p^\star$, i.e., s is primal-optimal.

To begin, consider the Lagrangian (in which we denote the primal variable by s for consistency with the previous presentation):

$$\begin{aligned} \mathcal{L}(s; \lambda) &= b \cdot s + \frac{1}{2} s^\top A s + \frac{\lambda}{2} \left[\|s\|_2^2 - \Delta^2 \right] \\ &= -\frac{\Delta^2}{2} \lambda + b \cdot s + \frac{1}{2} s^\top (A + \lambda I) s. \end{aligned}$$

We can compute

$$\inf_{s \in \mathbb{R}^n} \left\{ b \cdot s + \frac{1}{2} s^\top (A + \lambda I) s \right\} = \begin{cases} -\infty, & \text{if } \lambda \leq -\lambda_{\min}, \\ -\frac{1}{2} b^\top (A + \lambda I)^{-1} b, & \text{if } \lambda > -\lambda_{\min}. \end{cases}$$

The case $\lambda < -\lambda_{\min}$ is clear because $A + \lambda I$ has a negative eigenvalue in this case. Meanwhile the case $\lambda > \lambda_{\min}$ follows by solving the resulting convex quadratic optimization problem for $s = -(A + \lambda I)^{-1} b$ and plugging in. The boundary case $\lambda = \lambda_{\min}$ follows via 'continuity' from the fact that $b^\top (A + \lambda I)^{-1} b \to +\infty$ as $\lambda \to -\lambda_{\min}$ from above.

Thus the dual problem can be written

$$\begin{aligned} \underset{\lambda \in \mathbb{R}}{\text{maximize}} \quad & q(\lambda) := -\frac{\Delta^2}{2} \lambda - \frac{1}{2} b^\top (A + \lambda I)^{-1} b \\ \text{subject to} \quad & \lambda > -\lambda_{\min}. \end{aligned}$$

Note that the ordinary dual feasibility constraint $\lambda \geq 0$ is redundant because $-\lambda_{\min} \geq 0$.

We can solve the dual problem for $\lambda^\star$ by solving for a critical point of the dual objective $q(\lambda)$:

$$\begin{aligned} q'(\lambda) &= -\frac{\Delta^2}{2} + \frac{1}{2} b^\top (A + \lambda I)^{-2} b \\ &= -\frac{\Delta^2}{2} + \frac{1}{2} \|(A + \lambda I)^{-1} b\|^2, \end{aligned}$$

where we have used matrix calculus to differentiate the inverse of a matrix. (Note that we can simplify to scalar calculus by assuming a diagonal matrix A via diagonalization.) Setting $q'(\lambda) = 0$ indicates that $\lambda^\star$ is indeed the unique $\lambda^\star > -\lambda_{\min}$ satisfying $\|(A + \lambda^\star I)^{-1} b\| = \Delta$, which we constructed in the discussion after the statement of Theorem 2.72.

Therefore $s := -(A + \lambda^\star I)^{-1} b$ is primal-feasible, and it only remains to show that $f(s) = d^\star$. To get started, first plug in to compute $d^\star = q(\lambda^\star)$:

$$d^\star = -\frac{\Delta^2}{2} \lambda^\star - \frac{1}{2} b^\top (A + \lambda^\star I)^{-1} b. \tag{13.2}$$

Then plug in to compute $f(s)$:

$$\begin{aligned} f(s) &= b \cdot s + \frac{1}{2} s^\top A s \\ &= -b^\top (A + \lambda^\star I)^{-1} b + \frac{1}{2} b^\top (A + \lambda^\star I)^{-1} A (A + \lambda^\star I)^{-1} b. \end{aligned}$$

Now within the last term, we can substitute $A = (A + \lambda^\star I) - \lambda^\star I$ and simplify to deduce

$$\begin{aligned} f(s) &= -\frac{1}{2} b^\top (A + \lambda^\star I)^{-1} b - \frac{\lambda^\star}{2} b^\top (A + \lambda^\star I)^{-1} (A + \lambda^\star I)^{-1} b \\ &= -\frac{1}{2} b^\top (A + \lambda^\star I)^{-1} b - \frac{\lambda^\star}{2} \|(A + \lambda^\star I)^{-1} b\|^2 \\ &= d^\star, \end{aligned}$$

by comparison to (13.2), using the fact that $\|(A + \lambda^\star I)^{-1} b\| = \Delta$. This completes the proof in Case 2.

In Case 1, the derivation of the dual problem is largely the same, except that now $-\lambda_{\min} < 0$, so the dual is

$$\begin{aligned} \underset{\lambda \in \mathbb{R}}{\text{maximize}} \quad & q(\lambda) := -\frac{\Delta^2}{2} \lambda - \frac{1}{2} b^\top (A + \lambda I)^{-1} b \\ \text{subject to} \quad & \lambda \geq 0. \end{aligned}$$

If $q'(0) \geq 0$, i.e., if $\|A^{-1} b\| \leq \Delta$, then there can be no interior dual critical point, hence $\lambda^\star = 0$ is the dual optimizer. Otherwise, $\|A^{-1} b\| > \Delta$, and the dual optimizer satisfies $q'(\lambda^\star) = 0$ as in Case 2. The remaining arguments follows similarly. □

---

14 Exact and inexact augmented Lagrangian methods

Next we introduce a framework that can flexibly reduce an equality-constrained optimization problem to a sequence of unconstrained optimization problems. Extensions to explicit inequality constraints can also been considered, though they are beyond the scope of this course.

Indeed, consider the problem:

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) \\ \text{subject to} \quad & h(x) = 0. \end{aligned} \tag{14.1}$$

In spite of the fact that this formulation does not permit *explicit* inequality constraints, in fact we may include inequality constraints *implicitly* within the objective by allowing it to take the value $+\infty$. In terms of duality, this point is equivalent to noticing that *we do not need to dualize all the constraints that we can write down* (some constraints can remain effectively as constraints on the primal domain within the Lagrangian).

14.1 The augmented Lagrangian

A naive approach for attempting to reduce (14.1) to an unconstrained optimization problem is to simply solve

$$\underset{x\in\mathbb{R}^n}{\text{minimize}} \quad f(x) + \frac{\rho}{2}\|h(x)\|^2 \tag{14.2}$$

for a large value of $\rho > 0$. This is called a **penalty method**, and while it is not a completely useless idea, it is unsatisfying in that it requires us to take a limit as $\rho \to +\infty$ to recover an exact solution to the original problem. Meanwhile, the conditioning of the problem (14.2) (via the increasing L-smoothness parameter) becomes infinitely bad in this limit, and we face a vexing tradeoff between accuracy and computational tractability.

The augmented Lagrangian method is based on the following equivalent reformulation of (14.1):

$$\begin{aligned} \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad & f(x) + \frac{\rho}{2}\|h(x)\|^2 \\ \text{subject to} \quad & h(x) = 0, \end{aligned} \tag{14.3}$$

which keeps the equality constraint intact. While this maneuver may seem useless, it actually induces a different Lagrangian:

$$\mathcal{L}_\rho(x;\nu) = f(x) + \frac{\rho}{2}\|h(x)\|^2 + \nu \cdot h(x),$$

called the **augmented Lagrangian**, which is defined in terms of a parameter $\rho > 0$. A key point is that ρ does not have to be infinitely large to ensure consistency. In practice it will control a computational balance between the rates at which we decrease the objective and the feasibility error, respectively.

14.2 The augmented Lagrangian method (ALM)

The **augmented Lagrangian method (ALM)** is a *meta-algorithm* defined by the following update rule

$$\begin{aligned} x_{t+1} &:= \underset{x\in\mathbb{R}^n}{\operatorname{argmin}} \left\{\mathcal{L}_\rho(x;\nu_t)\right\}, \\ \nu_{t+1} &:= \nu_t + \rho\, h(x_t). \end{aligned}$$

Typically we initialize $\nu_0 = 0$ and x_0 is chosen via some *a priori* intuition, or else $x_0 = 0$.

Note that the first part of the ALM update is an exact minimization of the augmented Lagrangian over the primal variables, with the dual variables frozen. The second part can be viewed as a gradient *ascent* step on the augmented Lagrangian over the dual variables ν, with the primal variables frozen. Intriguingly, the step size is always fixed to be ρ, the augmented Lagrangian parameter. The justification for this choice will be explained below.

For now, we make the following basic observation:

Exercise 14.1. Any primal-dual optimizer $(x^\star; \nu^\star)$ is a fixed point of the ALM iteration.

We say that ALM is a meta-algorithm because we have not specified how to perform the primal minimization step defining x_{t+1} in terms of ν_t. This step requires us only to solve an *unconstrained* optimization problem, and we could for instance apply gradient descent or a Newton-type method.

In practice, we cannot solve each such problem exactly. A sensible convergence criterion is to insist that $\|\nabla_x \mathcal{L}_\rho(x; \nu_t)\| \leq \varepsilon$ for some tolerance $\varepsilon > 0$. More advanced strategies can be devised to avoid working too hard in the early iterations where it is not yet important to converge the solver to high accuracy.

Since ALM is a ***primal-dual*** method, our convergence metrics need not only to keep track of both feasibility and stationarity errors. In other words, we should measure the failure to satisfy all relevant KKT conditions. In this case, the KKT conditions simply require $h(x) = 0$ and $\nabla_x \mathcal{L}(x; \nu) = 0$. Therefore a reasonable convergence metric for a primal-dual pair (x, ν) is given by $\sqrt{\|h(x)\|^2 + \|\nabla_x \mathcal{L}(x; \nu)\|^2}$. Alternatively, we may wish to track $\|h(x)\|$ and $\|\nabla_x \mathcal{L}(x; \nu)\|$, respectively called the **primal and dual residuals**, separately.

14.3 ALM as dual proximal point method

Theorem 14.2 (ALM as PPM)**.** *If f is strongly convex and $h(x) = Ax - b$, i.e., the equality constraints are affine-linear, the ALM iterates satisfy*

$$\nu_{t+1} = \operatorname*{argmax}_{\nu \in \mathbb{R}^p} \left\{ q(\nu) - \frac{1}{2\rho} \|\nu - \nu_t\|_2^2 \right\}.$$

Remark 14.3. This is equivalent ot saying that ALM is a **proximal point method (PPM)** for the dual maximization problem. Note however that implementing the ALM does not actually require us to form the dual problem, which may in general be intractable to formulate analytically! Moreover, this result clarifies that ALM converges faster in terms of the number of iterations when ρ is larger, but note that the cost of each iteration may be greater when ρ is large due to the worsening of the conditioning.

Proof sketch. We will only give the intution of the result. Consider a modified Lagrangian

$$\mathcal{L}_{\rho,t}(x; \nu) = f(x) + \nu \cdot (Ax - b) - \frac{1}{2\rho} \|\nu - \nu_t\|_2^2.$$

We will claim that $(x_{t+1}; \nu_{t+1})$ is 'primal-dual optimal' for this Lagrangian. Let $(\tilde{x}; \tilde{\nu})$ denote the primal-dual optimal pair, assuming that it exists and a suitable minimax principle holds, such that in particular

$$\tilde{\nu} = \operatorname*{argmax}_{\nu \in \mathbb{R}^p} \mathcal{L}_{\rho,t}(\tilde{x}; \nu). \tag{14.4}$$

Then observe that

$$\inf_{x\in\mathbb{R}^n} \mathcal{L}_{\rho,t}(x;\nu) = q(\nu) - \frac{1}{2\rho}\|\nu-\nu_t\|_2^2$$

by the definition of q, hence also

$$\tilde{\nu} = \operatorname*{argmax}_{\nu\in\mathbb{R}^p} \left\{ q(\nu) - \frac{1}{2\rho}\|\nu-\nu_t\|_2^2 \right\}. \tag{14.5}$$

Meanwhile, for any x, we can compute analytically that

$$\operatorname*{argmax}_{\nu\in\mathbb{R}^p} \mathcal{L}_{\rho,t}(x;\nu) = \nu_t + \rho(Ax-b), \tag{14.6}$$

hence by plugging in we deduce that

$$\sup_{\nu\in\mathbb{R}^p} \mathcal{L}_{\rho,t}(x;\nu) = f(x) + \nu_t\cdot(Ax-b) + \frac{\rho}{2}\|Ax-b\|^2 = \mathcal{L}_\rho(x;\nu_t)$$

and in turn

$$\tilde{x} = \operatorname*{argmin}_{x\in\mathbb{R}^n} \left\{ \sup_{\nu\in\mathbb{R}^p} \mathcal{L}_{\rho,t}(x;\nu) \right\} = \operatorname*{argmin}_{x\in\mathbb{R}^n} \{\mathcal{L}_\rho(x;\nu_t)\} = x_{t+1},$$

by the definition of the ALM.

From (14.4) and (14.6) it then follows that

$$\tilde{\nu} = \nu_t + \rho(Ax_{t+1}-b) = \nu_{t+1},$$

hence by (14.5) we conclude. □

14.4 The alternating direction method of multipliers (ADMM)

Some approaches based on the augmented Lagrangian do not attempt to perform primal minimization exactly or even approximately within each iteration. Such approaches are called **inexact ALMs**, and the most famous example is called the **alternating direction method of multipliers (ADMM)**. In spite of its apparent specificity, ADMM actually offers a surprisingly flexible framework for easily implementable algorithms in a number of contexts, as we shall illustrate below.

We consider a structured case of (14.3) in which the primal variables split into two pieces x and y which are coupled via an affine-linear constraint and in terms of which the objective is separable:

$$\begin{aligned} \underset{x\in\mathbb{R}^n,\, y\in\mathbb{R}^m}{\text{minimize}} \quad & f(x)+g(y) \\ \text{subject to} \quad & Ax+By=c. \end{aligned} \tag{14.7}$$

We can write down a suitable augmented Lagrangian as above

$$\mathcal{L}_\rho(x,y;\nu) \;=\; f(x)+g(y)+\nu\cdot(Ax+By-c)+\frac{\rho}{2}\|Ax+By-c\|^2$$

but in stead of performing exact primal minimization over (x,y) in each iteration, we perform alternating exact updates of x and y before each dual variable update.

To wit, the ADMM algorithm is defined by the update rule:

$$
\begin{aligned}
x_{t+1} &:= \operatorname*{argmin}_{x \in \mathbb{R}^n} \left\{ \mathcal{L}_\rho(x, y_t; \nu_t) \right\}, \\
y_{t+1} &:= \operatorname*{argmin}_{y \in \mathbb{R}^m} \left\{ \mathcal{L}_\rho(x_{t+1}, y; \nu_t) \right\}, \\
\nu_{t+1} &:= \nu_t + \rho \left(A x_{t+1} + B y_{t+1} - c \right).
\end{aligned}
$$

The key point is that for many problems of interest (depending on the structure of f and g), the x and y updates can be performed *analytically*.

To measure convergence, we track the **primal residual** $\|Ax_{t+1} + By_{t+1} - c\|$ as well as the **dual residual** $\rho \|A^\top B(y_{t+1} - y_t)\|$, which again corresponds to a notion of error in the KKT stationarity condition, though the derivation of this formula is a bit tricky, cf. the famous paper of Boyd et al (2010).

14.5 Applications of ADMM

Next we explain how solvers for several important problems can be implemented simply in the ADMM framework.

14.5.1 LASSO via ADMM

Consider the problem

$$
\operatorname*{minimize}_{x \in \mathbb{R}^n} \quad \frac{1}{2} \|Ax - b\|_2^2 + \lambda \|x\|_1.
$$

We can transform this into the format (14.7) by introducing an auxiliary variable $y = x$:

$$
\begin{aligned}
\operatorname*{minimize}_{x \in \mathbb{R}^n,\, y \in \mathbb{R}^n} \quad & \frac{1}{2} \|Ax - b\|_2^2 + \lambda \|y\|_1 \\
\text{subject to} \quad & x = y.
\end{aligned}
$$

Then write out the augmented Lagrangian

$$
\mathcal{L}_\rho(x, y; \nu) = \frac{1}{2} \|Ax - b\|_2^2 + \lambda \|y\|_1 + \nu \cdot (x - y) + \frac{\rho}{2} \|x - y\|^2,
$$

observe that the x-update in ADMM can be computed exactly as it requires solving a convex quadratic optimization problem:

$$
\operatorname*{argmin}_{x \in \mathbb{R}^n} \mathcal{L}_\rho(x, y; \nu) = (A^\top A + \rho I)^{-1} (A^\top b + \rho y - \nu).
$$

Meanwhile, by completing the square we can show that the y-update reduces to solving

$$
\begin{aligned}
\operatorname*{argmin}_{y \in \mathbb{R}^n} \mathcal{L}_\rho(x, y; \nu) &= \operatorname*{argmin}_{y \in \mathbb{R}^n} \left\{ \lambda \|y\|_1 + \frac{\rho}{2} \|x - y + \rho^{-1} \nu\|^2 \right\} \\
&= \operatorname{prox}_{\rho^{-1} \lambda \|\cdot\|_1} \left(x + \rho^{-1} \nu \right) \\
&= S_{\rho^{-1} \lambda} \left(x + \rho^{-1} \nu \right),
\end{aligned}
$$

where S_α denotes the shrinkage-thresholding operator (cf. Exercise 6.16).

In summary, ADMM in this setting yields the update rule:

$$
\begin{aligned}
x_{t+1} &:= (A^\top A + \rho I)^{-1} (A^\top b + \rho y_t - \nu_t), \\
y_{t+1} &:= S_{\rho^{-1} \lambda} \left(x_{t+1} + \rho^{-1} \nu_t \right), \\
\nu_{t+1} &:= \nu_t + \rho \left(x_{t+1} - y_{t+1} \right)
\end{aligned}
$$

14.5.2 Projection onto an intersection

Let $C_1, C_2 \subset \mathbb{R}^n$ be two convex sets, and suppose that we can compute the projections Π_{C_1} and Π_{C_2} efficiently. We can use these operations to compute the projection $\Pi_{C_1 \cap C_2}$ using an iterative algorithm derived via ADMM.

Indeed, note that to compute $\Pi_{C_1 \cap C_2}(z)$, we want to solve the minimization problem:

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n}{\text{minimize}} & \frac{1}{2}\|x - z\|^2 \\ \text{subject to} & x \in C_1,\ x \in C_2. \end{array}$$

By introducing an auxiliary variable $y = x$ and incorporating the constraints $x \in C_1$ and $y \in C_2$ into the objective via indicator functions, we obtain the equivalent problem which fits the format (14.7):

$$\begin{array}{ll} \underset{x \in \mathbb{R}^n,\, y \in \mathbb{R}^n}{\text{minimize}} & \frac{1}{2}\|x - z\|^2 + \iota_{C_1}(x) + \iota_{C_2}(y) \\ \text{subject to} & x = y. \end{array}$$

Then write out the augmented Lagrangian

$$\mathcal{L}_\rho(x, y; \nu) = \frac{1}{2}\|x - z\|^2 + \iota_{C_1}(x) + \iota_{C_2}(y) + \nu \cdot (x - y) + \frac{\rho}{2}\|x - y\|^2.$$

We consider $\rho = 1$ for simplicity, observe that the ADMM updates can be computed exactly by completing squares to deduce that

$$\underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \mathcal{L}_1(x, y; \nu) = \underset{x \in C_1}{\operatorname{argmin}} \|x - (z + y - \nu)/2\|^2 = \Pi_{C_1}\left(\frac{1}{2}[z + y - \nu]\right)$$

and

$$\underset{y \in \mathbb{R}^n}{\operatorname{argmin}} \mathcal{L}_1(x, y; \nu) = \underset{y \in C_2}{\operatorname{argmin}} \|(x + \nu) - y\|^2 = \Pi_{C_2}\left(x + \nu\right).$$

In summary, ADMM in this setting yields the update rule:

$$\begin{aligned} x_{t+1} &:= \Pi_{C_1}\left(\frac{1}{2}[z + y_t - \nu_t]\right), \\ y_{t+1} &:= \Pi_{C_2}\left(x_{t+1} + \nu_t\right), \\ \nu_{t+1} &:= \nu_t + \rho\left(x_{t+1} - y_{t+1}\right). \end{aligned}$$

15 Optimal transport and Sinkhorn scaling

Finally we return to the optimal transport linear program and its entropy-regularized modification introduced in Example 2.82. Here we will discuss the dual perspective, which yields a famous fast algorithm for the regularized problem. Note that in this discussion, ν does not denote a dual variable. Instead we will instroduce ϕ and ψ as dual variables for the two vector equality constraints.

Exercise 15.1 (Kantorovich duality)**.** Show that the linear program dual to (2.20) is given by

$$\begin{aligned} \underset{\phi \in \mathbb{R}^m,\, \psi \in \mathbb{R}^n}{\text{maximize}} \quad & \mu \cdot \phi + \nu \cdot \psi \\ \text{subject to} \quad & \phi \mathbf{1}_n^\top + \mathbf{1}_m \psi^\top \leq C. \end{aligned}$$

The inequality constraint can be written entrywise as $\phi_i + \psi_j \leq C_{ij}$ for all $i = 1, \ldots, m$, $j = 1, \ldots, n$.

Exercise 15.2 (Entropy-regularized Kantorovich duality)**.** Show that the problem dual to (2.21) is given by

$$\underset{\phi \in \mathbb{R}^m,\, \psi \in \mathbb{R}^n}{\text{maximize}} \quad q_\beta(\phi, \psi) := \mu \cdot \phi + \nu \cdot \psi - \beta^{-1} \sum_{i=1}^{m} \sum_{j=1}^{n} e^{-\beta(C_{ij} - \phi_i - \psi_j)}.$$

The dual of the entropy-regularized optimal transport problem admits a simple fast algorithm based on block coordinate ascent, which repeats the following updates:

$$\begin{aligned} \phi &\leftarrow \underset{\phi \in \mathbb{R}^m}{\operatorname{argmax}}\ q_\beta(\phi, \psi), \\ \psi &\leftarrow \underset{\psi \in \mathbb{R}^m}{\operatorname{argmax}}\ q_\beta(\phi, \psi). \end{aligned}$$

These updates can be exactly computed according to the following exercise:

Exercise 15.3 (Sinkhorn scaling)**.** Show that the preceding block coordinate ascent algorithm can be implemented explicitly via:

$$\begin{aligned} \phi_i &\leftarrow \beta^{-1} \left[\log \mu_i - \log \sum_{j=1}^{n} e^{-\beta(C_{ij} - \psi_j)} \right], \quad i = 1, \ldots, m, \\ \psi_j &\leftarrow \beta^{-1} \left[\log \nu_j - \log \sum_{i=1}^{m} e^{-\beta(C_{ij} - \phi_i)} \right], \quad j = 1, \ldots, n. \end{aligned}$$

Hint: Solve the first-order optimality conditions $\nabla_\phi q_\beta(\phi, \psi) = 0$ and $\nabla_\psi q_\beta(\phi, \psi) = 0$.