

Convex Optimization by Prof. Stephen Becker

Jaden Wang

May 23, 2021

Contents

1	Theoretical Foundation	4
1.1	Introduction	5
1.1.1	Lipschitz continuity	7
1.1.2	Categorization	9
1.1.3	Minimizers	10
1.1.4	Convexity	12
1.2	Convex Sets [BV04 Ch.2]	16
1.2.1	Convex, affine, and cone	16
1.2.2	Important examples	18
1.3	Operations that preserve convexity	23
1.3.1	Linear-fractional and perspective functions	23
1.3.2	Generalized inequalities	24
1.3.3	separating and supporting hyperplanes	26
1.4	Convex Functions [BV04 Ch.3]	30
1.4.1	First-order conditions	34
1.4.2	Calculus	39
1.4.3	Lipschitz gradient	41
1.4.4	Examples [BV04 Ch.3.1.5]	44
1.4.5	Preserving convexity	46
1.4.6	Conjugate functions	51
1.4.7	Existence and uniqueness of minimizers	57
1.4.8	Proximity operator	59
1.4.9	Moreau envelope	62
1.5	Convex optimization problems	63

1.5.1	Tricks	63
1.5.2	Convex optimization problems [BV04 Ch.4.2]	65
1.5.3	Conic optimization problems [BV04 Ch.4.3, 4.4, 4.6]	67
1.6	Duality [BV04 Ch.5]	73
1.6.1	Lagrange dual function/problem	73
1.6.2	Saddle point interpretation [BV 4.2]	78
1.6.3	Game Theory connection	80
1.6.4	Fenchel-Rockafellar Duality [BC17]	81
1.6.5	Algorithms	83
1.6.6	Optimality conditions	84
1.6.7	Meta-rules	88
1.6.8	Perturbation and Sensitivity Analysis [BV 5.6]	90
1.6.9	Generalized Inequalities [BV 5.9]	92
2	Algorithms	94
2.1	Unconstrained Optimization	95
2.1.1	Proximal gradient descent	96
2.1.2	Gradient descent	98
2.1.3	Linear conjugate gradient method	106
2.1.4	non-linear conjugate gradient	108
2.1.5	Quasi-Newton Methods	110
2.1.6	Newton's methods	114
2.1.7	Nonlinear least squares	116
2.2	Methods for constrained problems	117
2.2.1	penalty methods	118
2.2.2	Augmented Lagrangian	120
2.2.3	Newton's Method revisited, [BV04] Ch.9	123
2.2.4	Primal Dual Methods	131
2.3	Linear Programs	133
2.3.1	Simplex Method	133
2.4	Bonus: find gradients	135
2.4.1	DFO	135
2.4.2	Finite differences	136
2.4.3	analytic solutions	137

2.4.4	Automatic differentiation	138
2.4.5	by hand	138
2.4.6	Adjoint state method	140

Chapter 1

Theoretical Foundation

1.1 Introduction

An optimization problem looks like

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

where $f(x)$ is the **objective function** and $C \subseteq \mathbb{R}^n$ is the **constraint set**. C might look like

$$C = \{x : g_i(x) \leq 0 \ \forall i = 1, \dots, m\}.$$

Remark 1.1.1 We can always turn a maximization problem into a minimization problem as the following:

$$\min_x f(x) = -\max_x -f(x).$$

Therefore, WLOG, we will stick with minimization.

Example 1.1.2

An assistant professor earns \$100 per day, and they enjoy both ice cream and cake. The optimization problem aims to maximize the utility (*e.g.* happiness) of ice cream $f_1(x_1)$ and of cake $f_2(x_2)$. The constraints we have is that $x_1 \geq 0, x_2 \geq 0$, and $x_1 + x_2 \leq 100$.

To maximize both utility, it might be natural to define

$$F(\text{vec } x) = \begin{pmatrix} f_1(x_1) \\ f_2(x_2) \end{pmatrix}, \text{vec } x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

and maximize F . However, this isn't a well-defined problem, because *there is no total order on \mathbb{R}^n* ! That is, we don't have a good way to compare whether a vector is bigger than another vector, except in the cases when the same direction of inequality can be achieved for all components of two vectors and a partial order can be established. For this kind of **multi-objective** optimization problem, we can look for Pareto-optimal points in these special cases. We can also try to convert the output into a scalar as the following:

$$\min_x f_1(x) + \lambda \cdot f_2(x_2)$$

for some $\lambda > 0$ that reflects our preference for cake vs ice cream. But this can be subjective.

Thus, For the remainder of this class, we are only going to assume $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

Moreover, for $f : \mathbb{R} \rightarrow \mathbb{R}$, it's very easy to solve by using root finding algorithms or grid search. So since interesting problems occur with vector inputs, we will simply use x to represent vectors.

Notation. \min asks for the minimum value, whereas $\arg \min$ asks for the minimizer that yields the minimum value.

1.1.1 Lipschitz continuity

Example 1.1.3

Let's consider a variant of the Dirichlet function, $f : \mathbb{R} \rightarrow \mathbb{R}$

$$f(x) = \begin{cases} x & \text{if } x \in \mathbb{Q} \\ 1 & \text{if } x \in \mathbb{R} \setminus \mathbb{Q} \end{cases}$$

Then the solution to the problem

$$\min_{x \in [0,1]} f(x) = 0$$

is $x = 0$ by observation. However, the function is not smooth and a small perturbation can yield wildly different values. Thus, it is not tractable to solve this numerically.

This requires us to add a smoothness assumption:

Definition 1.1.4 — $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is **L -Lipschitz continuous** with respect to a norm $\|\cdot\|$ if for all $x, y \in \mathbb{R}^n$,

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

Note. Lipschitz continuity implies continuity and uniform continuity. It is a stronger statement because it tells us *how* the function is (uniformly) continuous. However, it doesn't require differentiability.

Definition 1.1.5 — For $1 \leq p < \infty$,

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

For $p = \infty$,

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

Remark 1.1.6 $\|x\|_1$ and $\|x\|_2^2$ have separable terms as they are sums of their components. $\|x\|_2^2$ is also differentiable which makes it the nicest norm to optimize.

Example 1.1.7

Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$ be L -Lipschitz continuous w.r.t. $\|\cdot\|_\infty$. Let $C = [0, 1]^n$, *i.e.* in \mathbb{R}^2 , C is a square. To solve the problem

$$\min_{x \in C} f(x),$$

since we have few assumption, there is no better method (in the worst case sense) than the **uniform grid method**. The idea is that we pick $p + 1$ points in each dimension, *i.e.* $\{0, \frac{1}{p}, \frac{2}{p}, \dots, 1\}$, so we would have $(p + 1)^n$ points in total.

Let x^* be a global optimal point, then there exists a grid point \tilde{x} s.t.

$$\|x^* - \tilde{x}\|_\infty \leq \frac{1}{2} \cdot \frac{1}{p}.$$

Thus by Lipschitz continuity,

$$\begin{aligned} |f(x^*) - f(\tilde{x})| &\leq L \cdot \|x^* - \tilde{x}\|_\infty \\ &\leq \frac{1}{2} \frac{L}{p} \end{aligned}$$

So we can find \tilde{x} by taking the discrete minimum of all $(p + 1)^n$ grid points.

In (non-discrete) optimization, we usually can't exactly find the minimizer, but rather find something very close.

Definition 1.1.8 — x is a ϵ -optimal solution to $\min_{x \in C} f(x)$ if $x \in C$ and

$$f(x) - f^* \leq \epsilon$$

where $f^* = \min_{x \in C} f(x)$.

Our uniform grid method gives us an ε -optimal solution with $\varepsilon = \frac{L}{2p}$, and requires $(p+1)^n$ function evaluations. Writing p in terms of ε , we have $p = \frac{L}{2\varepsilon}$ so equivalently it requires $(\frac{2L}{\varepsilon} + 1)^n$ function evaluations, which approximately is ε^{-n} .

For $\varepsilon = 10^{-6}$, $n = 100$, it requires 10^{600} function evaluations. This is really bad!

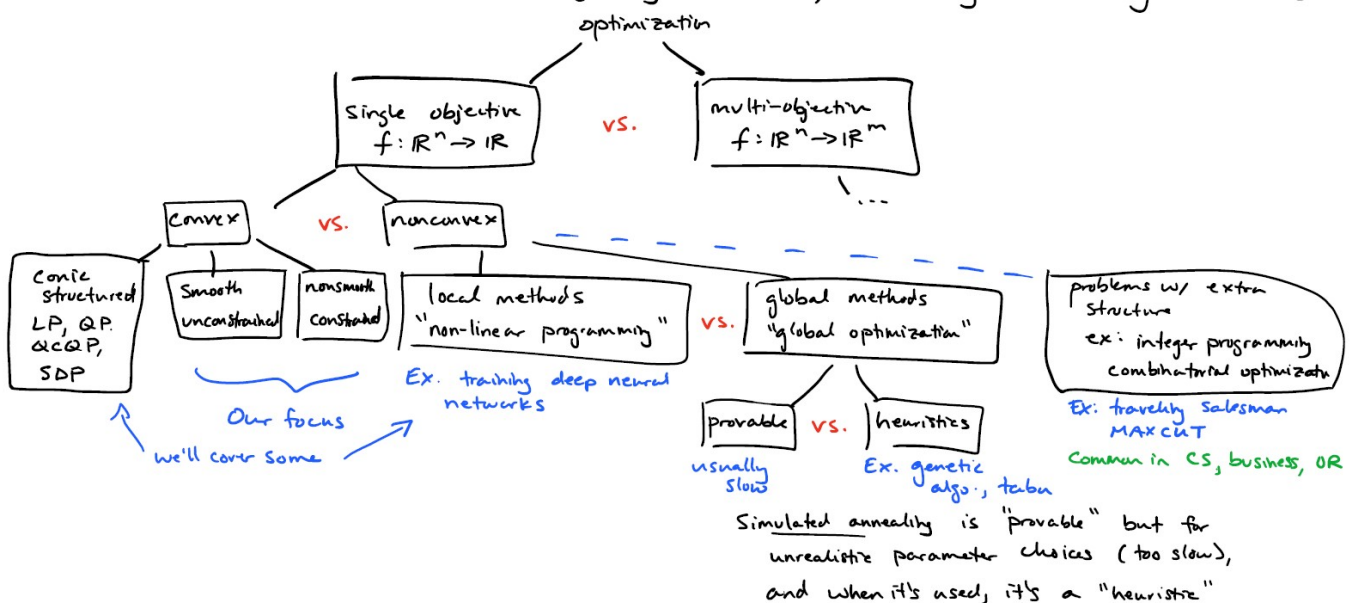
Take-aways from this example:

- curse-of-dimensionality: there can be trillions of variables in a Google Neural Network. It would be intractable using the grid method.
- we need more assumptions to allow us to use more powerful methods.

1.1.2 Categorization

Types of optimization problems

This classification isn't the only way to do it, and may reflect my own biases



1.1.3 Minimizers

We are given a generic problem $\min_{x \in C} f(x)$, $C \subseteq \mathbb{R}^n$. Then a **feasible point** x means $x \in C$. A **solution** or **minimizer** or **global minimizer** x^* means

- 1) $x^* \in C$
- 2) $\forall y \in C, f(x^*) \leq f(y)$

It might not be unique, *i.e.* $x^* \in \arg \min_{x \in C} f(x)$.

Example 1.1.9

$$\min_{x \in \mathbb{R}} f(x) \text{ where } f(x) = 0 \forall x.$$

Sometimes the solution may not exist (even for convex problems).

Example 1.1.10

$$\min_{x \in (0,1)} x^2.$$

x^* is a **local minimizer** if x^* is feasible and there exists an $\varepsilon > 0$ s.t. $f(x^*) \leq f(y) \forall y \in C \cap B_\varepsilon(x^*) := \{y : \|y - x^*\| < \varepsilon\}$. A **strict local minimizer** simply doesn't achieve equality. x^* is an **isolated local minimum** if it is a local minimum and no other local minimum are nearby. Notice that isolated implies strict but the converse is false.

Example 1.1.11 (strict but not isolated)

$$f(x) = \begin{cases} x^4 \cos\left(\frac{1}{x}\right) + 2x^4 & x \neq 0 \\ 0 & x = 0 \end{cases}$$

$x^* = 0$ is strict but not isolated due to the rapid oscillation near $x = 0$.

Notation. $f \in \mathcal{C}^3$ means f, f', f'', f''' all exist and are continuous. $f \in \mathcal{C}^3(\mathbb{R}^n)$ means $f, \nabla f, \nabla^2 f, \nabla^3 f$ all exist and are continuous.

Connections with Calculus 1

Recall that in Cal 1, we first find the stationary/critical points in the domain. Then we add the boundary points and minimize over the small (finite) set of candidates.

In high-dimension optimization, we cannot check critical points and the boundary separately because the set of points in the boundary becomes infinite. Moreover, there can be infinite critical points too.

Necessary condition: if x^* is a local or global minimizer and $C = \mathbb{R}^n$, then x^* is a **critical point**. But the converse is false.

Notation. The boundary of C is denoted as $\partial C := \overline{C} \setminus \text{int } C$.

If x^* is a critical point but is not a local or global minimizer, then it's a **saddle point**.

Theorem 1.1.12 (Weierstrass)

If f is continuous and C is compact, then f achieves its infimum over C .

That is,

$$\inf_{x \in C} f(x) = \min_{x \in C} f(x).$$

Note. This is pretty much the same as the Extreme Value Theorem.

Proof. First let's prove a claim.

Claim 1.1.13. Every compact set K is closed and bounded.

Closed: suppose not, the compact set K doesn't contain all its limit points. That is, there exists a limit point $x \notin K$ s.t. a sequence $(x_n) \subseteq K$ converges to x . But that also means that all subsequences of (x_n) converges to $x \notin K$ as well, contradicting with the definition of compactness that for every sequence in K there exists a subsequence that converges inside K .

Bounded: suppose not, for all $M > 0$, there exists a $x \in K$ s.t. $\|x\| > M$. This allows us to find a sequence $(x_n) \subseteq K$ s.t. $\|x_n\| > n$. This way every subsequence is also unbounded and cannot converge, contradicting with the definition of sequential compactness.

Now let's begin proof proper. Since C is compact and f is continuous, the image of C under f , $f(C)$, is also compact (this follows from sequential definition of continuity). By the claim $f(C)$ is bounded and closed, meaning that it has an infimum (completeness axiom) and contains the infimum (closed). Thus, f achieves its infimum over C . \square

Remark 1.1.14 It would be nice if our constraints C are compact. But at the very least, we want our constraint sets to be closed. For example, $\|Ax - b\| \leq \varepsilon$ instead of $\|Ax - b\| < \varepsilon$.

Several things to note about the feasible set C :

If $C = \emptyset$, the problem is infeasible. This is not always easy to spot.

In this class, C will usually be convex and not integral, *i.e.* \mathbb{Z}^n .

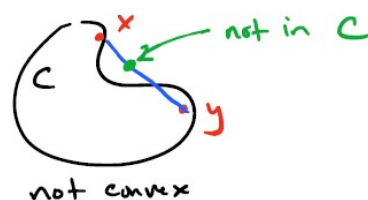
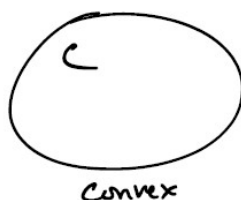
Integral constraint is problematic because the optimal integer solution might not be at all close to the optimal real solution, so we cannot obtain it by solving for the real solution first and then round it.

1.1.4 Convexity

Note. From now on we always assume the constraint set C is a subset of a vector space.

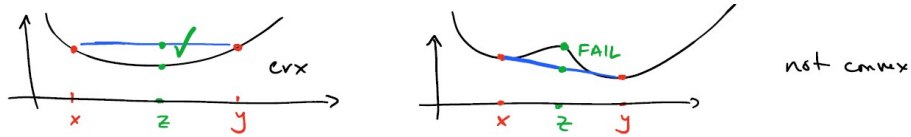
Definition 1.1.15 (convex set) — A set C is **convex** if for all $x, y \in C$ and for all $t \in [0, 1]$, then

$$tx + (1 - t)y \in C.$$



Definition 1.1.16 (convex function) — $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is a **convex function** if for all $x, y \in \mathbb{R}^n$ and $t \in [0, 1]$, then

$$f(tx + (1 - t)y) \leq tf(x) + (1 - t)f(y).$$



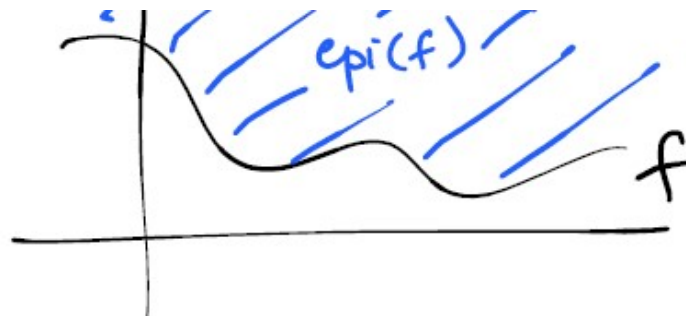
Remark 1.1.17 Linear or affine functions are both convex and concave.

Recall that a **graph** of a function is just the set of points we use to plot a function (now generalized to functions with domain of any dimension).

Definition 1.1.18 (epigraph) —

$$\text{epi}(f) = \{(x, s) : x \in \mathbb{R}^n, s \in \mathbb{R}, s \geq f(x)\}.$$

Intuition. The epigraph of f is sort of the "upper" partition of the vector space that the graph of f resides and partitions. We use epigraph to bridge the concepts of convex sets and convex functions.



Proposition 1.1.19

A function f is convex if and only if the set $\text{epi}(f)$ is convex.

Theorem 1.1.20

If f is convex and C is convex, then any local minimizer of $\min_{x \in C} f(x)$ is in fact global. The set of global solutions is also convex and in particular connected.

This is very neat for optimization!

Proof. Given a local minimizer $x^* \in C$, let's show that it is a global minimizer. Suppose not, that is, there exists a point $x \in C$ s.t. $f(x) < f(x^*)$. Since x^* is a local minimizer, there exists an $\varepsilon > 0$ s.t. $f(x^*) \leq f(y)$ for all $y \in C$ s.t. $\|y - x^*\| < \varepsilon$. Clearly $\|x^* - x\| \geq \varepsilon$ or x^* would not be a local minimizer. Choose $t < \frac{\varepsilon}{\|x^* - x\|} \in [0, 1]$. Since C is convex, we know that the point $x_0 = tx + (1 - t)x^* \in C$. Notice that

$$\begin{aligned} \|x^* - x_0\| &= \|x^* - (tx + (1 - t)x^*)\| \\ &= \|t(x^* - x)\| \\ &= t\|x^* - x\| \\ &< t \cdot \frac{\varepsilon}{t} \\ &= \varepsilon \end{aligned}$$

That is, x_0 in the ε -neighborhood of x^* and it follows that $f(x^*) \leq f(x_0)$. Since f is convex,

$$\begin{aligned} f(x_0) &= f(tx + (1 - t)x^*) \\ &\leq tf(x) + (1 - t)f(x^*) \\ &< tf(x^*) + (1 - t)f(x^*) \\ &= f(x^*) \end{aligned}$$

This contradicts with the fact that $f(x^*) \leq f(x_0)$. Hence we prove that any local minimizer x^* must also be a global minimizer.

To show that the set of global minimizers is connected, it suffices to prove that it is path-connected. The path we check is of course the line segment in the definition of convex set:

$$g(t) = ta + (1 - t)b, \quad a, b \in \operatorname{argmin}_{x \in C} f(x).$$

It's easy to see that $f(g(t)) \leq tf(a) + (1 - t)f(b) = \min_{x \in C} f(x)$ for all $t \in [0, 1]$. It follows that $f(g(t))$ must equal to the global minimum for all $t \in [0, 1]$. This makes $g(t) \in \operatorname{argmin}_{x \in C} f(x) \forall t \in [0, 1]$. Thus, the continuous function $g(t)$ is the path we seek. Since a, b are arbitrary global minimizers, we kill two birds in one stone and show that the set of global minimizers is 1. convex and 2. path-connected and therefore connected. \square

1.2 Convex Sets [BV04 Ch.2]

1.2.1 Convex, affine, and cone

Definition 1.2.1 — Let $x, y \in \mathbb{R}^n$ (or any vector space), then

- 1) $tx + (1 - t)y, t \in [0, 1]$ is a **convex combination** (of x, y).
- 2) $tx + (1 - t)y, t \in \mathbb{R}$ is a **linear combination**.
- 3) $tx + sy, t, s \geq 0$ is a **(convex) conic combination**.

Definition 1.2.2 — A set $C \subseteq \mathbb{R}^n$ is

- 1) **convex** if for all $x, y \in C$, it contains all convex combinations of x, y .
- 2) **affine** if for all $x, y \in C$, it contains all linear combinations of x, y .
- 3) a **cone** if for all $x \in C$, it contains all conic combinations of x .
- 4) a **convex cone** if it's convex and a cone. That is, for all $x, y \in C, t, s \geq 0, tx + sy \in C$.

Note. Affine implies convex based on definition.

Remark 1.2.3 An affine set/subspace is like a subspace except it is possible shifted (may not include 0). Think inhomogenous equation from differential equations. It's also analogous to cosets.

Recall from analysis, the **closure** of A , \overline{A} , is the union of A and all its limit points. We can also characterize \overline{A} as the smallest closed set containing A or equivalently the intersection of all closed sets containing A . We can do something similar here.

Definition 1.2.4 (affine hull) — The **affine hull** of C , $\text{aff}(C)$, is the smallest affine set containing C .

The **affine dimension** of C is $\dim(\text{aff}(C))$. For example, although the unit circle in \mathbb{R}^2 has dimension 1, its affine hull is all of \mathbb{R}^2 so its affine dimension is

2.

Definition 1.2.5 (convex hull) — The **convex hull** of C , $\text{conv}(C)$, is the smallest convex set containing C . It is equivalent to the set of all convex combinations of points in C :

$$\left\{ \sum_{i=1}^k t_i x_i : x_i \in C, t_i \geq 0, \sum_{i=1}^k t_i = 1 \right\}.$$

Intuition. Given an arbitrary set C , we wrap a rubber band around it and the region enclosed by the rubber band is $\text{conv}(C)$.



Definition 1.2.6 (conic hull) — The **conic hull** of C is the set of all conic combinations of points in C . That is,

$$\left\{ \sum_{i=1}^k t_i x_i : x_i \in C, t_i \geq 0 \right\}.$$

It is the smallest convex cone that contains C .

Definition 1.2.7 (relative interior) — The **relative interior** of a set C is the set

$$\text{ri}(C) = \{x \in C : \exists \varepsilon > 0, B_\varepsilon(x) \cap \text{aff}(C) \subseteq C\}$$

Note. This is really useful for studying symmetric matrices.

Example 1.2.8

Let $C = [0, 1] \subseteq \mathbb{R}$, then $\text{int}(C) = \text{ri}(C) = (0, 1)$.

However, if $C = [0, 1] \times \{0\}$ which has the same shape but is an embedding

in \mathbb{R}^2 , then $\text{int}(C) = \emptyset$ because for every $x \in C$, $B_\varepsilon(x)$ goes outside C along the second dimension. But $\text{ri}(C) = (0, 1)$ because $\text{aff}(C) = \mathbb{R} \times \{0\} \cong \mathbb{R}$.

1.2.2 Important examples

Definition 1.2.9 (hyperplane) — For $a \in \mathbb{R}^n, b \in \mathbb{R}$, the **hyperplane** would be the affine, $n - 1$ dimensional set

$$\{x \in \mathbb{R}^n : a^T x = b\}.$$

Alternatively,

$$\{x \in \mathbb{R}^n : a^T(x - x_0) = 0\}.$$

Note. Hyperplanes are convex and affine with $n - 1$ dimension. In 2D, it's a line. In 3D it's an actual plane. Also recall from cal 3 that a is the normal vector of the hyperplane.

Definition 1.2.10 (half-space) — A hyperplane partition \mathbb{R}^n into two **half-spaces**. They have the form

$$\{x \in \mathbb{R}^n : a^T x \leq b\}.$$

Note. Half spaces are convex but not affine.

Definition 1.2.11 (Euclidean ball) — Open ball: $B_\varepsilon(x) = \{y \in \mathbb{R}^n : \|y - x\| < \varepsilon\}$.

Closed ball: $\overline{B}_\varepsilon(x) = \{y \in \mathbb{R}^n, \|y - x\| \leq \varepsilon\}$.

Note. Balls are convex but not affine.

Definition 1.2.12 (ellipsoid) — An **ellipsoid** has the form

$$\mathcal{E} = \{x : (x - x_0)^T P^{-1} (x - x_0) \leq 1\}$$

for some matrix $P \succ 0$.

Notation. $A \succ 0$ in this course means A is symmetric and positive definite.

Note. Ellipsoid, like ball, is convex but not affine. If we choose $P = \varepsilon^2 I$, then we get an ε -ball.

Intuition. This is a generalization of Cal 3 ellipsoid using quadratic form. Recall that the P^{-1} in the middle of $x^T x$ is giving us a *weighted* sum. Since $y^T y = \|y\|^2 \leq 1$ is a unit ball, a weighted norm would help us transform an ellipsoid into the unit ball. We use the inverse of P in the definition because the image of this quadratic form is sort of a unit ball, but we are more interested in knowing how to go from the unit ball to the ellipsoid, and P encodes this transformation. Also since $P \succ 0$, using Spectral Theorem we can find the principle axes and length of the ellipsoid.

Example 1.2.13 (cones)

- positive orthant in \mathbb{R}^n , e.g. first quadrant in \mathbb{R}^2 .

$\mathbb{R}_+^n = \{x \in \mathbb{R}^n : x_i \geq 0\}$ is a cone.

However, $\mathbb{R}_{++}^n = \{x \in \mathbb{R}^n : x_i > 0\}$ is not a cone as a cone must include the additive identity 0 to be closed under non-negative scalar multiplication.

- Lorentz cone/2nd order cone/"ice cream cone"

$$C = \{(x, t) \in \mathbb{R}^{n+1}, x \in \mathbb{R}^n, t \in \mathbb{R} : \|x\|_2 \leq t\}.$$

- the set of positive semidefinite matrices (PSD): this is the most important nonpolyhedral cone. We assume PSDs are Hermitian and are denoted by $A \succeq 0$.

Notation. \mathbb{S}^n denotes the set of symmetric $n \times n$ matrices. Similar to the reals, we use \mathbb{S}_+^n to denote the set of symmetric positive semidefinite matrices.

Definition 1.2.14 (polyhedron) — A **polyhedron** $\mathcal{P} \subseteq \mathbb{R}^n$ is a set of the intersection of a *finite* number of half-spaces and hyperplanes. That is,

$$\mathcal{P} = \{x \in \mathbb{R}^n : a_j^T x \leq b_j, j = 1, \dots, m; c_j^T x = d_j, j = 1, \dots, p\}.$$

Note. Intersection of infinite number of half-spaces and hyperplanes are not necessarily a polyhedron (in the intuitive sense) because it can "smooth out" the edges and turn it into for example a ball, such as

$$\overline{B_1}(0) = \{x \in \mathbb{R}^n : a_j^T x \leq 1, a_j \in \text{unit circle}\}.$$

Note. Polyhedra are always convex since it's finite intersection of convex sets.

Note. The terms "polygon", "polyhedron", and "polytope" will be used interchangeably in this course.

Recall that if a set of points are *linearly independent*, then their linear combinations can equal zero only if all coefficients are zero. Moreover, in an n -dimensional vector space, there can at most be n linearly independent points.

Definition 1.2.15 (affinely independent) — A set of points $\{x_i\}_{i=0}^n$ is **affinely independent** if

$$\sum_{i=1}^n t_i (x_i - x_0) = 0 \Rightarrow t_i = 0.$$

Remark 1.2.16 It doesn't matter which x_i we choose to be x_0 . They are all equivalent. This is because an affine space can be think of as a translation of a vector space. That is, every element in the affine space is an element from a vector space offset by the same translation vector. When we subtract any two elements from the affine space, the translation vector cancels out and leaves us an element from the vector space so we go back to linear independence in the vector space. Again we can think of the solutions to inhomogeneous differential equations for a concrete example.

Remark 1.2.17 In a n -dimensional vector space, at most $n + 1$ points can be affinely independent. The "+1" comes from bringing the 0 in the vector space up to x_0 , elevating the n -dimensional vector space to an n -dimensional embedding in a $n + 1$ -dimensional vector space. For example, \mathbb{R}^n requires at least n points to fully describe it via the span. We can visualize \mathbb{R}^n as an origin-containing plane embedded in \mathbb{R}^{n+1} . If we translate \mathbb{R}^n by a vector x_0 , we get an n -dimensional affine space that now requires at least $n + 1$ points to describe.

Definition 1.2.18 (simplex) — For any set of $k + 1$ affinely independent points $\{x_i\}_{i=0}^n$ in \mathbb{R}^n , they determine a **simplex**

$$C = \text{conv}(\{x_i\}_{i=0}^k) = \left\{ x = \sum_{i=0}^k t_i x_i : t_i \geq 0, \sum_{i=0}^k t_i = 1 \right\}.$$

Note. The affine dimension of $k + 1$ point simplex is k , so we call it a k -dim simplex.

Intuition. Due to affine independence, we can think of the convex hull of the points as having "all its fat trimmed". Using the rubber band visualization, we can see why the following examples are true.

Example 1.2.19

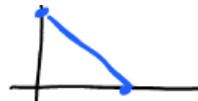
- 1-dim simplex = $\text{conv}(\{x_0, x_1\})$ is a line segment in $\mathbb{R}^n, n \geq 1$.
- 2-dim simplex = $\text{conv}(\{x_0, x_1, x_2\})$ is a filled triangle in $\mathbb{R}^n, n \geq 2$.
- 3-dim simplex is a tetrahedron in $\mathbb{R}^n, n \geq 3$. Clearly if $n = 2$, 4 points cannot be affinely independent and thus cannot generate a simplex.
- In \mathbb{R}^n , the **unit simplex** is the simplex generated by $\{0, e_1, e_2, \dots, e_n\}$, where e_i is the standard basis. It has affine dimension of n . This can be expressed as

$$\left\{ x \in \mathbb{R}^n : x \geq 0, \sum_{i=1}^n x_i \leq 1 \right\} \text{ or } \{x \in \mathbb{R}^n : x \geq 0, \mathbb{1}^T x \leq 1\}.$$

- In \mathbb{R}^n , the $(n-1)$ -dim **probability simplex** is generated by $\{e_1, e_2, \dots, e_n\}$ (basically unit simplex without 0). It can be expressed as

$$\{x \in \mathbb{R}^n : x \geq 0, \mathbb{1}^T x = 1\}.$$

The 2D unit simplex is  (in \mathbb{R}^2)

1D- probability simplex is 

Remark 1.2.20 Simplex is generated by finite points. *Atomic norm* generalizes this to infinite points and uses gauge functions for signal processing.

1.3 Operations that preserve convexity

- 1) Cartesian products: If $C_1 \subseteq \mathbb{R}^{n_1}$ and $C_2 \subseteq \mathbb{R}^{n_2}$ both convex, then $C_1 \times C_2 \subseteq \mathbb{R}^{n_1+n_2}$ is convex.
- 2) Arbitrary intersections (even uncountable): If C_1, C_2 are convex, then $C_1 \cap C_2$ is convex. *This is not true for unions.*
- 3) Image and preimage of an affine function $f(x) = Ax + b$:
 - $f(C)$ is convex if $C \subseteq \mathbb{R}^n$ is.
 - $f^{-1}(C)$ is convex if $C \subseteq \mathbb{R}^m$ is.

This implies that scaling, translation, rotation, and projection all preserve convexity.

So does **Minkowski sum**:

$$C_1 + C_2 := \{x + y : x \in C_1, y \in C_2\}.$$

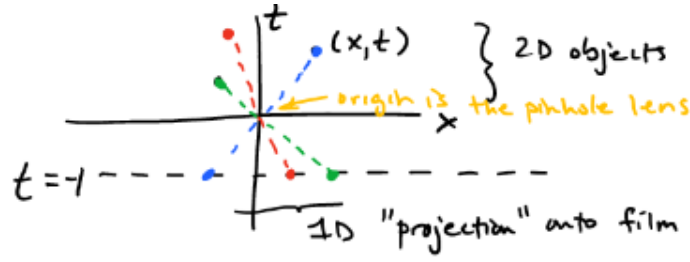
We should think of it like a convolution, where each element in C_1 is convolved with the entire C_2 .

1.3.1 Linear-fractional and perspective functions

Definition 1.3.1 (perspective function) — A **perspective function** is a function $P : \mathbb{R}^{n+1} = \mathbb{R}^n \times \mathbb{R}_{++} \rightarrow \mathbb{R}^n$ s.t. for $z \in \mathbb{R}^n$ and $t \in \mathbb{R}_{++}$,

$$P(z, t) = \frac{z}{t}.$$

Intuition. We can think of it as normalizing by t , $(\frac{z}{t}, 1)$, and then projecting to \mathbb{R}^n (or equivalently dropping the last component). Geometrically we can think of it as a "pin-hole" camera that projects 3D points in the t -positive half of \mathbb{R}^3 through the pin-hole at origin onto the 2D film at $t = -1$. This gives us $(-\frac{z}{t}, -1)$ which is the negative perspective. Then since all the points are on $t = -1$ we simply drop it.



If P is the perspective function, we can conclude that

- If $C \in \mathbb{R}^{n+1}$ is convex, then $P(C)$ is convex in \mathbb{R}^n .
- If $C = \mathbb{R}^n$ is convex, then $P^{-1}(C)$ is convex in \mathbb{R}^{n+1} .

Definition 1.3.2 (linear-fractional function) — A **linear-fractional function** is $f : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that composes the perspective P with an affine function g , where

$$g(x) = \begin{pmatrix} A \\ c^T \end{pmatrix} x + \begin{pmatrix} b \\ d \end{pmatrix} : \mathbb{R}^n \rightarrow \mathbb{R}^{m+1}, A \in \mathbb{R}^{m \times n}, b \in \mathbb{R}^m, c \in \mathbb{R}^n, d \in \mathbb{R}.$$

That is, $f = P \circ g$, and

$$f(x) = \frac{Ax + b}{c^T x + d}, \text{ with domain } \{x : c^T x + d > 0\}.$$

Note. Since the image and preimage of both affine and perspective functions preserve convexity, the image and preimage of a linear-fractional function again preserve convexity.

1.3.2 Generalized inequalities

Definition 1.3.3 (proper cone) — A cone $K \subseteq \mathbb{R}^n$ is called a **proper cone** if it satisfies the following:

- 1) convex
- 2) closed
- 3) solid or nonempty interior
- 4) pointed or contains no line or $x \in K, -x \in K \Rightarrow x = 0$.

Proposition 1.3.4

Any proper cone K induces a partial order:

$x \preceq_K y$ or simply $x \leq y$ if $y - x \in K$,

$x \prec_K y$ or simply $x < y$ if $y - x \in \text{int}(K)$.

Definition 1.3.5 (dual cone) — If K is a set, its **dual cone** is

$$K^* = \{y : \langle x, y \rangle \geq 0 \ \forall x \in K\}.$$

Note. The larger K is, the more restricted K^* becomes.

Properties of dual cones

- 1) K^* is a cone even if K isn't a cone.
- 2) K^* is convex even if K isn't convex.
- 3) $K_1 \subseteq K_2 \Rightarrow K_2^* \subseteq K_1^*$.
- 4) $K^{**} = K$ iff K is a proper cone.

Example 1.3.6

If K is a subspace, then $K^* = K^\perp$. This is because $-x \in K$ so equality is achieved in the definition.

Example 1.3.7 (self-dual)

$\mathbb{R}_+^n = (\mathbb{R}_+^n)^*$ because it is a proper cone.

Example 1.3.8 (PSD matrices)

$K = S_+^n$, and $x \in K \Rightarrow X = GG^T$ (Cholesky). Then

$$K^* = \{Y \in S^n : \langle Y, X \rangle \geq 0 \ \forall X \succeq 0\}.$$

Recall that

$$\begin{aligned}
 \langle Y, X \rangle &= \text{tr}(Y^T X) \\
 &= \text{tr}(Y X) \text{ since } Y = Y^T \\
 &= \text{tr}(Y G G^T) \\
 &= \text{tr}(G^T Y G) \text{ by cyclic property of trace}
 \end{aligned}$$

The last expression is ≥ 0 for all matrices G iff $Y \succeq 0$. Hence we show that S_+^n is self-dual.

1.3.3 separating and supporting hyperplanes

Theorem 1.3.9 (separating hyperplane)

Let C, D be convex, non-intersecting sets in \mathbb{R}^n , then there exists $a \in \mathbb{R}^n \setminus \{0\}$ and $\mu \in \mathbb{R}$ s.t.

$$\begin{aligned}
 a^T x &\leq \mu \quad \forall x \in C \\
 a^T x &\geq \mu \quad \forall x \in D
 \end{aligned}$$

Note. This reads as there exists a hyperplane that separates the two convex sets. It is clearly not true if the sets aren't convex. a is the normal to the hyperplane.

Definition 1.3.10 (Chebyshev set) — A set S is a **Chebyshev set** if for all x_0 , there exists a unique $x \in S$ s.t.

$$x = \operatorname{argmin}_{y \in S} \|y - x_0\|.$$

Note. This reads as there exists a unique best approximation point in the set S for any x_0 .

Example 1.3.11

Open unit ball isn't Chebyshev because it doesn't reach infimum.

Example 1.3.12

A nonconvex set isn't Chebyshev because there exists an x_0 where we have at least two best approximation points.

Theorem 1.3.13

Any nonempty, closed, convex set in a Hilbert space is Chebyshev.

Theorem 1.3.14 (supporting hyperplanes) (i) If C is convex, closed and

$D = \{x_0\}, x_0 \notin C$, then there exists $a \in \mathbb{R}^n$ s.t. $a^T x < a^T x_0 \forall x \in C$.

(ii) Same but C needs not be closed, $x_0 \notin \overline{C}$.

(iii) as in (ii) but allow $x_0 \in \overline{C} \setminus C$.

(i). WLOG let $x_0 = 0$ (since we can always translate C). C is Chebyshev so let y be the unique closest point to 0, and define $a = -y$ (normal of the hyperplane). We wish to show that $a^T x < a^T x_0 = 0 \forall x \in C$. That is, $y^T x > 0 \forall x \in C$.

Given $x \in C$, $y + \varepsilon(x - y) \in C$ by convexity. Since y is the best approximation point,

$$\begin{aligned} \|y\|^2 &\leq \|y + \varepsilon(x - y)\|^2 \\ &= \|y\|^2 + 2\varepsilon\langle y, x - y \rangle + \varepsilon^2\|x - y\|^2 \\ 0 &= 2\langle y, x \rangle - 2\langle y, y \rangle + \varepsilon\|x - y\|^2 \\ \langle y, x \rangle &\geq \|y\|^2 - \frac{\varepsilon}{2}\|x - y\|^2 \end{aligned}$$

Take $\varepsilon \rightarrow 0$, since $y \neq 0 \Rightarrow \|y\| > 0$, we obtain $y^T x > 0$ as required. \square

Remark 1.3.15 This is related to **Theorems of Alternatives**. Generally, they are stated as the following:

Either A is true, B is false, but not both.

Example 1.3.16 (Fredhold alternative, finite-dim)

Either $\{x : Ax = b\}$ is empty, or $\{\lambda : A^T \lambda = 0, \lambda^T b \neq 0\}$ is non-empty, but

not both.

Why do we care? To prove that there is a solution to $Ax = b$. We can simply find a solution x . This is a "certificate". But if professor asks you to prove there isn't a solution to $Ax = b$, we can try to show that A is singular, but if $b = 0$ even singular A works. Another way is to find a "certificate" λ . This is the first task of duality.

Example 1.3.17 (Farkas Lemma)

Either $\{Ax = b, x \geq 0\}$ is non-empty, or $\{\lambda : A^T \lambda \geq 0, \lambda^T b < 0\}$ is non-empty, but not both.

Theorem 1.3.18 (Theorem of Alternatives for strict linear inequalities)

The following statements are equivalent:

- (i) The set $\{x : Ax < b\}$ is empty.
- (ii) The sets $C = \{b - Ax : x \in \mathbb{R}^n\}$ and $D = \mathbb{R}_{++}^m$ do not intersect.
- (iii) The hyperplane separation theorem and its converse hold. That is,

$$\exists \lambda \geq 0 (\lambda \neq 0) \text{ s.t. } A^T \lambda = 0, \lambda^T b \leq 0.$$

Intuition. (ii) is just rephrasing (i). No intersection from (ii) can then be established by finding something that separates C, D in (iii).

Proof. (converse of hyperplane separation)

(iii) \Rightarrow (i): suppose such λ exists, and for contradiction, assume there exists x s.t. $Ax < b$. Then since $\lambda \geq 0$,

$$0 = (A^T \lambda)^T x = \lambda^T Ax < \lambda^T b.$$

So we obtain $0 < \lambda^T b \leq 0$, a contradiction.

(i) \Rightarrow (iii): By the separation theorem, we know there exists $\lambda \neq 0$ s.t.

$$\begin{aligned} \lambda^T (b - Ax) &\leq \mu, x \in \mathbb{R}^n \\ \lambda^T y &\geq \mu, y \in \mathbb{R}_{++}^n \end{aligned}$$

It follows from the first condition that $\lambda^T A x = 0$ because otherwise we can just choose a large negative x to exceed μ and get contradiction. Since this is true for all x , it must be that $\lambda^T A = A^T \lambda = 0$. From the second condition we have $\lambda \geq 0$, because otherwise if $\lambda_i < 0$, we can choose $y_i \rightarrow \infty$ to get contradiction. Moreover, we need $\mu \leq 0$ since if $\mu > 0$, we can take all components of y to 0^+ , so $\lambda^T y \rightarrow 0^+$. Then $\lambda^T (b - A^T x) \leq \mu \leq 0$ implies that $\lambda^T b \leq 0$.

Taken together, we have $\lambda \geq 0, \lambda \neq 0, A^T \lambda = 0$, and $\lambda^T b \leq 0$. \square

1.4 Convex Functions [BV04 Ch.3]

Definition 1.4.1 (convex function) — A function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is **convex** if $\text{dom}(f)$ is a convex set and for all $x, y \in \text{dom}(f)$ and $0 \leq t \leq 1$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y).$$

It is **strictly convex** if it has strict inequality. It is **strongly convex w.r.t. the norm $\|\cdot\|$ with parameter μ** if $\text{dom}(f)$ is a convex set and, for all $x, y \in \text{dom}(f), x \neq y, 0 \leq t \leq 1$,

$$f(tx + (1-t)y) \leq tf(x) + (1-t)f(y) - \frac{\mu}{2}t(1-t)\|x - y\|^2.$$

Intuition. For strongly convex, we can see that when $t = \frac{1}{2}$ or when the point is midway between x, y , the bound of inequality is a lot smaller than when t is closer to either point. This forces the function to have a large curvature.

Theorem 1.4.2 (simpler characterizations)

f is convex if $\text{epi}(f)$ is convex ($\Rightarrow \text{dom}(f)$ is convex).

f is strictly convex means it always has curvature and no straight lines.

f is strongly convex with parameter μ and w.r.t. $\|\cdot\|_2$ iff $x \mapsto f(x) - \frac{\mu}{2}\|x\|_2^2$ is convex (not true for general norms).

Note. Subtraction of convex function doesn't preserve convexity, except in the case of strongly convex w.r.t. Euclidean norm.

Remark 1.4.3 Convexity is a *global property*. This contrasts with continuity which is a local property.

Remark 1.4.4 In convex analysis, we allow the *extended value function* $f : \mathcal{H} \rightarrow [-\infty, \infty]$ or $f : \mathcal{H} \rightarrow (-\infty, \infty]$, where \mathcal{H} is a generic Hilbert space.

This way, if $x \in \text{dom}(f)$, we can pretend it is but define $f(x) = +\infty$. This wouldn't affect our minimization problem. Now we can redefine

$$\text{dom}(f) = \{x : f(x) < +\infty\}.$$

This will turn out to be convenient for minimization.

Example 1.4.5

Define the *indicator function* of a set C to be

$$I_C(x) = \begin{cases} 0, & x \in C \\ +\infty, & x \notin C \end{cases}$$

This is different than how we usually define indicator function. Now we can do the following:

$$\min_{x \in C} f(x) \Leftrightarrow \min_{x \in \mathbb{R}^n} f(x) + I_C(x).$$

That is, we can turn a constrained minimization problem into an unconstrained problem with huge penalty on going outside the constraint.

Definition 1.4.6 (proper function) — $f : \mathbb{R}^n \rightarrow [-\infty, \infty]$ is **proper** if

- 1) it never takes the value $-\infty$.
- 2) $\text{dom}(f) \neq \emptyset$. That is, the value doesn't always equal to $+\infty$.

Note. This way we can assume there exist feasible points, hence "proper".

Example 1.4.7

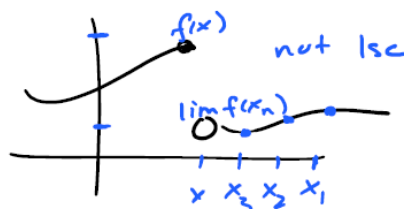
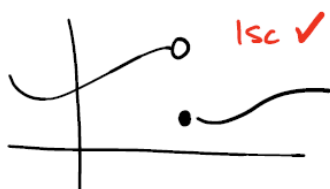
I_C is proper iff $C \neq \emptyset$.

Definition 1.4.8 (lower semi-continuous (lsc)) — $f : \mathbb{R}^n \rightarrow [-\infty, \infty]$ is **lower semi-continuous (lsc)** at $x \in \mathbb{R}^n$ if for all (x_n) s.t. $x_n \rightarrow x$,

$$f(x) \leq \liminf_n f(x_n) := \lim_{n \rightarrow \infty} \inf_{k \leq n} f(x_k).$$

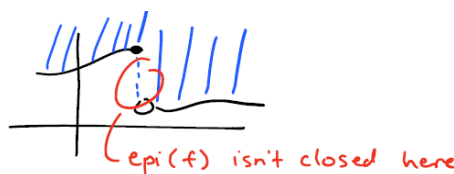
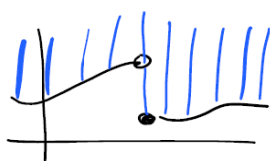
Intuition. This is a lot like sequential definition of continuity, except that we allow liminf to go higher than the function.

Definition 1.4.9 (lsc function) — f is a **lsc function** if f is lsc for all points $x \in \mathbb{R}^n$.



Theorem 1.4.10

In \mathbb{R}^n or any Hausdorff space, f is lsc iff $\text{epi}(f)$ is a closed set.



Example 1.4.11

I_C is lsc iff C is a closed set.

We can extend classical theorems involving continuity such as the Extreme Value Theorem to lsc.

Theorem 1.4.12

If C is compact, then f is lsc $\Rightarrow f$ achieves its minimum over C .

Remark 1.4.13 f is continuous iff f is lsc and usc.

Notation. $\Gamma(\mathbb{R}^n)$ is the set of all lsc and convex functions $f : \mathbb{R}^n \rightarrow [-\infty, \infty]$.

$\Gamma_0(\mathbb{R}^n) \subseteq \Gamma(\mathbb{R}^n)$ consists of such functions that is also proper, $f : \mathbb{R}^n \rightarrow (-\infty, \infty]$.

So $\Gamma_0(\mathbb{R}^n)$ is the standard class of functions for convex optimization.

Example 1.4.14

$I_C \in \Gamma_0(\mathbb{R}^n)$ for some $C \subseteq \mathbb{R}^n$ iff C is nonempty, closed, and convex.

Remark 1.4.15 Recall that the restriction to proper function is mild.

What about restricting to lsc functions? It's also mild in the context of convex functions, because "weird things with convex functions can only involve boundaries (and $+\infty$).

Theorem 1.4.16 (8.38 BC17)

If $f : \mathcal{H} \rightarrow (-\infty, \infty]$ is proper and convex, then f is continuous at $x \in \text{dom}(f)$ iff f is bounded above on a neighborhood of x .

Note. For convex functions, we won't see jumps like in the lsc case.

Corollary 1.4.17 (8.39)

Given the same setup, if one of the following is true:

- (i) f is bounded above on some neighborhood of x .
- (ii) f is lsc.
- (iii) \mathcal{H} is finite dimensional.

then f is continuous on the interior of its domain, $\text{int}(\text{dom}(f))$.

Note. Under these assumptions, weird discontinuous things can only happen at the boundary.

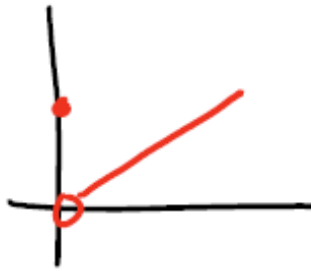


Figure 1.1: An example of a proper, convex function (not lsc) that isn't continuous due to discontinuity at the boundary. It is however continuous on the interior.

Remark 1.4.18 In summary, by the corollary if $f : \mathbb{R}^n \rightarrow \mathbb{R}$ (has full-domain and not equal to $\pm\infty$), then $\text{convex} \Rightarrow \text{continuous}$.

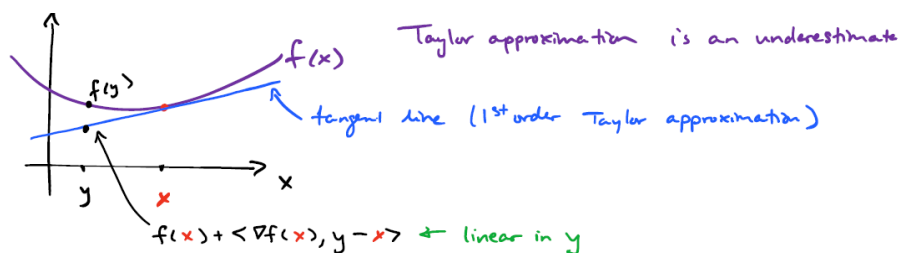
1.4.1 First-order conditions

Theorem 1.4.19

If $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is differentiable on $\text{dom}(f)$ and if $\text{dom}(f)$ is open and convex, then f is convex iff for all $x, y \in \text{dom}(f)$,

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle.$$

Note. This is the 1st order Taylor approximation (tangent line). The line is supporting the epigraph of f .



Theorem 1.4.20

Under the same assumption, f is convex iff ∇f is monotone. That is, for all $x, y \in \text{dom}(x)$,

$$\langle x - y, \nabla f(x) - \nabla f(y) \rangle \geq 0.$$

Intuition. Recall in 1D, f is convex if slope is non-decreasing. That is, if $x - y \geq 0$, then $f'(x) - f'(y) \geq 0$ and if $x - y \leq 0$ then $f'(x) - f'(y) \leq 0$. A concise way to express that is $(x - y)(f'(x) - f'(y)) \geq 0$. Here we generalize this to higher dimensions.

Theorem 1.4.21 (2nd-order condition)

$f : \mathbb{R}^n \rightarrow \mathbb{R}$. If the Hessian $\nabla^2 f(x)$ exists for all $x \in \text{dom}(f)$, then

- a) f is convex iff $\nabla^2 f(x) \succeq 0 \forall x \in \text{dom}(f)$.
- b) f is μ -strongly convex (w.r.t. $\|\cdot\|_2$) iff $\nabla^2 f(x) \succeq \mu I$.

If $\nabla^2 f(x) \succ 0$, then f is **strictly convex**.

Remark 1.4.22 f can be convex but $\nabla f, \nabla^2 f$ need not exist!

What if f isn't differentiable?

Definition 1.4.23 (subdifferential) — Let $f : \mathbb{R}^n \rightarrow (-\infty, \infty]$ be proper, then we define the **subdifferential** of f at x to be

$$\partial f(x) = \{d \in \mathbb{R}^n : \forall y \in \mathbb{R}^n, f(y) \geq f(x) + \langle d, y - x \rangle\}.$$

Note. d here is called a **subgradient**.

Theorem 1.4.24

If f is proper and convex then

$$x \in \text{ri}(\text{dom}(f)) \Rightarrow \partial f(x) \neq \emptyset.$$

Note. The proof is related to separating/supporting hyperplanes.

Proposition 1.4.25

$\partial f(x)$ is a singleton iff f is differentiable at x .

Example 1.4.26

$f(x) = |x|$. Then if $x \neq 0$, $f'(x) = \text{sgn}(x)$ and $\partial f(x) = \{f'(x)\}$. If $x = 0$, $f'(0)$ DNE. But $\partial f(0) = [-1, 1]$.

Theorem 1.4.27 (Fermat's Rule)

If f is a proper function, then

$$\underset{x}{\text{argmin}} f(x) = \{x : 0 \in \partial f(x)\}.$$

Proof. This just means that we can plug 0 into the definition of subdifferential and get

$$f(y) \geq f(x) + \langle 0, y - x \rangle = f(x) \quad \forall y.$$

This clearly shows that x is a global minimizer. □

Note. This generalizes the calculus idea of critical points for smooth functions.

Remark 1.4.28 Subdifferentials are a global notion (for all y) whereas gradients are a local notion. How do we reconcile that subdifferential can be the gradient? The answer is that the global property of convexity links the two.

Remark 1.4.29 So all we need to do is to invert ∂f . That is,

$$\operatorname{argmin} f(x) = \partial f^{-1}.$$

In fact, this is usually not practical or even possible especially for interesting problems. It may be possible for subproblems.

Definition 1.4.30 (normal cone) — The **normal cone** to a set C at point x is

$$N_C(x) = \begin{cases} \{d : \langle d, y - x \rangle \leq 0 \ \forall y \in C\} & \text{if } x \in C \\ \emptyset & \text{if } x \notin C \end{cases}$$

Example 1.4.31

Let $C \neq \emptyset$ be convex, so I_C is a proper convex function. Then $\partial I_C = N_C$.

Example 1.4.32

$x \in \operatorname{int} C \Rightarrow N_C(x) = \{0\}$. Why? WLOG, shift C so $x = 0$. If $\langle d, y \rangle \leq 0 \ \forall y \in C$. Then $x \in \operatorname{int} C \Rightarrow$ we can choose $y = \varepsilon d \in C$ for sufficiently small $\varepsilon > 0$. Then $\varepsilon \|d\|^2 \leq 0 \Rightarrow d = 0$.

Example 1.4.33

$x \in \partial C$ (the boundary). We want d s.t. $\langle d, y \rangle \leq 0 \ \forall y \in C$. Geometrically this means we want the angle between d, y to be perpendicular or obtuse. If the boundary is smooth, since d needs to be at least perpendicular to any y immediately to the left and right of x , it must be the normal ray of the tangent plane.

Example 1.4.34

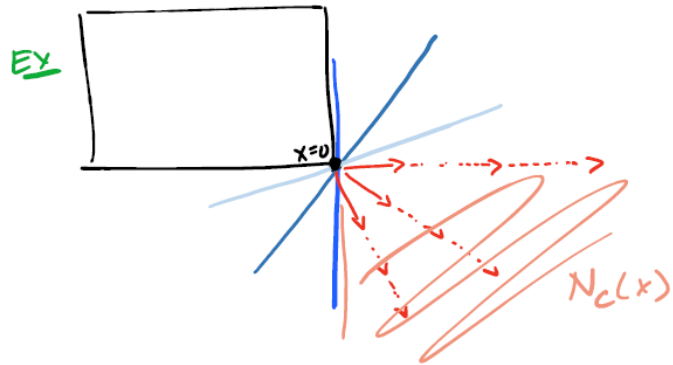


Figure 1.2: The normal cone at non-smooth boundary looks indeed like a cone.

Remark 1.4.35 An equivalent definition of normal cone is the set of all vectors that define a supporting hyperplane to C , passing through x .

Example 1.4.36

If C is a vector space, since C is closed under inverses, if we use $-y$ in addition to y in the definition we will get an equality which implies orthogonality. Hence

$$N_C(x) = \begin{cases} C^\perp & x \in C \\ \emptyset & x \notin C \end{cases}$$

Proposition 1.4.37 (6.47 BC17)

If $C \neq \emptyset$ is closed and convex, then $x = P_C(y)$ iff $y - x \in N_C(x)$, where $P_C(y)$ denotes the orthogonal projection of y onto C .

1.4.2 Calculus

Remark 1.4.38 Calculus is a set of rules we can use to calculate.

One such rule is that derivatives/gradients are linear.

Is it true that $\partial(f+g) = \partial f + \partial g$, where $+$ is the Minkowski sum? No! Although it's often true.

Example 1.4.39

$f = I_C, g = I_D \in \mathbb{R}^2$.



Then $\partial(f+g)(x) = \partial f(x) \partial g(x)$ for all x except at $x = 0$. At $x = 0$, recall that

$$\partial f(0) = N_C(0) = \mathbb{R}_+ \times \{0\}$$

$$\partial g(0) = N_D(0) = \mathbb{R}_- \times \{0\}$$

So $\partial f(0) + \partial g(0) = \mathbb{R} \times \{0\}$ But

$$\partial(f+g)(0) = N_{C \cap D}(0) \quad \text{by def of indicator}$$

$$= N_0$$

$$= \{d : \langle d, y - 0 \rangle \leq 0 \ \forall y \in \{0\}\} \quad \text{vacuous constraint}$$

$$= \mathbb{R}^2$$

We can see this counterexample is somewhat contrived, so linearity is often true.

Remark 1.4.40 Sufficient conditions to guarantee when this linearity is true are called **constraint qualifications (CQ)**.

Corollary 1.4.41 (16.48 (iv) BC17)

If $f, g \in \Gamma_0(\mathcal{H})$, and $\mathcal{H} = \mathbb{R}^n$, and one of the following holds:

(i) $\text{ri}(\text{dom}(f)) \cap \text{ri}(\text{dom}(g)) \neq \emptyset$.

(ii)

$\text{dom}(f) \cap \text{int}(\text{dom}(g)) \neq \emptyset$.

(iii) either f or g has full domain (all of \mathbb{R}^n).

Note. (iii) is most commonly used.

Since the previous example didn't satisfy a CQ, the linearity didn't hold. That is, $\text{dom } f = C, \text{dom } g = D, \text{int } C \cap \text{int } D = \emptyset$.

Remark 1.4.42 There are other cones including **tangent, polar, recession/asymptotic, and barrier cones**.

1.4.3 Lipschitz gradient

An easier way to show F is Lipschitz-continuous: if F' exists, then $\|F'\| \leq L \Rightarrow F$ is Lipschitz continuous (by the definition of derivative/Jacobian and some manipulation).

Notation. $\|\cdot\|$ denotes the appropriate operator norm, usually spectral norm if the original norm is Euclidean.

Remark 1.4.43 In optimization, "Jacobian" is often confusing, since it's unclear what F is. Of the objective function or of the gradient? Instead we prefer to say the Jacobian of the objective is the gradient (transposed). The Jacobian of the gradient is the Hessian.

Remark 1.4.44 The Hessian can be thought of as a bilinear operator $\langle d, \nabla^2 f(x) d \rangle$

Theorem 1.4.45

Suppose convex $f \in \mathcal{C}^2(U)$ for some open set $U \subseteq \mathbb{R}^n$, then

$$\nabla f \text{ is } L\text{-Lipschitz continuous on } U \Leftrightarrow \forall x \in U, \nabla^2 f(x) \preceq LI.$$

That is, all eigenvalues of $\nabla^2 f(x) \leq L \Rightarrow \|\nabla^2 f(x)\| \leq L$.

Theorem 1.4.46

Same setup, then

$$f \text{ is } \mu\text{-strongly convex on } U \Leftrightarrow \forall x \in U, \mu I \preceq \nabla^2 f(x).$$

Note. We assume $\mu > 0$ since $\mu = 0$ would give us plain old convexity.

Remark 1.4.47 One of our common assumption will be ∇f is L -Lipschitz continuous ($\nabla^2 f \preceq LI$) and a bit less common, also assume strong convexity ($\mu I \preceq \nabla^2 f$).

Example 1.4.48 (best function ever)

Consider $f(x) = \frac{1}{2}\|x\|_2^2$, $\nabla f(x) = x$, $\nabla^2 f(x) = I$. So $L = 1, \mu = 1$. This is the only function with this property.

This is the nicest function ever for optimization!

Definition 1.4.49 (condition number) — The **condition number** of f is $k_f = \frac{L}{\mu}$. $k_f \approx 1$ is good. Larger is bad.

Why do we care about these assumptions?

Recall from calculus, Taylor's theorem states that

$$f(y) = f(x) + f'(x)(y - x) + \frac{1}{2}f''(\xi)(y - x)^2,$$

where $\xi \in [x, y]$. If $f''(\xi) \leq L \forall \xi$, then

$$f(y) \leq f(x) + f'(x)(y - x) + \frac{L}{2}(y - x)^2.$$

Theorem 1.4.50

If ∇f is L -Lipschitz continuous and f is μ -strongly convex, then for all $x, y \in \text{dom}(f)$,

$$\frac{\mu}{2}\|y - x\|^2 \leq f(y) - (f(x) + \langle \nabla f(x), y - x \rangle) \leq \frac{L}{2}\|y - x\|^2.$$

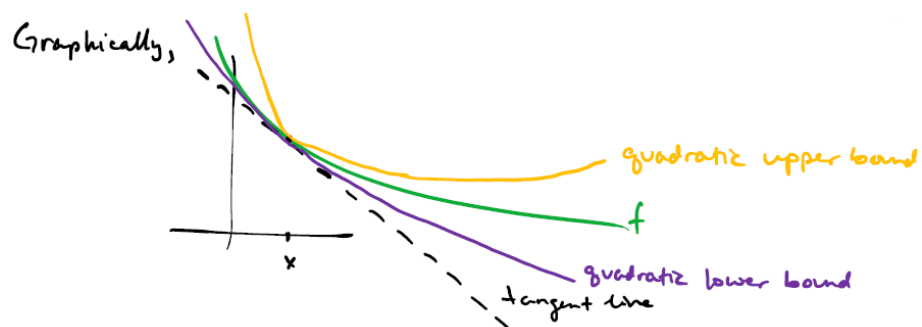


Figure 1.3: If f is complicated but we can "sandwich" it between a quadratic upper bound and a quadratic lower bound ($\mu > 0$) or a linear lower bound ($\mu = 0$), then we can work with the quadratics to understand the behavior of f since quadratics are much easier to deal with.

See more properties from this section in the Github handout [StrongConvexityLipschitz.pdf](#).

1.4.4 Examples [BV04 Ch.3.1.5]

Examples of convex functions $f : \mathbb{R} \rightarrow \mathbb{R}$:

- $e^{ax}, a \in \mathbb{R}$.
- x^a on $x \in \mathbb{R}_{++}$ if $a \leq 0$ or $a \geq 1$. (It's concave on $0 \leq a \leq 1$).
- $|x|^a$ on all of \mathbb{R} , if $a \geq 1$.
- $-\log_b(x)$ on \mathbb{R}_{++} if $b > 1$.
- On \mathbb{R}^+ ,

$$\begin{cases} x \cdot \log(x) & x > 0 \\ 0 & x = 0 \end{cases}$$

since $f''(x) = \frac{1}{x} > 0$.

Examples of convex functions $f : \mathbb{R}^n \rightarrow \mathbb{R}$:

- any norm/seminorm (follows directly from triangle inequality).
- $f(x) = \max\{x_1, \dots, x_n\}$.
- $f(x, y) = x^2/y$, $\text{dom}(f) = \mathbb{R} \times \mathbb{R}_{++}$. "Quadratic over linear".

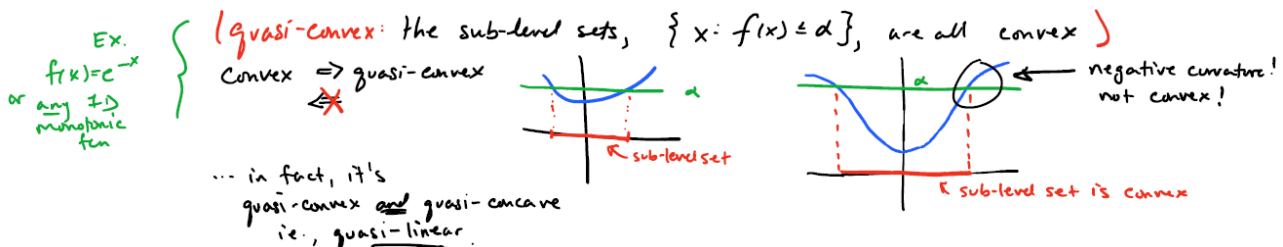
$$f(x, y) = \|x\|_2^2/y, \text{ dom}(f) = \mathbb{R}^{n-1} \times \mathbb{R}_{++}.$$

$$f(x, Y) = x^T Y^{-1} x, \text{ dom}(f) = \mathbb{R}^n \times S_{++}^n. \text{ "Matrix fractional function".}$$

Note. "Linear fractional function"

$$g(x) = \frac{Ax + b}{c^T x + d}, \quad \text{dom}(g) = \{x : c^T x + d > 0\}$$

is not convex but it is **quasi-convex**. It is defined by having all convex sub-level sets $\{x : f(x) \leq \alpha\}$.



- "log-sum exp" aka "soft-max"

$$f(x) = \frac{1}{\alpha} \log(e^{\alpha x_1} + \dots + e^{\alpha x_n}), \alpha > 0.$$

This is differentiable but needs to be careful about numerical under/overflow.

- geometric mean $f(x) = (\prod_{i=1}^n x_i)^{\frac{1}{n}}$ on \mathbb{R}_{++}^n .
- $-\log \det(X) = -\log(\prod \lambda_i) = -\sum \log(\lambda_i)$ on S_{++}^n .

Theorem 1.4.51 (Jensen's Inequality)

$$f(E[x]) \leq E[f(x)].$$

Remark 1.4.52 Let X be a random variable that outputs points in $\text{dom}(f)$ with probability in $[0, 1]$, then the inequality follows from definition of convex function.

Example 1.4.53

In machine learning, we often prove something like

$$E[\|\text{error}\|^2] \leq \varepsilon.$$

Let $f(x) = x^2$. So by Jensen's inequality:

$$\begin{aligned} (E[\|\text{error}\|])^2 &\leq E[\|\text{error}\|^2] \leq \varepsilon \\ E[\|\text{error}\|] &\leq \sqrt{E[\|\text{error}\|^2]} \leq \sqrt{\varepsilon} \end{aligned}$$

Recall that $\|\text{error}\|^2$ is the nicest function ever.

Remark 1.4.54 Hölder's inequality/Cauchy-Schwarz can also be proved via Jensen.

Theorem 1.4.55 (Hölder's inequality)

If $\frac{1}{p} + \frac{1}{q} = 1$,

$$|\langle x, y \rangle| \leq \|x\|_p \cdot \|y\|_q.$$

Remark 1.4.56 We can use Jensen's to prove Holder inequality.

1.4.5 Preserving convexity

Rule 0: non-negative (weighted) sums

If f_1, \dots, f_m are convex, $\alpha_i \geq 0$, then $x \mapsto \sum \alpha_i f_i(x)$ is convex too.

Subtraction (negative weights) doesn't work.

It works for integrals too:

If for all y , $f(\cdot, y)$ is convex, and $w(y) \geq 0$. Then

$$x \mapsto \int_{\Omega} f(x, y) w(y) dy$$

is convex.

Rule 1: perspective function

Definition 1.4.57 (perspective) — Let $f : \mathbb{R}^n \rightarrow \mathbb{R}$, then its **perspective** is $g : \mathbb{R}^{n+1} \rightarrow \mathbb{R}$,

$$g(x, t) = t \cdot f\left(\frac{x}{t}\right), \quad \text{dom}(g) = \{(x, t) : x/t \in \text{dom}(f), t > 0\}.$$

Proposition 1.4.58

$f : \mathbb{R}^n \rightarrow \mathbb{R}$ is convex \Rightarrow its perspective is convex.

Example 1.4.59

$f(x) = \|x\|^2$ is convex. Its perspective is

$$t \cdot \left\| \frac{x}{t} \right\|^2 = t \cdot \frac{\|x\|^2}{t^2} = \frac{\|x\|^2}{t}.$$

This is the quadratic-over-linear example we saw earlier. This is the proof that it is convex.

Example 1.4.60

$f(x) = -\log(x)$ is convex. Its perspective is

$$-t \cdot \log\left(\frac{x}{t}\right) = t \cdot \log(t) - t \cdot \log(x), x, t > 0.$$

This is the **relative entropy** of t, x . More generally, the **Kullback-Leibler divergence** is

$$D_{KL}(u, v) = \sum_{i=1}^n u_i \log\left(\frac{u_i}{v_i}\right) - u_i + v_i.$$

This is an example of **Bregman Divergence**, which we often use to measure "distance" as an alternative to metric. It's especially good for probability distributions.

Rule 2: special types of compositions

Composition of convex functions typically doesn't preserve convexity!

Theorem 1.4.61

f is convex if

- (i) h is convex and
- (ii) if $k = 1$, h is nondecreasing and g is convex or h is nonincreasing and g is concave.
- (iii) if $k > 1$, we enforce (ii) to each argument of h and each g_i .

Note. For nonincreasing/decreasing, we must take into account $\pm\infty$, since in convex analysis we assign infinity to any point not in the domain. So although $h(x) = x$ is nondecreasing on \mathbb{R} , if we restrict $\text{dom}(h) = [0, 1]$ then it is not nondecreasing anymore.

Theorem 1.4.62 (tattoo-worthy)

$f = h \circ g$ is convex if h is convex and g is affine.

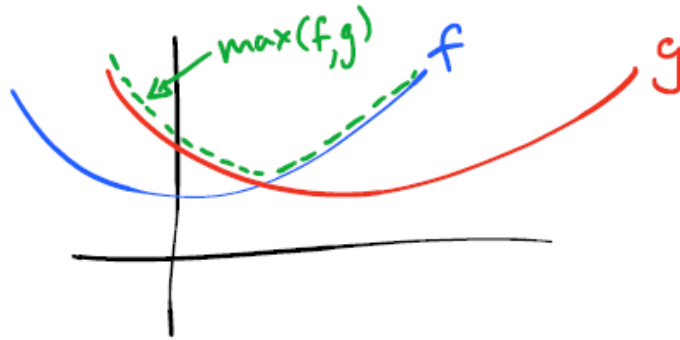
Example 1.4.63

$f(x) = \|Ax - b\|^2$ is convex by this theorem.

Rule 3: min/max**Proposition 1.4.64**

If f, g both convex, then $x \mapsto \max\{f(x), g(x)\}$ is convex.

Proof. The epigraph of the maximum is the intersection of two convex epigraphs. Convex sets are closed under arbitrary intersections. \square



Note. This works for supremum too due to closure under arbitrary intersections.

Example 1.4.65

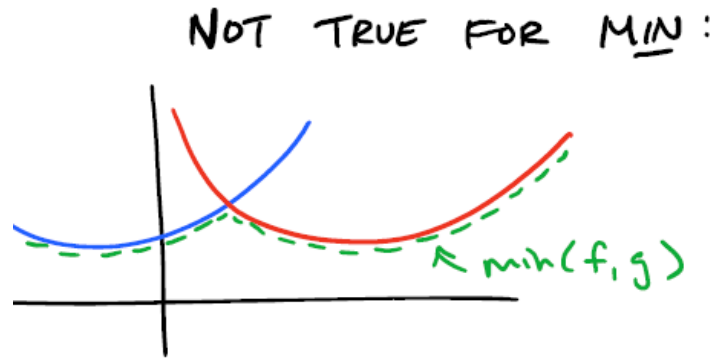
$$f(x) = \sup_{y \in \mathcal{A}} f(x; y)$$

is convex as long as $f(\cdot; y)$ is convex $\forall y \in \mathcal{A}$, where \mathcal{A} is an arbitrary set that can be uncountable.

Example 1.4.66 (spectral norm)

$$f(A) = \|A\|_{\infty} = \sup_{\|x\|_2=1} \|Ax\|_2$$

is convex since $\forall x, A \mapsto \|Ax\|_2$ is convex (composition of convex and affine).



It's easy to see that min doesn't necessarily preserve convexity because it unions epigraphs instead. We need to impose more restrictions to make it work:

Theorem 1.4.67

If $f : \mathbb{R}^n \times \mathbb{R}^m$ is (jointly) convex and if $C \neq \emptyset$ is a convex set, then

$$g(x) = \inf_{y \in C} f(x, y) \text{ is convex.}$$

Example 1.4.68

$\min\{f_1(x), f_2(x)\}$ is not usually convex since this is like taking

$$f(x, y) = \begin{cases} f_1(x), & y = 1 \\ f_2(x), & y = 2 \end{cases}$$

and constraint $C = \{1, 2\}$ is not convex.

Example 1.4.69

The distance to a convex set is a convex function. Let $C \neq \emptyset$ be convex,

$$f(x) = \inf_{y \in C} \|x - y\|.$$

Prove $(x, y) \mapsto \|x - y\|$ is convex.

Proof. We know $z \mapsto \|z\|$ is convex. Consider the linear operator $A(x, y) = x - y$. That is,

$$A \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} I & -I \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = x - y.$$

Then the composition of convex and affine is still convex. \square

1.4.6 Conjugate functions

Definition 1.4.70 (Fenchel-Legendre conjugate) — The **F.L. conjugate** of f is

$$f^*(y) = \sup_x \{\langle y, x \rangle - f(x)\}.$$

Note. For matrix inputs, use $\text{tr}(Y^T X)$.

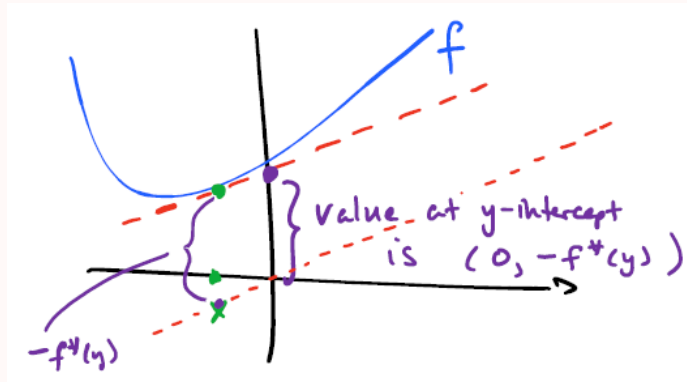
Proposition 1.4.71

f^* is always convex (whether f is or not).

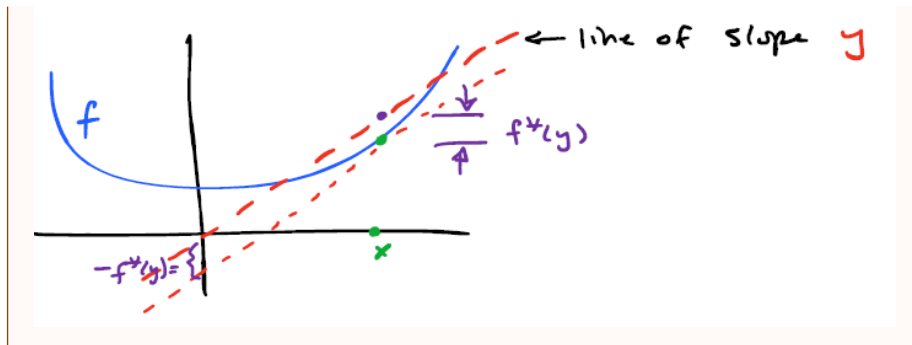
Proof. $y \mapsto \langle y, x \rangle - f(x)$ is an affine function of y which is convex, and supremum preserves convexity. \square

Example 1.4.72 (1D Legendre Transform)

Assume f is strictly convex (so f' is strictly monotone/invertible). So $f^*(y)$ is maximized (unique global) when $0 = y - f'(x) \Rightarrow f'(x) = y \Rightarrow x = (f')^{-1}(y)$. We can interpret y as the slope of $f(x)$.



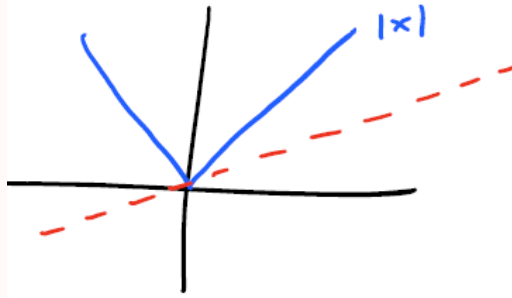
Alternatively, we can think of $f^*(y)$ as finding the x that maximize the signed separation of the "line: $\langle y, x \rangle$ and f , where the line is on top of f . Then $f^*(y)$ would be the maximized distance.



Remark 1.4.73 We can generalize the intuition from 1D example, where we think of y as the slope of $f(x)$ and think of $f^*(y)$ as the (negative) value of the intercept, to arbitrary dimensions. For a closed, strictly convex function f , we know its epigraph has a supporting hyperplane at every x . This yields a collection of affine minorants of f , in the form $f(x) \geq \langle y, x \rangle - b$ for every x . The conjugate function $f^*(y)$ says that the supporting hyperplane with slope y has the (negative) intercept $b = f^*(y)$ because to be a supporting hyperplane, the supremum difference between the affine minorant $\langle y, x \rangle - b$ and $f(x)$ must be zero. Now the affine minorants are of the form $\langle y, x \rangle - f^*(y)$, we can recover the original function by taking the pointwise supremum of this collection of affine minorants. It's equivalent to taking the intersection of all supporting hyperplanes to recover the epigraph of f . Hence $f^{**} = f$. This is the essence of duality here. For more details, please see math.stackexchange.com/questions/223235/please-explain-the-intuition-behind-the-dual-problem-in-optimization.

Example 1.4.74

$f(x) = |x|$. What is $f^*\left(\frac{1}{2}\right)$?



$f^*\left(\frac{1}{2}\right) = 0$ because the line is always below f except at 0.

What is $f^*(2)$? Answer: $x \rightarrow \infty$.

This is in fact an indicator function of the set $[-1, 1]$!

Rules

- 1) Affine transformations: let $g(x) = f(Ax+b)$, A invertible. Let $z = Ax+b$, $A^{-*} := (A^{-1})^*$, then

$$\begin{aligned}
 g^*(y) &= \sup_x \langle y, x \rangle - f(Ax+b) \\
 &= \sup_z \langle y, A^{-1}(z-b) \rangle - f(z) \\
 &= -\langle y, A^{-1}b \rangle + \sup_z \langle A^{-*}y, z \rangle - f(z) \\
 &= \langle -y, A^{-1}b \rangle + f^*(A^{-*}y)
 \end{aligned}$$

- 2) Sums of functions: Let $f(x) = f_1(x) + f_2(x)$. Is $f^*(x) = f_1^*(x) + f_2^*(x)$?
NO in general!

But it works if the f_i are independent/separable. That is, if $f(u, v) = f_1(u) + f_2(v)$, then $f^*(w, z) = f_1^*(w) + f_2^*(z)$.

Example 1.4.75

$$f(x) = I_C.$$

$$\begin{aligned}
 f^*(y) &= \sup_x \langle y, x \rangle - I_C(x) \\
 &= \sup_{x \in C} \langle y, x \rangle \quad -\infty \text{ for } \sup \text{ is equivalent to } \infty \text{ for } \inf
 \end{aligned}$$

This is called the **support function** of C .

Remark 1.4.76 It is not hard to see that support function is related to supporting hyperplane. HOW? TODO

Question 1.4.77. Let $C = \{x : \|x\|_p \leq 1\}$. What is the support function of this set?

Per the definition, the support function is

$$\begin{aligned} f^*(y) &= \sup_{\|x\|_p \leq 1} \langle y, x \rangle \\ &= \|y\|_q \end{aligned} \quad \text{Holder's inequality}$$

Thus, the conjugate of the indicator function of a norm ball is the dual norm.

Properties

By definition of supremum,

$$f^*(y) \geq \langle x, y \rangle - f(x) \quad \forall x.$$

Rearranging gives us

Theorem 1.4.78 (Fenchel-Young inequality)

$$f(x) + f^*(y) \geq \langle x, y \rangle \quad \forall x, y.$$

Corollary 1.4.79 (16.23 BC17)

$$y \in \partial f(x) \Leftrightarrow f(x) + f^*(y) = \langle x, y \rangle.$$

Corollary 1.4.80 (16.24)

$$f \in \Gamma_0(\mathbb{R}^n) \Rightarrow (\partial f)^{-1} = \partial f^*.$$

Remark 1.4.81 This comes from symmetry: $x \in \partial f^*(y) \Leftrightarrow f(x) + f^*(y) = \langle x, y \rangle \Leftrightarrow y \in \partial f(x)$.

Corollary 1.4.82

$$0 \in \partial f(x) \Leftrightarrow \partial f^*(0).$$

Proposition 1.4.83

$$f \in \Gamma_0(\mathbb{R}^n) \Leftrightarrow f = f^{**}.$$

See [Remark 1.4.73](#) for intuition of these properties.

Question 1.4.84. Is it possible that $f = f^{**}$?

Yes for exactly one function: $f(x) = \frac{1}{2}\|x\|_2^2$.

Question 1.4.85. What if f isn't convex?

f^{**} is convex even when f isn't. So we can "approximate" f using f^{**} . This is one way to do *convex relaxation*.

Definition 1.4.86 — The **lsc convex envelope** \check{f} of f is

$$\check{f} = \sup \{g \in \Gamma(\mathbb{R}^n) : g \leq f\}.$$

It follows that

$$\text{epi}(\check{f}) = \overline{\text{conv}(\text{epi}(f))}.$$

The problem with \check{f} is that just like the convex hull, it's usually too abstract to be useful. It turns out that often $f^{**} = \check{f}$ and is easier to find.

Example 1.4.87

Let $f = h + g$ where h is nonconvex and g is convex. To minimize f , our first thought might be to minimize $\check{f} = f^{**}$ instead, but it is usually hard

to find. We can relax it further by minimizing $h^{**} + g$. Even though this relaxation doesn't necessarily give us the same minimizer as the original, it is often good enough. Let's look at an example.

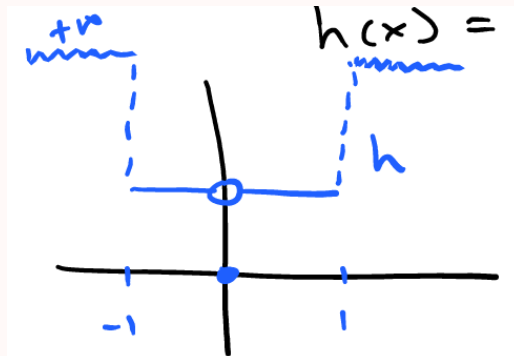
Example 1.4.88

(compressed sensing) In this problem we have the primal problem

$$\begin{aligned} \min \quad & \|x\|_0 \\ \text{subject to} \quad & \|x\|_\infty \leq 1 \\ & Ax = b \end{aligned}$$

This is a highly non-convex problem due to ℓ_0 norm. Let's perform convex relaxation on it. First define $h(x) = \|x\|_0 + I_{\|x\|_\infty \leq 1}$. We see that h is separable so we only consider the scalar case:

$$h(x) = \begin{cases} 0 & x = 0 \\ 1 & x \neq 0, |x| \leq 1 \\ \infty & |x| > 1 \end{cases}$$



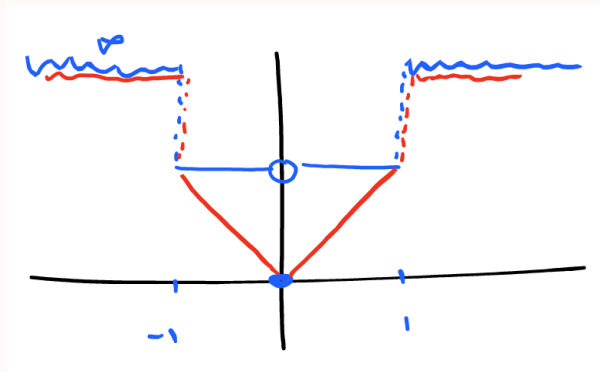
Then simply by eyeballing, we see that

$$h^{**}(x) = |x| + I_{|x| \leq 1}$$

This is just ℓ_1 norm plus the indicator function! Thus the convex relaxation

of the primal problem is

$$\begin{aligned} \min \quad & \|x\|_1 \\ \text{subject to} \quad & \|x\|_\infty \leq 1 \\ & Ax = b \end{aligned}$$



1.4.7 Existence and uniqueness of minimizers

Existence

Intuition. We want to generalize EVT to functions in $\Gamma_0(\mathbb{R}^n)$.

Definition 1.4.89 (coercive) — $f : \mathbb{R}^n \rightarrow [-\infty, \infty]$ is **coercive** if

$$\lim_{\|x\| \rightarrow \infty} f(x) = \infty.$$

Note. In other words, it grows!

Proposition 1.4.90

f is coercive iff all sub-level sets $\{x : f(x) \leq \alpha\}$ are bounded.

Intuition. There is no "horizontal asymptote".

Proposition 1.4.91

If $f \in \Gamma_0(\mathbb{R}^n)$, f is coercive iff there exists α s.t. $\{x : f(x) \leq \alpha\}$ is non-empty and bounded.

Remark 1.4.92 Convex + coercive need not be coercive.

Theorem 1.4.93 (existence)

Let C be closed, convex, $f \in \Gamma_0(\mathbb{R}^n)$, $C \cap \text{dom}(f) \neq \emptyset$, then $\min_{x \in C} f(x)$ exists if either

- 1) f is coercive or
- 2) C is bounded.

Uniqueness**Theorem 1.4.94** (Uniqueness)

Let C be convex, f is proper and convex, $C \cap \text{dom}(f) \neq \emptyset$, then there is at most one minimizer of $\min_{x \in C} f(x)$ if either

- 1) f is strictly convex or
- 2) $C \cap \text{argmin } f = \emptyset$ and C is strictly convex.

Intuition. If the unconstrained minimizer is not in the constrained set, then C and not f dictates uniqueness.

Definition 1.4.95 (strictly convex set) — A set C is **strictly convex** if there are no line segments on its boundary. That is,

$$\frac{x+y}{2} \in \text{int}(C) \quad \forall x, y \in C, x \neq y.$$

Remark 1.4.96 In summary: let $f \in \Gamma_0(\mathbb{R}^n)$. If f is strictly convex \Rightarrow at most 1 minimizer. If f is coercive \Rightarrow at least 1 minimizer.

Corollary 1.4.97

If f is strongly convex, then there exists a unique minimizer.

Proof. Strongly convex \Rightarrow strictly convex. Also notice that if we choose $x = 0$, then

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle + \frac{\mu}{2} \|y - x\|^2$$

will be "coerced" to go to infinity as $\|y\| \rightarrow \infty$. \square

1.4.8 Proximity operator

Definition 1.4.98 (orthogonal projection) — The **orthogonal projection** of a point y onto a set C is

$$\text{Proj}_C(y) = \underset{x \in C}{\operatorname{argmin}} \|x - y\|_2 = \underset{x \in C}{\operatorname{argmin}} \frac{1}{2} \|x - y\|_2^2.$$

Note. Applying a monotone transformation to the objective doesn't change the location of minimizer, hence we can use norm squared.

Remark 1.4.99 The solution to the projection exists and is unique if C is a Chebyshev set *i.e.* closed and convex.

Definition 1.4.100 (proximity operator) — The **proximity operator** of a function $f \in \Gamma_0(\mathbb{R}^n)$ is

$$\text{prox}_f(y) = \underset{x}{\operatorname{argmin}} \left(\frac{1}{2} \|x - y\|_2^2 + f(x) \right).$$

Question 1.4.101. Is this minimizer unique? Does it even exist?

The solution is unique because f is convex and the norm squared is strongly convex in x , making the sum strongly convex which guarantees a unique minimizer.

Example 1.4.102

Let $f = I_C$ be an indicator function, then

$$\text{prox}_f(y) = \text{Proj}_C(y).$$

Remark 1.4.103 We often insert a scaling constant for convenience. This constant turns out to be the step size in an iterative algorithm:

$$\text{prox}_{tf}(y) = \underset{x}{\operatorname{argmin}} \frac{1}{2} \|x - y\|_2^2 + tf(x).$$

Example 1.4.104

$f(x) = \frac{1}{2} \|x\|_2^2$. Then

$$\text{prox}_f(y) = \underset{x}{\operatorname{argmin}} \frac{1}{2} \|x - y\|_2^2 + \frac{t}{2} \|x\|_2^2$$

Fermat and differentiable $\Rightarrow 0 = x^* - y + tx^*$

$$x^* = (1 + t)^{-1}y$$

Example 1.4.105 (taking gradient)

$f(x) = \frac{1}{2} \|Ax - b\|^2$, what is ∇f ?

Think of it as $f = g \circ h$, where $g(x) = \frac{1}{2} \|x\|^2$.

Example 1.4.106

$f(x) = \|x\|_1$. So

$$\begin{aligned} \text{prox}_{tf}(y) &= \underset{x}{\operatorname{argmin}} \frac{1}{2} \|x - y\|_2^2 + t \|x\|_1 \\ &= \underset{x}{\operatorname{argmin}} \frac{1}{2} \sum_{i=1}^n (x_i - y_i)^2 + t \sum_{i=1}^n |x_i| \end{aligned}$$

Notice each i is separable, so WLOG we just need to find the optimal x_i , dropping the i in notation:

$$\underset{x}{\operatorname{argmin}} \frac{1}{2} (x - y)^2 + t|x|$$

By Fermat's rule,

$$0 \in \partial \left(\frac{1}{2}(x - y)^2 + t|x| \right)$$

$$0 \in x - y + t\partial(|x|) \quad \text{via CQ such as full domain}$$

$$0 \in x - y + \begin{cases} t & x > 0 \\ [-t, t] & x = 0 \\ -t & x < 0 \end{cases}$$

Case (1). If $x > 0$, we have

$$0 = x - y + t \Rightarrow x = y - t$$

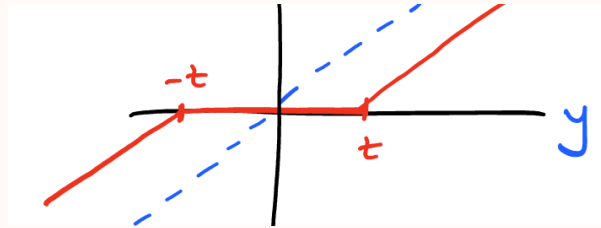
This is only valid if $y - t > 0 \Rightarrow y > t$.

Case (2). If $x < 0$, we have $0 = x - y - t \Rightarrow x = y + t \Rightarrow y < -t$.

Case (3). If $x = 0$, we have $0 \in -y + [-t, t] \Rightarrow y \in [-t, t]$.

This perfectly covers all cases.

$$\begin{aligned} \text{prox}_{t|x|}(y) &= \begin{cases} y - t & y > t \\ 0 & y \in [-t, t] \\ y + t & y < -t \end{cases} \\ &= \text{sgn}(y) \cdot [|y| - t]_+ \quad \text{where } [\alpha]_+ = \max(\alpha, 0) \end{aligned}$$



The last expression is called **soft thresholding** and is more computationally efficient. Therefore, $\text{prox}_{t\|\cdot\|_1}$ is just componentwise soft-thresholding.

Rules

If $\text{prox}_f, \text{prox}_g$ are known, there is not a general formula for prox_{f+g} . Even $\text{prox}_{f \circ L}$ isn't easy.

1.4.9 Moreau envelope

See supplemental lecture notes in "Proximity Operator".

1.5 Convex optimization problems

1.5.1 Tricks

The standard form of an optimization problem looks like:

$$\begin{aligned} & \min f_0(x) \\ & \text{subject to } f_i(x) \leq 0, i = 1, \dots, m \\ & \quad h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

1) Max to min:

$$\max_{x \in C} f(x) = - \min_{x \in C} -f(x).$$

2) Equivalent problems:

Example 1.5.1

$f(x) = \sqrt{|x|}$. Then $\operatorname{argmin} f(x) = \operatorname{argmin} f(x)^2$ and we turn it into a convex problem.

Example 1.5.2

$f(x) = \|Ax - b\| + \lambda\|x\|^2$. Equivalently, we can solve

$$\min \|Ax - b\|^2 + \sigma\|x\|^2.$$

So we need to adjust the constant (Lagrange multipliers).

3) Change of variables: This works especially well for affine transformation because it doesn't change convexity.

4) Eliminate equality constraints:

Example 1.5.3

$$\min_{Ax=b} f(x).$$

We can decompose $x = x_p + \ker A$ (particular solution + homogeneous solution). Let F be the basis of $\ker(A)$, then $x = x_p + Fz$. This way

we change the problem to

$$\min_z f(x_p + Fz).$$

Notice this eliminates the constraint by (affine) change of variable.

5) Slack variables:

$f_i(x) \leq 0$ iff there exists a $s_i \geq 0$ s.t. $f_i(x) + s_i = 0$. Then we turn $\min_x f_0(x)$, subject to $f_1(x) \leq 0$ into

$$\begin{aligned} \min_{x,s} f_0(x) \\ \text{subject to } f_1(x) + s &= 0 \\ s &\geq 0 \end{aligned}$$

This is less important nowadays since softwares are less constrained by the form we give them.

6) Epigraph:

$$\min_{x \in \mathbb{R}^n} f(x) \Leftrightarrow \min_{x \in \mathbb{R}^n, t \in \mathbb{R}} t, \quad f(x) \leq t.$$

Example 1.5.4

$$\begin{aligned} \min \|Ax - b\|_1 &= \min \sum_{i=1}^m |a_i^T x - b_i| \\ &= \min_{t \in \mathbb{R}^m, x \in \mathbb{R}^n} \mathbb{1}t \\ &\quad a_i^T x - b_i \leq t_i \\ &\quad a_i^T x - b_i \geq -t_i \end{aligned}$$

7) Solve coupled functions $\min f(x) + g(x)$. This is equivalent to

$$\min_{x,z} f(x) + g(z) \text{ subject to } x = z$$

This way we decouple the functions and make it easier to solve.

8) Marginalization:

$$\begin{aligned}\min_{x,y} f(x,y) &= \min_x \left(\min_y f(x,y) \right) \\ &= \min_x g(x)\end{aligned}$$

Note. We can always commute extremization of the same type.

1.5.2 Convex optimization problems [BV04 Ch.4.2]

We wish to make both the function and the constraint sets to be convex. A typical problem:

$$\begin{aligned}\min & f_0(x) \\ \text{subject to} & f_i(x) \leq 0, i = 1, \dots, m \\ & h_i(x) = 0, i = 1, \dots, p\end{aligned}$$

A convex problem would be

$$\begin{aligned}\min & f_0(x) \\ \text{subject to} & f_i(x) \leq 0 \\ & a_i^T x = b_i\end{aligned}$$

where f_0, \dots, f_m are convex functions and the equality constraints are affine.

Theorem 1.5.5

Consider the convex problem, $\min f(x), x \in C$. Assume $f \in \mathcal{C}^1$. Then x is optimal iff

- 1) $x \in C$
- 2) $\forall y \in C, \langle \nabla f(x), y - x \rangle \geq 0$ (Euler inequality).

Proof. (\Leftarrow):

$$f(y) \geq f(x) + \langle \nabla f(x), y - x \rangle \geq f(x)$$

(\Rightarrow): Suppose x is optimal but there exists a $y \in C$ s.t. $\langle \nabla f(x), y - x \rangle < 0$. Then for $t \in (0, 1]$, the 1D parameterization yields:

$$\begin{aligned}\phi(t) &= f(x + t(y - x)) \\ &= \phi(0) + \phi'(0)t + \frac{\phi''(\xi)}{2}t^2 && \text{Taylor} \\ &\leq f(x) + t\langle \nabla f(x), y - x \rangle && \phi \text{ convex by composition} \\ &< f(x)\end{aligned}$$

Clearly $\phi(t)$ is feasible and this contradicts that x is optimal. \square

Corollary 1.5.6

If $\langle d, z \rangle \geq 0 \forall z \Rightarrow d = 0$.

Proof. Take $z = -d$ and result follows from positive definiteness of norm. \square

Remark 1.5.7 Euler inequality is an example of a **variational inequality**, where we can replace the gradient with any operator.

unconstrained problems

$C = \mathbb{R}^n$, $\text{dom}(f) = \mathbb{R}^n$. Choose $y = x - \nabla f(x)$, we have

$$-\|\nabla f(x)\|^2 \geq 0 \Leftrightarrow \nabla f(x) = 0,$$

and we recover Fermat.

convex equality/affine constraints

$C = \{x : Ax = b\}$ where $A \in \mathbb{R}^m \times \mathbb{R}^n$. Then (2) becomes for all y s.t. $Ay = b$, $\langle \nabla f(x), y - x \rangle \geq 0$. We can write $y = y_p + v, v \in \ker(A)$. Since $Ax = b$, we can take $y_p = x$ so $y = x + v$. Then it becomes $\forall v \in \ker A, \langle \nabla f(x), v \rangle \geq 0$. Since $\ker A$ is closed under negation, we need $\langle \nabla f(x), -v \rangle \geq 0$. Thus we get the necessary and sufficient condition for x to be optimal:

$$\langle \nabla f(x), v \rangle = 0.$$

This means $\nabla f(x) \perp \ker A$. Recall that the coimage space is orthogonal to the kernel, $\nabla f(x) \in \text{coim } A = \text{im } A^T$. Thus the condition is equivalent to there exists a $\nu \in \mathbb{R}^m$ s.t. $\nabla f(x) = A^T(-\nu)$. Rearranging yields:

$$\nabla f(x) + A^T \nu = 0.$$

The ν here is the **Lagrange multipliers**.

1.5.3 Conic optimization problems [BV04 Ch.4.3, 4.4, 4.6]

Linear programs

Linear objective with polyhedra constraints.

Standard forms:

$$\begin{array}{lll} \min \langle c, x \rangle & \min \langle c, x \rangle & \min \langle c, x \rangle \\ \text{subject to } Gx \leq h & x \geq 0 & Ax \leq b \\ & Ax = b & \end{array}$$

Example 1.5.8 (convert first to second form)

Use slack variable $s \geq 0$ s.t. $Gx + s = b$, $x = x^+ - x^-$, $x^+, x^- \geq 0$.

Let $\tilde{x} = \begin{pmatrix} x^+ \\ x^- \\ s \end{pmatrix} \geq 0$. Then we can combine the equality constraints into $\tilde{A}\tilde{x} = \tilde{b}$.

Example 1.5.9

$\min_x \|x\|_1$ s.t. $Ax = b$. Since this is separable/piecewise-linear, we use slack variables to rewrite $x_i = x_i^+ - x_i^-$, $x_i^+, x_i^- \geq 0$. Thus, $|x_i| \leq x_i^+ + x_i^-$ which will be equality when we minimize. Now we have an equivalent problem

$$\begin{array}{ll} \min_{x^+, x^-} & \mathbb{1}^T(x^+ + x^-) \\ \text{subject to} & Ax^+ - Ax^- = b \end{array}$$

Example 1.5.10 (epigraph trick)

$$\begin{aligned} \min_x & \|x\|_\infty \\ \text{subject to } & Ax = b \end{aligned}$$

is equivalent to

$$\begin{aligned} \min_{x,t} & t \\ \text{subject to } & Ax = b \\ & \|x\|_\infty \leq t \end{aligned}$$

Remark 1.5.11 LP is the most well-studied kind of convex optimization problem. It has wide applications in industry and can handle very large dimensions. However, integer LP is not convex and is NP-hard. But it is not impossible. In fact, it just takes a little longer to run and we can still prove that we have the best solution when we find one. CPLEX, MOSEK, Gurobi...

Theorem 1.5.12

LP solution set always includes a vertex of the polyhedron.

Quadratic programs

$$\begin{aligned} \min_x & \frac{1}{2} \langle x, Px \rangle + \langle q, x \rangle + r \\ \text{subject to } & Gx \leq h \\ & Ax = b \end{aligned}$$

Note. Quadratic programs are not always convex. It must have $P \succeq 0$ to be convex.

Example 1.5.13 (Regression)

$$\min_x \frac{1}{2} \|Ax - b\|^2 = \frac{1}{2} \langle x, \underbrace{A^T A}_{\succeq 0} x \rangle - \langle x, A^T b \rangle + \|b\|^2$$

Quadratically constrained quadratic program (QCQP)

$$\begin{aligned} \min_x \quad & \frac{1}{2} \langle x, Px \rangle + \langle q, x \rangle + r \\ \text{subject to} \quad & \frac{1}{2} \langle x, P_i x \rangle + \langle q_i, x \rangle + r_i \leq 0 \quad \forall i = 1, \dots, m \\ & Ax = b \end{aligned}$$

Second-order cone program (SOCP)

It generalizes convex QCQP:

$$\begin{aligned} \min_x \quad & \langle c_0, x \rangle \\ \text{subject to} \quad & \|A_i x + b_i\|_2 \leq \langle c_i, x \rangle + d_i \\ & Fx = g \end{aligned}$$

The inequality constraint is an affine composition with 2nd order cone (recall $(y, t) \in \mathcal{K} \Leftrightarrow \|y\|_2 \leq t$). This can be solved efficiently via cvxpy.

Conic programming

The standard form (2nd form) of LP is an example of conic program where the proper cone $\mathcal{K} = \mathbb{R}_+^n$. More generally, a **conic program** is

$$\begin{array}{ll} \min \langle c, x \rangle & \min \langle c, x \rangle \\ \text{subject to } Fx + g \preceq_{\mathcal{K}} 0 & x \succeq_{\mathcal{K}} 0 \\ Ax = b & Ax = b \end{array}$$

where \mathcal{K} is a proper cone (closed, convex, solid, pointed) and $y \succeq_{\mathcal{K}} 0$ means $y \in \mathcal{K}$.

Recall that if $\mathcal{K}_1, \mathcal{K}_2$ are proper cones, so is $\mathcal{K} := \mathcal{K}_1 \times \mathcal{K}_2$. Thus for two different

inequality constraints in the first form, we can instead write

$$\begin{pmatrix} F_1 \\ F_2 \end{pmatrix} x + \begin{pmatrix} g_1 \\ g_2 \end{pmatrix} \preceq_{\mathcal{K}} 0.$$

This is how SOCPs is a conic program.

Semi-definite programs

This is a powerful tool that revolutionized engineering. Here the proper cone $\mathcal{K} = \mathbb{R}_+^n$ (and direct products of these).

Notation. \mathcal{K} is omitted in \succeq here because SDP is so common.

$$\begin{aligned} & \min \langle C, X \rangle \\ & \text{subject to } \langle A_i, X \rangle = b_i, i = 1, \dots, m \Leftrightarrow \mathcal{A}(X) = b \\ & X \succeq 0 \end{aligned}$$

Note. Here \mathcal{A} is some abstract linear operator that can be flatten to a $m \times n^2$ matrix if we vectorize X .

Remark 1.5.14 $\langle C, X \rangle = \text{tr}(C^T X) = \langle \text{vec } C, \text{vec } X \rangle$. To implement this, we do not want to multiple the matrices. Since $\mathbb{R}^{n \times n} \cong \mathbb{R}^{n^2}$, we can vectorize it and achieves $\mathcal{O}(n^2)$.

Remark 1.5.15 $\text{vec} : \mathbb{R}^{n \times n} \rightarrow \mathbb{R}^{n^2}$ and $\text{mat} : \mathbb{R}^{n^2} \rightarrow \mathbb{R}^{n \times n}$ are inverses.

$S^n \cong \mathbb{R}^{n(n+1)/2}$ because we can remove redundancy from symmetry. Although if memory is not an issue, it is usually easier to just use the whole thing since we have to reweight off-diagonal terms.

Remark 1.5.16 We can treat a matrix as a vector using trace for inner product and Frobenius norm or as a transformation using spectral norm (doesn't have inner product because it's in Banach space not in Hilbert space).

Remark 1.5.17 We have the Kronecker product trick

$$\text{vec}(AXB) = (B^T \otimes A) \text{vec}(X).$$

Not all linear operator on matrices $\mathcal{A}(X)$ are in this form because we need to factorize the operator. Moreover, LHS is computationally cheaper than RHS.

Linear matrix inequalities (LMI): dual problem of SDPs

$$\begin{aligned} & \min_{x \in \mathbb{R}^n} \langle c, x \rangle \\ & \text{subject to } \sum_{i=1}^m x_i F_i + G \preceq 0, i = 1, \dots, m, F_i, G \in S^k \\ & Ax = b \end{aligned}$$

Note. $\sum_{i=1}^m x_i F_i$ is like $A^*y = \sum_{i=1}^m y_i \mathbf{a}_i$ in the case when $Ax = b \Rightarrow \mathbf{a}_i^T x = b$.

Remark 1.5.18 We can recover linear programs by letting F_i, G be diagonal matrices.

Remark 1.5.19 We can also recover SOCP's. TODO HOW??? Let $A \in S_{++}^r, C \in S^s, B \in \mathbb{R}^{r \times s}$. Then

$$\begin{pmatrix} A & B \\ B^T & C \end{pmatrix} \succeq 0 \Leftrightarrow \underbrace{C - B^T A^{-1} B}_{\text{Schur complement}} \succeq 0.$$

Schur complement might be computationally cheaper especially for example when $C = 0$.

Let K_1, K_2 be proper cones, then $K_1 \times K_2$ is also a proper cone.

Example 1.5.20

$X \succeq 0, Y \succeq 0$, we can write

$$\begin{pmatrix} X & Z \\ Z^T & Y \end{pmatrix} \succeq 0, Z = 0 \text{ (linear constraint)}.$$

However, this is horrible for computation. For example, in the case of negative log barrier, we can separate each constraint and projecting to \mathbb{R}_+^n is easy. We can also project to S_+^n by making the eigenvalues to nonnegative. But doing this on a bigger matrix is expensive since finding eigenvalues is super-linear.

1.6 Duality [BV04 Ch.5]

1.6.1 Lagrange dual function/problem

Consider $p^* = \min_{x \in C} f(x)$. Here $x \in C$ is **primal feasible** and we can find it by finding the smallest upper bound. We wish to find a dual feasible point s.t. it is the largest lower bound on p^* .

First start with the primal problem without assuming convexity:

$$\begin{aligned} \min f_0(x) \\ \text{subject to } f_i(x) \leq 0, i = 1, \dots, m \\ h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

Given an non-empty domain $D = \bigcap_{i=1}^m \text{dom}(f_i) \cap \bigcap_{i=1}^p \text{dom}(h_i)$. Note that $\text{dom}(f_0)$ is implicit constraint, *i.e.* $f_0(x) = x^2 + I_{\{x \geq 0\}}$.

Definition 1.6.1 (lagrangian) —

$$\mathcal{L}(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x),$$

where λ, ν are **dual variables**, λ is associated with inequality constraints whereas ν is associated with equality constraints.

Remark 1.6.2 We can be creative about whether we put the constraints explicitly in f_i, h_i or implicitly in f_0 .

Definition 1.6.3 (dual function) —

$$g(\lambda, \nu) = \inf_{x \in D} \mathcal{L}(x, \lambda, \nu).$$

Note. This problem is usually easier than the primal problem because it doesn't have constraints.

Question 1.6.4. Is g convex?

Recall the infimum preserves convexity iff C is convex and $f(x, y)$ is jointly convex, whereas the supremum preserves convexity iff $f(x, y)$ is convex for all

x . Since the Lagrangian is most likely not jointly convex, but it is in fact affine in terms of the dual variables. Therefore,

$$g(\lambda, \nu) = - \sup_{x \in D} \mathcal{L}(x, \lambda, \nu)$$

is concave.

Dual problem (D):

$$d^* = \max_{\lambda \geq 0} g(\lambda, \nu).$$

Proposition 1.6.5 (weak duality)

Define p^* to be the optimal value for the primal problem and d^* be the maximum value for the dual problem. Then if $\lambda \geq 0$ and ν is anything, then

$$g(\lambda, \nu) \leq p^*.$$

Hence $d^* \leq p^*$.

Proof.

$$\begin{aligned} g(\lambda, \nu) &= \inf_x \mathcal{L}(x, \lambda, \nu) \\ &\leq \mathcal{L}(x, \lambda, \nu) \quad \forall x \in D \text{ and feasible} \\ &= f_0(x) + \sum_{i=1}^m \underbrace{\lambda_i}_{\geq 0} \underbrace{f_i(x)}_{\leq 0} + \sum_{i=1}^p \nu_i \underbrace{h_i(x)}_{=0} \quad x \text{ is feasible} \\ &\leq f_0(x) \end{aligned}$$

□

Remark 1.6.6 "Strong duality" $d^* = p^*$ tend to happen if (P) is convex.

Example 1.6.7 (dual of a LP)

$$\begin{aligned}
& \min \langle c, x \rangle \\
& \text{subject to } x \geq 0 \Leftrightarrow -x_i \leq 0 \\
& Ax = b
\end{aligned}$$

Then

$$\begin{aligned}
\mathcal{L}(x, \lambda, \nu) &= \langle c, x \rangle - \langle \lambda, x \rangle + \nu^T (Ax - b) \\
&= \langle c, x \rangle - \langle \lambda, x \rangle + \langle A^T \nu, x \rangle - \langle \nu, b \rangle \\
g(\lambda, \nu) &= \inf_x \mathcal{L}(x, \lambda, \nu) \\
&= -\langle \nu, b \rangle + \inf_{\lambda} \langle c - \lambda + A^T \nu, x \rangle = \begin{cases} -\langle \nu, b \rangle & c - \lambda + A^T \nu = 0 \\ -\infty & \text{else} \end{cases}
\end{aligned}$$

Thus the dual problem $\max_{\lambda \geq 0} g(\lambda, \nu)$ becomes

$$\max_{\lambda \geq 0} -\langle \nu, b \rangle, \lambda = c + A^T \nu$$

or

$$-\min \langle \nu, b \rangle, c + A^T \nu \geq 0.$$

This is a LP!

Dual of linear programs

The primal (P) is:

$$\begin{aligned}
& \max_{x \in \mathbb{R}^n} \langle c, x \rangle \\
& \text{subject to } x \geq 0 \\
& y : Ax \leq b
\end{aligned}$$

The dual (D) is:

$$\begin{aligned}
& \min \langle b, y \rangle \\
& \text{subject to } y \geq 0 \\
& x : A^T y \geq c
\end{aligned}$$

Rules to transform (P) to (D):

- "SOB" mnemonic: sensible, odd, and bizarre from a business perspective

no constraint: odd

constraints in primal when maximizing $a_i^T x \leq b_i$: sensible (think budget)

$$a_i^T x = b_i: \text{ odd}$$
$$a_i^T x \geq b_i: \text{bizarre}$$

Example 1.6.8

$$\begin{array}{ll} (P) & \min_{\substack{x \geq 0 \\ x \in \mathbb{R}^2}} \quad 3x_1 + 2x_2 \\ & y_1 : \quad x_1 + 2x_2 \geq 5 \quad S \\ & y_2 : \quad \underbrace{x_2 \leq 2}_{\begin{pmatrix} 1 & 2 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}} \quad B \end{array}$$

$$(D) \quad \begin{array}{ll} \min_{\substack{y \in \mathbb{R}^2 \\ y_1 \geq 0 \quad S \\ y_2 \leq 0 \quad B}} & 5y_1 + 2y_2 \\ x_1 : & y_1 + 0y_2 \leq 3 \quad S \\ x_2 : & 2y_1 + y_2 \leq 2 \quad S \end{array}$$

Observe: $f(x) = 3x_1 + 2x_2 = \underbrace{2x_1}_{\geq 0} + \underbrace{(x_1 + 2x_2)}_{\geq 5} \geq 5$. We proved $5 \leq p^*$.
 However, this is not the tightest bound. The dual variables give us the tightest: 3 times the first constraint and -4 times the second constraint yields $7 \leq p^*$.

Duality gap: x, λ feasible, $f_0(x) - g(\lambda, \nu)$.

Strong duality results

- If (P) isn't convex, strong duality is unlikely except certain nonconvex QP: s-lemma/s-procedure (see Appendix of BV).
- If (P) is convex, strong duality holds under certain constraint qualifications (CQ) such as Slater's condition.

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, i = 1, \dots, m \\ & Ax = b \end{aligned}$$

Definition 1.6.9 (Slater's conditions) — They hold if there exists a strictly feasible point, $x \in \text{ri}(\text{dom}(f_0))$ and

if f_i is affine, $f_i(x) \leq 0$ (feasible)

if f_i isn't affine, $f_i(x) < 0$ (strictly feasible)

and $Ax = b$.

Theorem 1.6.10

If (P) is convex and Slater's conditions hold, then

- (i) we have strong duality, $d^* = p^* < \infty$
- (ii) there exists an optimal solution to the dual problem.

Note. Slater's does NOT imply there exists an optimal *primal* solution.

Example 1.6.11

$\inf_{x \in \mathbb{R}} e^x$. It is convex, lsc, proper. But it is not coercive so it doesn't have an optimal primal solution.

Remark 1.6.12 Often we want Slater's condition on the dual. Since the dual of the dual is the primal, then we guarantee an optimal solution.

Corollary 1.6.13 (Slater for LP)

Slater's conditions hold iff the LP is feasible *i.e.* $p^* < \infty$.

$p^* < \infty \Rightarrow d^* = p^*$ and dual optimal solution exists.

$d^* > -\infty \Rightarrow d^* = p^*$ and primal optimal solution exists.

Hence if either p^* or $d^* \in \mathbb{R}$ (not $\pm\infty$), then optimal primal and dual solutions exist.

Note. $d^* = -\infty, p^* = -\infty$ is possible but rare. This is not strong duality.

1.6.2 Saddle point interpretation [BV 4.2]

Here we want to find the saddle points as we want to minimize the primal but maximize the dual.

$$\begin{aligned} p^* &= \min_x f_0(x) \\ \text{subject to } f_i(x) &\leq 0, i = 1, \dots, m \\ Ax &= b \end{aligned}$$

This is equivalent to

$$\begin{aligned} &\min_x f_0(x) + \sup_{\lambda \geq 0, \nu} \left\{ \sum \lambda_i f_i(x) + \nu^T (Ax - b) \right\} \\ &= \min_{x \in D} \sup_{\lambda \geq 0, \nu} \mathcal{L}(x, \lambda, \nu) \end{aligned}$$

This is because if $f_i(x) > 0$ or $a_i^T x - b_i \neq 0$ for some i , then we get ∞ in the supremum, encoding it as infeasible.

Then the dual is

$$d^* = \max_{\lambda \geq 0, \nu} g(\lambda, \nu) = \max_{\lambda \geq 0, \nu} \min_{x \in D} \mathcal{L}(x, \lambda, \nu).$$

The weak-duality is equivalent to "min-max" inequality:

$$d^* = \max_{\lambda \geq 0, \nu} \min_{x \in D} \mathcal{L}(x, \lambda, \nu) \leq \min_{x \in D} \max_{\lambda \geq 0, \nu} \mathcal{L}(x, \lambda, \nu) = p^*.$$

And equality is achieved if strong-duality holds.

Note. All "min/max" should be "inf/sup" until proven.

Theorem 1.6.14

Saddle point occurs when

- (1) strong-duality/strong max/min
- (2) inf/sup are achieved.

That is, $(x^*, (\lambda^*, \nu^*))$ is a saddle point of $\mathcal{L}(x, (\lambda, \nu))$ if

$$\begin{aligned} \mathcal{L}(x^*, (\lambda^*, \nu^*)) &= \inf_x \mathcal{L}(x, (\lambda^*, \nu^*)) \\ \mathcal{L}(x^*, (\lambda^*, \nu^*)) &= \sup_{\lambda, \nu} \mathcal{L}(x^*, (\lambda, \nu)) \end{aligned}$$

Corollary 1.6.15

If we know λ^*, ν^* , then we can find x^* by solving the unconstrained problem

$$\min_x \mathcal{L}(x, (\lambda^*, \nu^*)).$$

This allows us to solve problems with shared Lagrangians.

Shared Lagrangian

Example 1.6.16

$$\begin{aligned} \min \quad & \|x\|_1 \\ \text{subject to} \quad & \|Ax - b\|_2 \leq \varepsilon \quad \Leftrightarrow \quad \|Ax - b\|_2^2 - \varepsilon^2 \leq 0 \end{aligned}$$

Let

$$\mathcal{L}(x, \lambda) = \|x\|_1 + \lambda \left(\|Ax - b\|_2^2 - \varepsilon^2 \right).$$

With the correct λ^* , this is equivalent to

$$\min_x \|x\|_1 + \lambda^* \|Ax - b\|_2^2,$$

because dropping the constant doesn't affect minimizer. This unconstrained problem is much nicer because the least squares is differentiable, whereas the original constraint is hard to project.

Even if we don't know λ^* ,

- (1) guess λ , solve $x = x(\lambda)$, check if the constraint is active, update λ (solve the dual problem).
- (2) often ε is not known (hyper-parameter) and set via cross-validation so we can do cross-validation on λ directly (evaluate trade-off in modeling).

We assume existence of saddle points here, which is given by the following:

Proposition 1.6.17

Slater's on both primal and dual \Rightarrow existence of saddle points.

1.6.3 Game Theory connection

Consider a finite, 2-person, 0-sum game: "matrix game" (not Prisoner's dilemma).

This involves the Minimax Theorem of Von Neumann.

Example 1.6.18 (rock-paper-scissors)

Player 1 wants to minimize and Player 2 wants to maximize utility. The payoff matrix looks like

	P	S	R
P	0	1	-1
S	-1	0	1
R	1	-1	0

Row: Player 1; Column: Player 2

$u^T P v$ is the payoff, intuitively it means player 1 chooses a row and player 2 chooses a column. For a fair game, the payoff value is 0. Since $A = -A^T$ is antisymmetrical, it's fair. But in reality, u and v actually encode the probability of choose each row/column, which sums up to 1.

Define probability simplex $\Delta = \{u : u \geq 0, \sum u_i = 1\}$.

Case (Player 2 knows player 1's strategy). If u is known, Then the decision is easy: choose $v \in \arg\max_{v \in \Delta} u^T P v$.

If Player 1 knows Player 2 knows Player 1's strategy, then Player 1 should select u to minimize Player 2's payoff:

$$p_1^* = \min_{u \in \Delta} \max_{v \in \Delta} u^T P v.$$

This is in fact a LP.

Case (Player 1 knows Player 2's strategy).

$$p_2^* = \max_{v \in \Delta} \min_{u \in \Delta} u^T P v.$$

Intuitively, whoever has knowledge of opponent's move gets an edge, so the payoff when Player 2 has an edge in maximizing will be at least the payoff when Player 1 has an edge in minimizing. That is, $p_1^* \geq p_2^*$. This is weak duality. Slater's condition for LP requires only a feasible point. Since Δ is nonempty, we have strong duality $p_1^* = p_2^*$.

1.6.4 Fenchel-Rockafellar Duality [BC17]

$$(P) \quad \min_x f(x) + g(Ax),$$

where $f, g \in \Gamma_0$ and allows $+\infty$ values, and A is a $m \times n$ matrix.

$$(D) \quad \min_v f^*(A^* v) + g^*(-v).$$

Connections to Lagrangian duality

Recall

$$f^*(y) = \sup_x \langle x, y \rangle - f(x).$$

Take the same (P) , recast as

$$\min_{x,z} f(x) + g(z) \text{ s.t. } z = Ax \quad \Rightarrow \quad \mathcal{L}(x, z, v) = f(x) + g(z) + \langle z - Ax, v \rangle.$$

Then the Lagrangian dual function $h(v)$ is

$$\begin{aligned} h(v) &= \inf_{x,z} \mathcal{L}(x, z, v) = \inf_x (f(x) - \langle Ax, v \rangle) + \inf_z (g(z) + \langle z, v \rangle) \\ &= -\sup_x (\langle x, A^*v \rangle - f(x)) - \sup_z (-\langle z, v \rangle - g(z)) \\ &= -(f^*(A^*v) + g^*(-v)) \end{aligned}$$

Thus, the Lagrangian and F-R dual problems only differ by a minus sign:

$$\max_v h(v) = -\min_v -h(v) = -\min_v f^*(A^*v) + g^*(-v).$$

Saddle-point interpretation

If $g \in \Gamma_0$, then $g = g^{**}$, so $g(x) = \sup_y \langle x, y \rangle - g^*(y)$. Using this fact we can rewrite the primal problem as

$$\begin{aligned} (P) \quad \min_x f(x) + g(Ax) &= \min_x \sup_v f(x) + \langle Ax, v \rangle - g^*(v) \\ &= \sup_v \min_x f(x) + \langle x, A^*v \rangle - g^*(v) \quad \text{if saddle point exists} \\ &= \sup_v -f^*(-A^*v) - g^*(v) \\ &= \sup_v -f^*(A^*v) - g^*(-v) \quad (D) \end{aligned}$$

Proposition 1.6.19 (18.9)

Let $f \in \Gamma_0(\mathcal{H})$, if f^* is strictly convex, then f is (Gateaux) differentiable on $\text{int dom}(f)$.

Proposition 1.6.20 (18.15)

If f is continuous and convex, then

f is (Frechet) differentiable and ∇f is L -Lipschitz continuous if and only if f^* is L^{-1} strongly convex,
and $f \in \Gamma_0, f = f^*$.

1.6.5 Algorithms

- (1) gradients: If I know ∇g , can I find $\nabla(g \circ A)$? Yes. It's $A^*(\nabla g \circ A)$.
- (2) projections/proximity operators: Let $C = \{x : \|x\|_2 \leq 1\}$ and $C \circ A = \{x : \|Ax\|_2 \leq 1\}$. In general, if I know prox_g , I don't know $\text{prox}_{g \circ A}$. Here there is no chain rule nor linearity.

Remark 1.6.21 We can use the dual to shift the linear operator from the proximity term to the differentiable term.

Theorem 1.6.22 (15.23 generalized Slater)

If $0 \in \text{ri}(\text{dom } g - A(\text{dom } f))$ (CQ), then strong duality holds. That is,

$$\inf_x f(x) + g(Ax) = -\min_v f^*(A^*v) + g^*(-v),$$

and the dual solution is obtained.

Note. In finite dimensions, for CQ we just need to show $\text{ri}(\text{dom } g) \cap A(\text{ri}(\text{dom } f)) \neq \emptyset$. Or, if f, g are polyhedral, $\text{dom } g \cap A(\text{dom } f) \neq \emptyset$. This is essentially saying we want a strictly feasible point.

Example 1.6.23

$$\begin{aligned} (P) \quad & \min \quad \frac{1}{2} \|x - x_0\|^2 \\ & \text{subject to} \quad \|Ax - b\| \leq \varepsilon \end{aligned}$$

This is hard to solve directly using gradient descent, as we would need to project which requires finding the SVD of A . Let $f(x) = \frac{1}{2}\|x - x_0\|^2$ and $g(y) = I_{\|y-b\|\leq\varepsilon}$. Then

$$\begin{aligned}
f^*(v) &= \sup_x \langle v, x \rangle - \frac{1}{2}\|x - x_0\|^2 \\
&= \langle v, x_0 \rangle + \sup_{\tilde{x}} \langle v, \tilde{x} \rangle - \frac{1}{2}\|\tilde{x}\|^2 \\
&= \langle v, x_0 \rangle + \frac{1}{2}\|v\|^2 \\
g^*(v) &= \sup_{\|y-b\|\leq\varepsilon} \langle v, y \rangle \\
&= \langle v, b \rangle + \sup_{\|\tilde{y}\|\leq\varepsilon} \langle v, \tilde{y} \rangle \\
&= \langle v, b \rangle + \varepsilon\|v\|_2
\end{aligned}$$

Hence we obtain the dual:

$$(D) \quad \min_v \left(\langle A^*v, x_0 \rangle + \frac{1}{2} \right).$$

1.6.6 Optimality conditions

Complementary slackness

Suppose we have primal and dual optimal solutions, x^*, λ^*, ν^* , and have strong duality (no need for convexity). That is, we assume saddle points exist. Observe

$$\begin{aligned}
p^* &= \min_x f_0(x) = f_0(x^*) = g(\lambda^*, \nu^*) = \inf_x \mathcal{L}(x, \lambda^*, \nu^*) \\
\text{subject to } & f_i(x) \leq 0, i = 1, \dots, m \\
& h_i(x) = 0, i = 1, \dots, p
\end{aligned}$$

Thus,

$$\begin{aligned}
p^* &\leq \mathcal{L}(x^*, \lambda^*, \nu^*) \\
&= f_0(x^*) + \sum \underbrace{\lambda_i^*}_{\geq 0} \underbrace{f_i(x^*)}_{\leq 0} + \sum \underbrace{\nu_i^* h_i(x^*)}_{=0} \\
&\leq f_0(x^*) = p^*
\end{aligned}$$

Hence, for all i , $\lambda_i^* f_i(x^*) = 0$. This is called **complementary slackness**. That is, either $\lambda_i^* = 0$ or $f_i(x^*) = 0$.

If $f_i(x^*) < 0$, then λ_i^* must be zero, meaning that the constraints have no effect on the solution (not tight).

KKT conditions

Theorem 1.6.24 (KKT 1 "necessary")

Suppose f_i, h_i are differentiable. If x^* is primal optimal, and λ^*, ν^* dual optimal, and no duality gap (strong duality), then necessarily x^*, λ^*, ν^* satisfy

(1) stationarity:

$$0 = \nabla f_0(x^*) + \sum \lambda_i^* \nabla f_i(x^*) + \sum \nu_i^* \nabla h_i(x^*) = \nabla_x \mathcal{L}(x^*, \lambda^*, \nu^*).$$

(2) primal feasibility: $f_i(x^*) \leq 0, h_i(x^*) = 0$.

(3) dual feasibility: $\lambda^* \geq 0$.

(4) complementary slackness: $\lambda_i^* f_i(x^*) = 0$.

Note. There is no need for convexity here.

Remark 1.6.25

(a) These are only necessary for saddle points and may not be enough without saddle points or strong duality.

Example 1.6.26

$\min_x e^{-x}$ satisfies the conditions but has no saddle points. It in fact doesn't have a minimizer.

(b) only needed if differentiable.

(c) If functions are nonconvex, these may not be sufficient. That is, there may be non-optimal solutions.

Remark 1.6.27 If the primal problem is convex, then the existence of saddle points guarantees strong duality.

We can generalize stationarity:

$$(1) \ x^* \in \operatorname{argmin}_x \mathcal{L}(x, \lambda^*, \nu^*).$$

By convexity, we have

$$0 \in \partial \mathcal{L}(x^*, \lambda^*, \nu^*).$$

If we have CQ,

$$0 \in \partial f_0(x^*) + \sum \lambda_i^* \partial f_i(x^*) + \sum \partial h_i(x^*).$$

Theorem 1.6.28 (KKT 2 sufficient)

Suppose (P) is convex (f_i are convex, h_i are affine). If (x^*, λ^*, ν^*) solve the KKT conditions, then they are the primal and dual optimal and there is no duality gap.

Proof. Assume (x^*, λ^*, ν^*) satisfies the KKT conditions.

$$\begin{aligned} p^* &\leq f_0(x^*) && x^* \text{ is feasible} \\ &= \mathcal{L}(x^*, \lambda^*, \nu^*) && \text{since feasible and comp. slack.} \\ &\inf_x \mathcal{L}(x, \lambda^*, \nu^*) && \text{stationarity} \\ &= g(\lambda^*, \nu^*) && \text{dual feasibility} \\ &\leq d^* \end{aligned}$$

Thus we prove strong duality, and $p^* = f_0(x^*) \Rightarrow x^*$ is optimal. Same for duals. □

Remark 1.6.29 Complementary slackness means if $f_j(x^*) < 0$ is an inactive constraint, then $\lambda_j^* = 0$, so

$$\mathcal{L}(x, \lambda^*, \nu^*) = f_0(x) + \sum_{i \neq j} \lambda_i^* f_i(x) + \sum \nu_i^* h_i(x)$$

$$\begin{aligned} (P') \quad & \min f_0(x) \\ & \text{subject to } f_i(x) \leq 0, \forall i \neq j \\ & h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

This is only true if we have KKT conditions.

Example 1.6.30 (1)

$$\begin{aligned} & \min_{x \in \mathbb{R}} x \\ & \text{subject to } x \geq 0 \\ & x \leq 1 \text{ this is not tight/active constraint} \end{aligned}$$

We can just remove the inactive constraint and still get the same solution.

Example 1.6.31 (2)

$$\begin{aligned} & \min_{x \in \mathbb{R}} x \\ & \text{subject to } x \geq 0 \quad \text{inactive} \\ & x^2 \geq 1 \end{aligned}$$

But if we remove the inactive constraint here, we would get $-\infty$ as the solution instead, so we can't just drop inactive constraint for non-convex problems.

1.6.7 Meta-rules

Suppose $C \subseteq \mathbb{R}^n$, possibly nonconvex.

- (1) switch between min and max with double minus signs or between argmin and argmax with single minus sign (since we don't care about function value).

- (2) If ϕ is monotone on $\text{im}(f)$, then

$$\operatorname{argmin}_{x \in C} \phi(f(x)) = \operatorname{argmin}_{x \in C} f(x).$$

Example 1.6.32

$$\frac{1}{2} \|Ax - b\|^2 \text{ and } \|Ax - b\|.$$

- (3) If we have all mins or all maxs, we can swap order

$$\min_x \min_y f(x, y) = \min_y \min_x f(x, y) = \min_{x, y} f(x, y).$$

- (4) If $D \subseteq C$ where C can be seen as a relaxation, then

$$\min_{x \in C} f(x) \leq \min_{x \in D} f(x).$$

And we can obtain a lower bound this way.

- (5) "superadditivity":

$$\min_{x \in C} f(x) + g(x) \geq \min_{x \in C} f(x) + \min_{x \in C} g(x).$$

Example 1.6.33 (solving convex problems using KKT)

Recall that the solution to the least squares problem $\min_x \frac{1}{2} \|Ax - b\|^2$ when A has more rows than columns ($m \geq n$) is

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

In the case when $m < n$, the system is underdetermined so $Ax = b$ has many solutions, so we instead want to find the solution with the least Euclidean norm (since we can add any vector from $\ker A$ to arbitrarily

inflate the norm of the solution):

$$\begin{aligned} \min \quad & \frac{1}{2} \|x\|^2 \\ \text{subject to} \quad & Ax = b \end{aligned}$$

And the solution is

$$x^* = A^T (AA^T)^{-1} b.$$

To see this, consider the more general quadratic problem

$$\begin{aligned} \min \quad & \frac{1}{2} \langle x, Px \rangle + \langle q, x \rangle + r, P \succeq 0 \\ & Ax = b \end{aligned}$$

Note that we recover the problem when $P = I, q = r = 0$. The Lagrangian is

$$\mathcal{L}(x, \nu) = \frac{1}{2} \langle x, Px \rangle + \langle q, x \rangle + r + \nu^T (Ax - b).$$

The KKT conditions are the following:

(1) stationarity:

$$\nabla_x \mathcal{L}(x, \nu) = Px + q + A^T \nu = 0.$$

(2) primal feasibility:

$$Ax = b.$$

(3) dual feasibility: N/A.

(4) Complementary slackness: N/A.

Since the conditions are linear equations, we can combine them into a larger system of equations:

$$\begin{pmatrix} P & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ \nu \end{pmatrix} = \begin{pmatrix} -q \\ b \end{pmatrix}.$$

So when $P = I, q = r = 0$, we have

$$\begin{aligned} \begin{pmatrix} A & AA^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ \nu \end{pmatrix} &= \begin{pmatrix} 0 \\ b \end{pmatrix} \\ \begin{pmatrix} x \\ \nu \end{pmatrix} &= \begin{pmatrix} A^T (AA^T)^{-1} b \\ - (AA^T)^{-1} b \end{pmatrix} \end{aligned}$$

Example 1.6.34

Consider the problem

$$\begin{aligned} \min \quad & \frac{1}{2} \|Ax - b\|^2 \\ \text{subject to} \quad & \mathbb{1}^T x \leq \tau \end{aligned}$$

The Lagrangian is

$$\mathcal{L}(x, \lambda) = \frac{1}{2} \|Ax - b\|^2 + \lambda(\mathbb{1}^T x - \tau).$$

The KKT conditions are

(1)

$$\nabla_x \mathcal{L}(x, \lambda) = A^T(Ax - b) + \lambda \mathbb{1} = 0.$$

(2) $\mathbb{1}^T x - \tau \leq 0$.

(3) $\lambda \geq 0$.

(4) $\lambda = 0$ or $\mathbb{1}^T x - \tau = 0$.

In the case when $\lambda = 0$, the problem reduces to least squares and we've already solved it. When $\lambda \neq 0$ and $\mathbb{1}^T x = r$ instead, we can solve it the following way:

$$\begin{aligned} x &= (A^T A)^{-1} (A^T b - \lambda \mathbb{1}) \\ \tau &= \mathbb{1}^T x = \mathbb{1}^T (A^T A)^{-1} (A^T b - \lambda \mathbb{1}) \\ \lambda &= \frac{\mathbb{1}^T (A^T A)^{-1} A^T b - \tau}{\mathbb{1}^T (A^T A)^{-1} \mathbb{1}} \end{aligned}$$

Note that λ is just a scalar.

1.6.8 Perturbation and Sensitivity Analysis [BV 5.6]

Given the primal problem

$$\begin{aligned} (P) \quad & p^* = \min_x f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, i = 1, \dots, m \\ & h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

and let's perturb this a little:

$$\begin{aligned}
(P_{u,v}) \quad p(u,v) = \min \quad & f_0(x) \\
\text{subject to} \quad & f_i(x) \leq u_i, i = 1, \dots, m \\
& h_i(x) = v_i, i = 1, \dots, p
\end{aligned}$$

So $p^* = p(0, 0)$, what is $p(u, v)$?

We can do a local analysis and find out the derivatives or a global analysis using a bound.

global analysis

We can use Slater's condition (strict feasibility) to check strong duality and existence of optimal dual points.

$$\begin{aligned}
p^* &= p(0, 0) = d^* \\
&= \sup_{\lambda \geq 0} g(\lambda, \nu) \\
&= g(\lambda^*, \nu^*) \\
&= \inf_x \mathcal{L}(x, \lambda^*, \nu^*) \\
&= \inf_x f_0(x) + \sum_{\lambda_i^* \geq 0} \underbrace{f_i(x)}_{\leq u_i} + \sum \nu_i^* \underbrace{h_i(x)}_{v_i} \\
&\leq f_0(\bar{x}) + \langle \lambda^*, u \rangle + \langle \nu^*, v \rangle \\
p(u, v) &= f_0(\bar{x}) \geq p^* - \langle \lambda^*, u \rangle - \langle \nu^*, v \rangle
\end{aligned}$$

where we chose an \bar{x} that is feasible for $(P_{u,v})$.

Case (1). λ_i^* is large, then if we tighten i th inequality constraint ($u_i < 0$),

$$p(u, 0) \geq p^* - \lambda_i u_i$$

So $p(u, 0)$ is much larger than p^* .

Case (2). λ_i^* is small, and we loosen i th inequality constraint $u_i > 0$, then

$$p^* \geq p(u, 0) \geq p^* - \underbrace{\lambda_i^* u_i}_{>0}.$$

So loosening constraint doesn't help much to reduce the minimum.

Case (3). $\lambda_i^* = 0$, for example when $f_i(x) \leq 10^6$, the constraint is inactive.

Case (4). When λ_i^* is large, loosen $(u_i) > 0$, or if $\lambda_i^* = 0$, tighten $(u_i < 0)$, then the analysis can't help us.

Case (5). If $\nu_i^* \gg 0, v_i < 0$ or $\nu_i^* \ll 0, v_i > 0$, then $p(0, v) \gg p^*$ and we have a big change.

Case (6). If $|v_i^*| \ll 1$ or $\nu_i^* > 0, v_i < 0$ or $\nu_i^* < 0, v_i > 0$, then $p(0, v)$ doesn't change much.

local sensitivity analysis

We can show that $(P_{u,v})$ is convex using the minimizing conditions. Recall that for a convex function,

$$f(y) \geq f(x) + \langle \partial f(x), y - x \rangle.$$

Comparing with Equation above we see that the dual variables are just the subgradients of $p(u, v)$! If $p(u, v)$ is differentiable,

$$\frac{\partial p(0,0)}{\partial u_i} = -\lambda_i^*, \quad \frac{\partial p(0,0)}{\partial v_i} = -\nu_i^*.$$

This is symmetric! Now we can write the Taylor expansion:

$$p(u, v) = p(0, 0) + \langle \frac{\partial p}{\partial u}, u \rangle + \langle \frac{\partial p}{\partial v}, v \rangle + \text{higher-order terms}.$$

This is only accurate with small perturbation.

In economics, dual variables is referred to as "shadow prices". In statistics, it's called "score test".

1.6.9 Generalized Inequalities [BV 5.9]

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, f_i : \mathbb{R}^n \rightarrow S^m \\ & h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

Example 1.6.35 (SDP)

Punchline: we get analogous KKT conditions. Instead of $\lambda_i \geq 0$ now we require $\Lambda_i \succeq 0$.

Caveat: Before, if $\lambda \geq 0, y \geq 0$ and $\langle \lambda, y \rangle = 0$, then $\forall i, \lambda_i = 0$ or $y_i = 0$. However, in the matrix case, if $\Lambda \succeq 0, Y = f_i(x) \succeq 0$ and $\langle \Lambda, Y \rangle \geq 0$, it does NOT mean $Y = 0$ or $\Lambda = 0$.

But if $\Lambda \succ 0, Y \succeq 0, \langle \Lambda, Y \rangle = 0$, then $Y = 0$

Chapter 2

Algorithms

2.1 Unconstrained Optimization

We assume reasonable smoothness of the objective. Here is an overview of the algorithms:

- (1) gradient descent

$$x_{k+1} = x_k - t \cdot \nabla f(x_k), \quad t = \frac{1}{L}.$$

where L is the Lipschitz constant of the gradient.

- (2) Newton's method

$$x_{k+1} = x_k - (\nabla^2 f(x_k))^{-1} \nabla f(x_k).$$

This can reduce to gradient descent when we have $\nabla^2 f(x) \preceq L \cdot I$ and we just bound the Hessian with $L \cdot I$.

- (3) Quasi-Newton

2.1.1 Proximal gradient descent

$$\min \underbrace{f(x)}_{\text{smooth, strongly convex}} + \underbrace{g(x)}_{\text{simple, convex}}$$

Note we can add indicator function to g :

$$g(x) = I_C(x) + h(x),$$

i.e. when we have constraint $x \in C$.

motivation

We can try the first-order Taylor approximation of f . However, recall minimizing a linear function would go to negative infinity, so we need to go to 2nd order.

$$\begin{aligned} x_{k+1} &= \operatorname{argmin}_x f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}L\|x - x_k\|_2^2 + g(x) \\ &= \operatorname{argmin}_x \frac{1}{L} \left(\langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2}L\|x - x_k\|_2^2 + g(x) \right) \\ &= \operatorname{argmin}_x \frac{1}{2} \left\| x - \left(x_k - \frac{1}{L} \nabla f(x_k) \right) \right\|_2^2 + \frac{1}{L} \cdot g(x) \quad \text{complete the square and ignore constants} \\ &= \operatorname{argmin}_x \frac{1}{2} \|x - \tilde{x}\|_2^2 + \frac{1}{L} \cdot g(x) \\ &= \operatorname{prox}_{\frac{1}{L} \cdot g}(\tilde{x}) \end{aligned}$$

Note that this solution is unique because we have strong convexity.

algorithm

$$x_{k+1} = \operatorname{prox}_{tg}(x_k - t \cdot \nabla f(x_k)) \quad t \text{ via line search or } t = \frac{1}{L}$$

Remark 2.1.1 If $g(x) = 0$, proximal operator is the identity function so it reduces to gradient descent.

Example 2.1.2

$g(x) = I_C$. Then

$$\text{prox}_{tg}(\tilde{x}) = \text{Proj}_C(x)$$

Recall from linear algebra: if $\text{Proj}_V(\tilde{x})$ is the projection of $\tilde{x} \rightarrow V$, then

$$\tilde{x} = \text{Proj}_V(\tilde{x}) + \text{Proj}_{V^\perp}(\tilde{x}).$$

We can generalize this result to **Moreau's decomposition**:

$$\tilde{x} = \text{prox}_g(\tilde{x}) + \text{prox}_{g^*}(\tilde{x})$$

Example 2.1.3

$$\text{Proj}_{\|x\|_\infty \leq 1} = \tilde{x} - \text{prox}_{\|\cdot\|_1}(\tilde{x}).$$

$$\text{prox}_{t\|\cdot\|_1}(y) = \underset{x}{\text{argmin}} \frac{1}{2}\|x - y\|_2^2 + L\|x\|_1 \text{ this is separable!}$$

By Fermat's rule,

$$\text{prox}_g(y) = \underset{x}{\text{argmin}} \frac{1}{2}\|x - y\|^2 + g(x)$$

$$\Rightarrow 0 \in x - y + \partial g(x)$$

$$y \in x + \partial g(x)$$

$$y \in (I + \partial g)(x)$$

$$x \in (I + \partial g)^{-1}(y)$$

$$x = (I + \partial\|\cdot\|_1)^{-1}y \text{ unique solution s.c.}$$

We derived earlier that the solution to $\text{prox}_{t\|\cdot\|_1}$ is

$$x = \text{sgn}(y) \cdot [|y| - t]_+.$$

alternative derivation

By Fermat

$$0 \in \partial(f + g)(x)$$

$$0 \in \nabla f(x) + \partial g(x) \text{ under CQ}$$

$$x = x + \nabla f(x) + \partial g(x)$$

$$x - \nabla f(x) \in x + \partial g(x) = (I + \partial g)(x)$$

$$x = (I + \partial g)^{-1} (I - \nabla f)(x) \text{ fixed point eqn}$$

$$\begin{aligned} x_{k+1} &= (I + \partial g)^{-1} (I - \nabla f)(x_k) \\ &= \text{prox}_g(x_k - \nabla f(x_k)) \end{aligned}$$

If $f = 0$, we get

$$\begin{aligned} x_{k+1} &= \text{prox}_{t g}(x_k) \text{ here } t \text{ is anything we want since } f = 0 \\ &= \argmin_t t \cdot g(x) + \frac{1}{2} \|x - x_k\|^2 \end{aligned}$$

Remark 2.1.4 Forward Euler exactly corresponds to gradient descent, whereas backward Euler exactly corresponds to proximal gradient descent. Thus, proximal gradient descent is also called "forward-backward method".

2.1.2 Gradient descent

Problem: we want to solve $\min_x f(x)$, $f : \mathbb{R}^n \rightarrow \mathbb{R}$, $f \in \Gamma_0(\mathbb{R}^n)$ (proper, lsc, convex) and ∇f is L -Lipschitz continuous (strongly smooth).

Attempt 1

$$x_{k+1} = \argmin_x \left[f(x_k) + \underbrace{\langle \nabla f(x_k), x - x_k \rangle}_{q_k(x) \text{ 1st order surrogate}} \right].$$

Linearization is a common trick to simplify problems. However, this fails because $\min_x q_k(x) = -\infty$ for a linear function (unless it's already optimal). We can fix this by add a compact constraint. Then it's called **Frank-Wolfe** or **conditional gradient**. We omit this discussion as it's a bit niche.

Attempt 2

Consider the 2nd order Taylor series:

$$x_{k+1} = \operatorname{argmin}_x \underbrace{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} \langle x - x_k, \nabla^2 f(x_k)(x - x_k) \rangle}_{q_k(x) \text{ quadratic surrogate}}.$$

Since f is convex, $\nabla^2 f(x) \succeq 0 \Rightarrow q_k(x)$ is a convex quadratic (sum of convex functions).

To minimize $q_k(x)$, we use Fermat's rule:

$$\begin{aligned} 0 &= \nabla q_k(x) \\ &= \nabla f(x_k) + \nabla^2 f(x_k)(x - x_k) \quad \text{gradients of linear and quadratic terms} \\ x_{k+1} &= x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k) \end{aligned}$$

This is **Newton's method**, a generalization of the "Newton-Raphson" for 1D root-finding, applied to the gradient. It is a **2nd-order method** because it involves the derivative of the gradient which is the second derivative (Hessian).

Remark 2.1.5 Unlike 1D root-finding, 2nd order methods in higher dimensions converge quickly but each iteration may be costly because we need to invert the Hessian and solving system of equations. This is about $\mathcal{O}(n^3)$. 1st-order methods only use $\nabla f(x)$ and usually converge more slowly but each step is cheap at about $\mathcal{O}(n)$.

What to use?

It depends:

- Structure matters (is $\nabla^2 f$ easy to invert? Is it ill-conditioned (which hurts 1st order more)?)
- For small/medium problem size, high accuracy, we use 2nd order. This is default for cvx/cvxpy.
- In between problems: try both?

Other types

- 3rd order: usually not worth the complexity. See recent Nesterov work for a plausible implementation.
- 0th order: Extremely slow and finding gradient is cheap anyway, usually not worth it.
- coordinate descent: heavily depends on the structure.

Attempt 3

By assumption, $0 \preceq \nabla^2 f(x) \preceq LI$. Thus, for all y ,

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

This allows us to upper bound the quadratic surrogate and simplify it further by removing the Hessian. Notice $(LI)^{-1} = \frac{1}{L}I$ which replaces $(\nabla^2 f)^{-1}$. So we can modify Newton's method as

$$\begin{aligned} x_{k+1} &= \operatorname{argmin}_x \underbrace{f(x_k) + \langle \nabla f(x_k), x - x_k \rangle + \frac{1}{2} L \|x - x_k\|^2}_{q_k(x)} \\ &= x_k - \frac{1}{L} \nabla f(x_k) \end{aligned}$$

This is $\mathcal{O}(n)$. Here $q_k(x) \geq f(x) \forall x$ is more than a linearization but is less than the full 2nd order Taylor expansion. It is a **majorizer** of f .

fig

Majorization-minimization (MM)

MM can always guarantee making progress on the minimization. The framework is

- 1) Assume we can always construct a majorizer q_k s.t.

$$(i) \quad \forall x, f(x) \leq q_k(x)$$

$$(ii) \quad f(x_k) = q_k(x_k)$$

- 2) Iterate: $x_{k+1} \in \operatorname{argmin}_x q_k(x)$.

This algorithm is a **descent algorithm**. That is, it never makes things worse.

Proof.

$$\begin{aligned} f(x_{k+1}) &\leq q_k(x_{k+1}) && \text{by (i)} \\ &\leq q_k(x_k) && \text{by 2)} \\ &= f(x_k) && \text{by (ii)} \end{aligned}$$

□

Usually we might eventually show that (no convexity needed):

- If $f(x)$ is bounded below, then $f(x_k)$ converges by MCT.
- If (x_k) converges and f is lsc, then the limit $x_k \rightarrow x$ is a stationary point *i.e.* $\nabla f(x) = 0$.

Example 2.1.6 (usually non-convex)

- 1) Expectation maximization (EM) for maximum-likelihood estimation.
- 2) Difference of convex functions (DC) or convex + concave:

$$f(x) = g(x) - h(x),$$

where g, h are both convex. Although $-h$ is concave, $-h$ is majorized by its tangent line which is convex. Then

$$q_k(x) = g(x) - \underbrace{(h(x_k) + \langle \nabla h(x_k), x - x_k \rangle)}_{\text{affine in } x}$$

is a majorizer, and $q_k(x)$ is convex.

The takeaway is that not all non-convex problems are equally hard.

Convergence of gradient descent

Theorem 2.1.7

Consider the problem

$$f^* = \min_x f(x).$$

$f \in \Gamma_0(\mathbb{R}^n)$. We assume ∇f is L -Lipschitz continuous. Choose $t = \frac{1}{L}$.

Then gradient descent with step size t converges with rate $\mathcal{O}\left(\frac{1}{k}\right)$.

Proof. We wish to bound $f(x_{k+1}) - f^*$ by the local linear and quadratic lower and upper bounds. L -Lipschitz continuous implies that $\nabla^2 f(x) \preceq L \cdot I$. Recall that $x_{k+1} = x_k - \frac{1}{L} \nabla f(x_k)$.

$$\begin{aligned} f(x_{k+1}) &\leq f(x_k) + \langle \nabla f(x_k), x_{k+1} - x_k \rangle + \frac{L}{2} \|x_{k+1} - x_k\|^2 \\ &= f(x_k) - \frac{1}{2L} \|\nabla f(x_k)\|^2 \text{ descent method guarantees progress} \\ &\leq f^* + \langle \nabla f(x_k), x_k - x^* \rangle - \frac{1}{2L} \|\nabla f(x_k)\|^2 \text{ by convexity} \\ &= f^* + \frac{L}{2} \left(\|x_k - x^*\|^2 - \left\| x_k - x^* - \frac{1}{L} \nabla f(x_k) \right\|^2 \right) \\ &= f^* + \frac{L}{2} \left(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2 \right) \\ \sum_{i=1}^k f(x_i) - f^* &= \frac{L}{2} \sum_{i=1}^k \|x_{i-1} - x^*\|^2 - \|x_i - x^*\|^2 \\ &= \frac{L}{2} (\|x_0 - x^*\|^2 - \|x_k - x^*\|^2) \text{ telescope} \\ &\leq \frac{L}{2} \|x_0 - x^*\|^2 \\ f(x_k) - f^* &\leq \frac{1}{k} \sum_{i=1}^k f(x_i) - f^* \leq \frac{L}{2k} \|x_0 - x^*\|^2 \\ &= \mathcal{O}\left(\frac{1}{k}\right) \end{aligned}$$

□

Question: is this the best we can?

- (1) Is our analysis tight? Yes.
- (2) This is worst-case complexity.
- (3) Are there similar methods (*i.e.* first-order) with faster rates? More pre-

cisely, first-order method satisfies (Lanczos/CG):

$$x_k \in \mathcal{L}_k = \text{span}\{x_0, \nabla f(x_0), \nabla f(x_1), \dots, \nabla f(x_{k-1})\}.$$

The answer is yes, by Nesterov 1983.

Theorem 2.1.8 (Nesterov 1983)

For any 1st order method, there exists a $f \in \Gamma_0(\mathbb{R}^n)$ with ∇f L -Lipschitz continuous and

$$f(x_k) - f^* \geq \frac{3}{32} \cdot \frac{L}{k^2} \cdot \|x_0 - x^*\|^2 \text{ for } k \leq \frac{1}{2}(n-1)$$

and

$$x_k - x^* \geq \frac{1}{8} \|x_0 - x^*\|^2$$

Proof. Sketch: The adversarial function is

$$f(x) = \frac{L}{4} (\langle x, Ax \rangle - \langle e_1, x \rangle), A = \begin{pmatrix} 2 & -1 & 0 & \dots \\ -1 & 2 & -1 & 0 \\ \dots & & & \\ 0 & 0 & -1 & 2 \end{pmatrix}$$

$$\nabla f(x) = \frac{L}{4} (Ax, e_1)$$

$$x^* = A^{-1} e_1$$

Assume $x_0 = 0$ (we can shift). At x_k , only first k coordinates are nonzero. Since A^{-1} is a dense matrix, so x^* has nonzero elements, so we can get a high norm difference. \square

Theorem 2.1.9 (Nesterov)

$$y_0 = x_0$$

$$x_{k+1} = y_k - t_k \nabla f(y_k)$$

$$y_{k+1} = x_{k+1} + \frac{k}{k+3} (x_{k+1} - x_k)$$

This has convergence rate of $\mathcal{O}\left(\frac{1}{k^2}\right)$.

Remark 2.1.10 Since we cannot get better than $O\left(\frac{1}{k^2}\right)$ and this algorithm achieves it, so it is optimal.

Gradient descent analysis with strong convexity

What do we want to analyze? Error metrics.

- (1) $f(x_{k+1}) - f_{x_k}, \|x_{k+1} - x_k\|, \|\nabla f(x_k)\|$: can always be practical termination criteria, although they might not be good.
- (2) $f(x_k) - f^*$: we can use this sometimes if we know $f^* = 0$ or in the primal/dual problem when we can squeeze the gap between bounds.
- (3) $\|x_k - x^*\|$: often can't use.

Remark 2.1.11 $f(x_k) = \sum_{j=1}^k \frac{1}{j}$, then $f(x_{k+1}) - f(x_k) \rightarrow 0$ but we don't have a minimum since the series diverge!

suboptimality bounds (see PDF handout)

If ∇f is L -Lipschitz continuous, then

- (1) $\|\nabla f(x)\| \leq L\|x - x^*\|_2$.
- (2) $f(x) - f^* \leq \frac{L}{2}\|x - x^*\|^2$ by continuity.
- (3) $\|\nabla f(x)\|^2 \leq 2L(f(x) - f^*)$.

Remark 2.1.12 By these bounds, bounding $\|x - x^*\|$ is the nicest if possible but usually out of reach. The next nicest to bound is $f(x) - f^*$. "x is an ε -sub-optimal point" means $f(x) - f^* \leq \varepsilon$.

If f is μ -strongly convex, then

- (1) $\|x - x^*\|^2 \leq \frac{2}{\mu}(f(x) - f^*)$.
- (2) $\|x - x^*\| \leq \frac{1}{\mu}\|\nabla f(x)\|$.
- (3) Polyak-Lojasiewicz (PL): $f(x) - f^* \leq \frac{1}{2\mu}\|\nabla f(x)\|^2$.

Recall from last time, we derive

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2L} \|\nabla f(x_k)\|^2.$$

So if we add μ -strongly convex to the assumption of gradient descent analysis, then

$$f(x_{k+1}) - f(x_k) \leq -\frac{1}{2L} \|\nabla f(x_k)\|^2 \leq -\frac{\mu}{L} (f(x_k) - f^*) \text{ by PL}$$

$$f(x_{k+1}) \leq f(x_k) - \frac{\mu}{L} (f(x_k) - f^*)$$

$$f(x_{k+1}) - f^* \leq \left(1 - \frac{\mu}{L}\right) (f(x_k) - f^*)$$

Since $\mu I \preceq \nabla^2 f \preceq LI$. So $0 < \rho := \frac{\mu}{L} < 1$. By contraction mapping theorem, this converges.

$$\|x_k - x^*\|^2 \leq \frac{2}{\mu} \rho^k (f(x_0) - f^*)$$

Remark 2.1.13 $\kappa = \frac{L}{\mu}$ is the condition number of the Hessian, *i.e.* the largest singular value over the smallest.

Convergence rate

rate	iteration number	example
$\mathcal{O}\left(\frac{1}{k^{1/4}}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon^4}\right)$	non-convex subgradient method
$\mathcal{O}\left(\frac{1}{\sqrt{k}}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$	subgradient descent or SGD
$\mathcal{O}\left(\frac{1}{k}\right)$	$\mathcal{O}\left(\frac{1}{\varepsilon}\right)$	gradient-descent with Lipschitz
$\mathcal{O}\left(\frac{1}{k^2}\right)$	$\mathcal{O}\left(\frac{1}{\sqrt{\varepsilon}}\right)$	Nesterov acceleration
$\mathcal{O}(\rho^k)$	$\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)$	gradient descent with Lipschitz and strong convexity
$\mathcal{O}\left(\rho^{2^k}\right)$	$\log_2\left(\mathcal{O}\left(\log\left(\frac{1}{\varepsilon}\right)\right)\right)$	Newton's method locally

2.1.3 Linear conjugate gradient method

CG solves (usually approximately) $Ax = b$ if $A \succ 0$. More details and intuition can be found at cs.cmu.edu/~quake-papers/painless-conjugate-gradient.pdf.

Example 2.1.14

Consider the least squares problem. Let $\phi(x) = \frac{1}{2}x^T \tilde{A}^T \tilde{A}x - \tilde{b}^T \tilde{A}x + \frac{1}{2}\tilde{b}^T \tilde{b} =: \frac{1}{2}x^T Ax - b^T x + \text{const.}$. Here $A \succeq 0$ always and $A \succ 0$ if $m > n$ and full rank. Then we can solve $\nabla \phi(x) = Ax - b$.

Note. Don't form the Gram matrix. Instead use LSQR method.

One idea to min $\phi(x)$ is coordinate descent/alternating minimization. This slowly converges to the solution via a zigzag path. If we change to the eigenvector basis, it is guaranteed to converge in n steps. However, finding the eigenvector basis is just as expensive as solving the normal equation directly at $\mathcal{O}(n^3)$.

Definition 2.1.15 (conjugate directions) — $\{p_i\}$ are conjugate directions if they are A -orthogonal. That is,

$$\langle p_i | A | p_j \rangle := \langle p_i, Ap_j \rangle = 0 \text{ if } i \neq j.$$

Note. If we have $\{p_i\}_{i=0}^{n-1}$, it's a basis. If p_i are eigenvectors of a symmetric matrix A , then they are A -orthogonal.

Our goal is to find $\{p_i\}$ more cheaply than eigenvectors.

Theorem 2.1.16 (conjugate direction method (abstract))

Assume $\{p_i\}_{i=0}^{n-1}$ are conjugate directions. Then

$$x_{k+1} = x_k + \alpha_k p_k$$

where a_k solves $\min_{\alpha} \phi(x_k + \alpha p_k)$ which is exact line search. The solution to this 1D problem has a closed form:

$$a_k = -\frac{\langle r_k | p_k \rangle}{\langle p_k | A | p_k \rangle}$$

where $r_k = Ax_k - b$. Then $x_n = x^*$.

Proof. Since $\{p_i\}$ is a basis, we can write

$$\begin{aligned} x^* - x_0 &= \sum_{i=0}^{n-1} \sigma_i p_i \\ p_k^T A(x^* - x_0) &= \sum_{i=0}^{n-1} \sigma_i \langle p_k | A | p_i \rangle \\ &= \sigma_k \langle p_k | A | p_k \rangle \\ \sigma_k &= \frac{\langle p_k | A | x^* - x_0 \rangle}{\langle p_k | A | p_k \rangle} \quad \forall k \end{aligned}$$

Moreover,

$$\begin{aligned} x_k - x_0 &= \sum_{i=0}^{k-1} \alpha_i p_i \\ p_k^T A(x_k - x_0) &= \sum_{i=0}^{k-1} \alpha_i \langle p_k | A | p_i \rangle \\ \langle p_k | A | x_k - x_0 \rangle &= 0 \end{aligned}$$

Substituting x_k as x_0 ,

$$\sigma_k = \frac{\langle p_k | A | x^* - x_k \rangle}{\langle p_k | A | p_k \rangle} = \alpha_k$$

Therefore, $x_n = x^*$ since they have the same expression in the basis. \square

Remark 2.1.17 We can think of this process as either building up x^* component-by-component or cutting the error $x^* - x_k$ component-by-component.

Facts:

- $r_{k+1} = r_k + \alpha_k A p_k$
- $\langle r_k, p_i \rangle = 0, i < k$.
- x_k minimizes ϕ over $K(r_0, k) = \text{span}\{r_0, A r_0, \dots, A^{k-1} r_0\}$. This is the Krylov subspace. Note that $K(r_0, n-1) = \mathbb{R}^n$.
- $\langle r_k, r_i \rangle = 0, i < k$.
- $p_k, r_k \in K(r_0, k)$.

Theorem 2.1.18 (conjugate gradient)

Given arbitrary $x_0, r_0 = Ax_0 - b, p_0 = -r_0$. Compute iteratively

$$\beta_k = \frac{\langle r_k | A | p_{k-1} \rangle}{\langle p_{k-1} | A | p_{k-1} \rangle}, \text{ chosen s.t. } \langle p_k | A | p_{k-1} \rangle = 0$$

$$p_k = -r_k + \beta_k p_{k-1}$$

$$\alpha_k = -\frac{\langle r_k | p_k \rangle}{\langle p_k | A | p_k \rangle}$$

$$x_{k+1} = x_k + \alpha_k p_k$$

$$r_{k+1} = r_k - \alpha_k A p_k$$

The magic is that $\langle p_k | A | p_i \rangle = 0 \forall i \leq k-1$ (see Nocedal and Wright for proof).

The cost is one matrix-vector multiply per step.

Theorem 2.1.19 (convergence of CG)

$$\|x_k - x^*\|_A \leq 2 \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \|x_0 - x^*\|_A$$

where $\kappa = \kappa(A)$ is the condition number of A .

2.1.4 non-linear conjugate gradient

Recall that in linear CG, $\phi(x)$ is quadratic and $\nabla \phi(x) = Ax - b = r$ the residual.

In non-linear, neither holds true, but we can replace the parts that no longer

hold with more general expressions:

Theorem 2.1.20 (non-linear CG)

$$a_k \approx \underset{\alpha}{\operatorname{argmin}} \phi(x_k + \alpha p_k) \quad \text{linesearch}$$

$$x_{k+1} = x_k + \alpha_k p_k$$

$$\beta_{k+1}^{FR} = \frac{\|\nabla \phi_{k+1}^2\|}{\|\nabla \phi_k\|^2}$$

$$p_{k+1} = -\nabla \phi_{k+1} + \beta_{k+1} p_k$$

Remark 2.1.21

- it can be fast, but has to be unconstrained optimization. It might work very well on "almost quadratic" functions.
- too sensitive to parameters: Hager+Zhang has the most advanced version which uses a lot of parameters.
- quasi-Newton methods can be just as fast and more stable.

2.1.5 Quasi-Newton Methods

Remark 2.1.22 This is the gold-standard for large-scale, smooth, unconstrained optimization.

Definition 2.1.23 (quadratic approximation) —

$$m_k(p) = f_k + \langle \nabla f_k, p \rangle + \frac{1}{2} \langle p | B_k | p \rangle, \quad B_k \succ 0$$

$$\tilde{x}_{k+1} = x_k + p_k, \quad p_k = \underset{p}{\operatorname{argmin}} m_k(p) = -B_k^{-1} \nabla f_k$$

Sometimes we do a linesearch, $x_{k+1} = x_k + \alpha p^*$.

Example 2.1.24

Gradient descent is when $B_k = L \cdot I$.

Example 2.1.25

Newton's method is when $B_k = \nabla^2 f(x_k)$.

Theorem 2.1.26 (Quasi-Newton)

Using the quadratic approximation framework, choose B_k s.t.

- (1) B_k is more accurate than $L \cdot I$
- (2) B_k is cheaper than Newton (in terms of forming and inverting).

The main trick is to construct B_{k+1} by updating B_k .

To find x_{k+2} ,

$$m_{k+1}(p) = f_{k+1} + \langle \nabla f_{k+1}, p \rangle + \frac{1}{2} \langle p | B_{k+1} | p \rangle$$

$$\nabla m_{k+1}(p) = \nabla f_{k+1} + B_{k+1} p$$

Thus, $\nabla m_{k+1}(0) = \nabla f_{k+1}$.

Then the following conditions are automatically satisfied:

- (1) $m_{k+1}(0) = f_{k+1}$
- (2) $\nabla m_{k+1}(0) = \nabla f_{k+1}$

This gives us freedom in choosing B_{k+1} . Let $s_k = x_{k+1} - x_k$ and $y_k = \nabla f_{k+1} - \nabla f_k$. Moreover, let's impose

$$\begin{aligned}\nabla m_{k+1}(-s_k) &= \nabla f_{k+1} - B_{k+1}s_k = \nabla f_k \\ B_{k+1}s_k &= y_k \quad \text{secant equation}\end{aligned}$$

Most Quasi-Newton methods choose B_{k+1} to satisfy the secant equation.

Notice that we want $B_{k+1} \succ 0$. Then $\langle s_k | B_k | s_k \rangle = \langle s_k, y_k \rangle > 0$. This is a necessary condition called **curvature condition**. That is,

$$\langle x_{k+1} - x_k, \nabla f_{k+1} - \nabla f_k \rangle > 0.$$

This is strictly monotone, which is satisfied when f is strictly convex. If f isn't strictly convex, it would complicate quasi-Newton method (*e.g.* might need to add a line search).

$x \in \mathbb{R}^n, B \in \mathbb{R}^{n \times n}$, so B has $n(n+1)/2$ degrees of freedom, and n constraints from the secant equation.

Example 2.1.27

When $n = 1$, degree of freedom and constraint are both 1, B_{k+1} is completely determined. This is called the *secant method*.

When $n > 1$, B is underdetermined. The standard ways to choose B is

$$\begin{aligned}B_{k+1} &= \operatorname{argmin}_{B \succ 0, B s_k = y_k} \|B - B_k\|_w^2 \\ \text{or } B_{k+1}^{-1} &= \operatorname{argmin}_{B^{-1} \succ 0, B^{-1} y_k = s_k} \|B^{-1} - B_k^{-1}\|_w^2\end{aligned}$$

where $\|\cdot\|_w$ is some norm chosen so the problem has a closed-form solution. The class of methods on choosing B is called the **Broyden class**.

Notation. B approximates $\nabla^2 f$, and H approximates $(\nabla^2 f)^{-1}$. That is, $B_k^{-1} = H_k$.

Observe that it's cheaper to just approximate the inverse Hessian, although it is actually not an issue because B_{k+1} is a low-rank update of B_k , so we can use Sherman-Morrison-Woodbury formula to obtain the inverse very cheaply.

BFGS

Using a specific weighted norm that satisfies the secant equation:

Theorem 2.1.28 (BFGS)

$$H_{k+1} = (I - \rho_k s_k y_k^T) H_k (I - \rho_k y_k s_k^T) + \rho_k s_k s_k^T, \quad \rho_k = \frac{1}{\langle y_k, s_k \rangle}$$

Remark 2.1.29

- Iteration count: win (compared to gradient descent)
- Flops: $\mathcal{O}(n^2) < \mathcal{O}(n^3)$: win (compared to Newton)
- Memory: $\mathcal{O}(n^2)$: loss (same as Newton)

Theorem 2.1.30 (L-BFGS)

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T$$

$$H_{k+1}(w) = V_k^T H_k (V_k w) + \rho_k s_k s_k^T w$$

where $z := V_k w$ uses y_k, s_k and $s_k^T w$ uses y_k, s_k . Both are cheap. Then $H_k(z)$ depends on $y_{k-1}, s_{k-1}, H_{k-1}$. We can do this recursively down to the first. Instead we stop at $(k-m)$ th term. That is,

$$H_{k-m} = \frac{\langle y_{k-m}, s_{k-m} \rangle}{\|y_{k-m}\|^2} \cdot I$$

We can start with gradient descent to initialize. That is, $H_0 = \frac{1}{L} \cdot I$.

Then the storage becomes $2(m+1)n$. Commonly choose $m \in \{3, 20\}$.

Remark 2.1.31 Usually $B_k \not\rightarrow \nabla^2 f(x^*)$.

Theorem 2.1.32 (convergence)

If $0 < \mu I \leq \nabla^2 f(x) \leq L \cdot I$, then BFGS converges and usually superlinearly.

Open question: if f is non-convex, does BFGS converge to a stationary point?

Remark 2.1.33 If $m = 0$ "memoryless" BFGs plus exact linesearch yields nonlinear CG.

Remark 2.1.34 What if we have constraints? Recall that for gradient descent we can do proximal/projected gradient descent. That is,

$$x_{k+1} = \text{Proj}(x_k - t \cdot \nabla f(x_k)).$$

Can we do the same thing for any quasi-Newton method?

$$x_{k+1} = \text{Proj}_{B_k}(x_k - B_k^{-1} \nabla f_k).$$

This is usually not feasible since the scaled projection is hard to compute.

2.1.6 Newton's methods

Let $\Delta x = \nabla^2 f(x_k)^{-1} \nabla f(x_k)$.

- (1) computational: "inexact-Newton", "matrix-free", "truncated-Newton", or "Newton-CG" mean approximate Δx . That is, we wish to solve

$$\nabla^2 f(x_k) \cdot \Delta x = \nabla f(x_k)$$

We can solve this with linear CG with only a few steps (adaptive). We can use preconditioners such as incomplete Cholesky or BFGS.

Often Hessian is structured and we can exploit that in computing the Hessian-gradient product.

- (2) convergence:

In practice, we use a linesearch or even better a trust-region to "globalize". We wish to avoid bad saddle points.

For trust-region, we minimize a quadratic model.

Trust-region for nonconvex

$$\begin{aligned} x_{k+1} &= x_k + p_k \\ &= x_k + \underset{p}{\operatorname{argmin}} \langle \nabla f_k, p \rangle + \frac{1}{2} \langle p, B_k, p \rangle \end{aligned}$$

If $B_k \succ 0$, then this is a convex quadratic, the gradient equals 0 is the sufficient condition. Then the Newton step is the minimizer $p_k = B_k^{-1} \nabla f_k$. If not, Newton step isn't the minimizer. Now we have to use trust-region.

$$\begin{aligned} x_{k+1} &= x_k + \underset{p}{\operatorname{argmin}} \langle \nabla f_k, p \rangle + \frac{1}{2} \langle p, B_k, p \rangle \\ \text{s.t. } \|p\| &\leq \Delta \Leftrightarrow \frac{1}{2} \|p\|^2 \leq \frac{1}{2} \Delta^2 \end{aligned}$$

Remark 2.1.35 If B_k is indefinite, we see that if we pretend the gradient term isn't there, then the quadratic form is minimized by the leftmost eigenvector (associated with the negative-most eigenvalue) of B_k , scaled to the trust region radius. See N+W for tricks to solve.

The KKT conditions are necessary.

$$\mathcal{L}(p, \lambda) = \langle \nabla f_k, p \rangle + \frac{1}{2} \langle p | B_k | p \rangle + \frac{1}{2} \lambda (\|p\|^2 - \Delta^2)$$

Stationarity:

$$\begin{aligned} \nabla f_k + B_k p + \lambda p &= 0 \\ p &= \underbrace{(B_k + \lambda I)^{-1}}_{\text{regularity}} \nabla f_k \end{aligned}$$

Remark 2.1.36 Typical: trust-region methods can sometimes guarantee a local minimizer even if the problem isn't convex.

Remark 2.1.37 Alternatively, we can use

- cubic regularization
- perturbed gradient descent

2.1.7 Nonlinear least squares

The objective is

$$\begin{aligned} f(x) &= \frac{1}{2} \sum_{j=1}^m r_j^2(x), \quad r_j : \mathbb{R}^n \rightarrow \mathbb{R} \\ &:= \frac{1}{2} \|R(x)\|^2, \quad R : \mathbb{R}^n \rightarrow \mathbb{R}^m \end{aligned}$$

This is perhaps the most common in engineering and sciences. We use squares here not only because it's easier to differentiate but also because if our data have Gaussian noise, then this becomes the maximum likelihood estimation.

Let the Jacobian of R be $J(x)$.

$$\begin{aligned} J(x)_{i,j} &= \frac{\partial r_i}{\partial x_j} \\ J(x) &= \begin{pmatrix} \nabla r_1(x)^T \\ \vdots \\ \nabla r_m(x)^T \end{pmatrix} \\ \nabla f(x) &= \nabla \left(\frac{1}{2} \sum_{j=1}^m r_j^2(x) \right) \\ &= \frac{1}{2} \sum_{j=1}^m \nabla(r_j^2(x)) && \text{linearity} \\ &= \frac{1}{2} \sum_{j=1}^m 2r_j(x) \nabla r_j(x) \\ &= J(x)^T R(x) \\ \nabla^2 f(x) &= J(x)^T J(x) + \sum_{j=1}^m r_j(x) \cdot \nabla^2 r_j(x) \end{aligned}$$

In the least squares case $r_j(x) = a_i^T x - b_i$, we see that $\nabla^2 f(x) = A^T A$ just as we expect.

Gauss-Newton method

$$x_{k+1} = x_k - B_k^{-1} \nabla f_k$$

where $B_k = J_k^T J_k$. So we ignore the sum term to approximate the Hessian. This is worse than Newton but better than gradient descent's constant $L \cdot I$

approximation, and we get the approximation "for free" as we need to compute $J(x)$ for the gradient anyway. Although inverting it can get expensive.

Another derivation:

$$\begin{aligned}
 R(x) &\approx R(x_k) + J(x_k)(x - x_k) && \text{1st order Taylor} \\
 f(x) &= \frac{1}{2} \|R(x)\|^2 \\
 &\approx \frac{1}{2} \|R_k + J_k(x - x_k)\|^2 && \text{linear ls model!} \\
 x_{k+1} &= (J_k^T J_k)^{-1} J_k^T (J_k x_k - R_k) && \text{normal eq} \\
 &= x_k - (J_k^T J_k)^{-1} J_k^T R_k \\
 &= x_k - (J_k^T J_k)^{-1} \nabla f_k
 \end{aligned}$$

Levenberg-Marquardt

This is Gauss-Newton with a trust-region.

Softwares:

- Matlab: lsqnonlin
- python: scipy.optimize.least_squares, lmfit (modeling)

2.2 Methods for constrained problems

Special nice constraints

e.g. $x \geq 0, \ell_i \leq x_i \leq u_i, \|x - x_0\| \leq 3$

- (1) projected gradient descent (with Nesterov acceleration).
- (2) active-set methods (*e.g.* L-BFGS-B).

Example 2.2.1

$x \in \mathbb{R}^2, 0 \leq x_i \leq 1$. Suppose $x_1^k = 1, x_2^k = .3$. Then for $k+1$, let $x_1^{k+1} = x_1^k$ unchanged, and ignore constraint for x_2 and use $L-BFGS$ and check it satisfies.

not-so-nice constraints

e.g. $g(x) \leq 0$ where g is non-linear.

- (1) penalty methods: both convex and non-convex
- (2) augmented Lagrangian: both
- (3) sequential quadratic programming (SQP): both
- (4) ADMM (alternating direction method of multipliers, also applied to non-convex) and DR (Douglas-Rachford): convex
- (5) Primal Dual methods: mostly convex
- (6) Interior-point methods (IPM): convex. IPOPT: non-convex.

2.2.1 penalty methods

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, i = 1, \dots, m \\ & h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

Example 2.2.2

$$\begin{aligned} \min \quad & \|x\|_1 \\ \text{subject to} \quad & \|Ax - b\|_2^2 \leq \varepsilon^2 \end{aligned}$$

Then

$$\mathcal{L}(x, \lambda) = \|x\|_1 + \lambda(\|Ax - b\|_2^2 - \varepsilon^2)$$

If strong-duality holds and there exists saddle points,

$$x^* \in \operatorname{argmin}_x \mathcal{L}(x, \lambda^*)$$

We typically use the **quadratic penalty**:

For equality constraints, define

$$Q_\mu(x) = f_0(x) + \frac{\mu}{2} \sum_{i=1}^m h_i(x)^2$$

Solve

$$x^{(k)} = \operatorname{argmin} Q_{\mu_k}(x)$$

update μ_{k+1} increasing

$$x^{(k+1)} = \operatorname{argmin}_x Q(\mu_{k+1})(x)$$

This is a warm-start with $x^{(k)}$.

Theorem 2.2.3

Suppose $\mu_k \rightarrow \infty$. If $(x^{(k)})$ has a limit point x^* , then x^* is optimal.

Note. No convexity is needed but usually need convexity to update $x^{(k+1)}$.

For inequality constraints, define

$$Q_\mu(x) = f_0(x) + \frac{\mu}{2} \left(\sum_{i=1}^m h_i^2(x) + \sum_{i=1}^m [f_i(x)]_+^2 \right)$$

Note. The floor function makes it usually non-smooth.

Remark 2.2.4 The general idea is to put constraints into the objective:

$$\min f_0(x), \quad x \in C \Rightarrow \min f_0(x) + g(x)$$

Methods

- (1) $g(x) = I_C(x)$: this is mathematically equivalent but no computational benefit
- (2) penalty:

$$g_\mu(x) = \begin{cases} \mu \cdot x^2 & x < 0 \\ 0 & x \geq 0 \end{cases}$$

When $\mu \rightarrow \infty$, the smooth quadratic barrier converges the non-smooth infinite barrier.

(3) barrier:

$$g_\mu(x) = -\frac{1}{\mu} \cdot \log x$$

The barrier is not define for $x \leq 0$, so it forces the solution to stay strictly feasible.

Remark 2.2.5 Drawback: QP is often ill-conditioned as $\mu \rightarrow \infty$.

Example 2.2.6

$$\begin{aligned} \min \quad & \frac{1}{2}x^T Px \\ & Ax = b, A \in \mathbb{R}^{m \times n}, m < n \\ Q_\mu(x) = & \frac{1}{2}x^T Px + \frac{\mu}{2}\|Ax - b\|^2 \\ 0 = \nabla Q = & Px + \mu A^T(Ax - b) \\ x = (P + & \mu A^T A)^{-1} A^T b \end{aligned}$$

Since the condition number of the matrix usually depends on μ , as $\mu \rightarrow \infty$ the condition number also becomes very large.

In addition to quadratic penalty, we can use exact penalty: use $|h_i(x)|$ instead. It destroys smoothness.

2.2.2 Augmented Lagrangian

Assume we only have equality constraints.

$$\begin{aligned} (P) \quad & \min \quad f_0(x) \\ & h_i(x) = 0, i = 1, \dots, p \end{aligned}$$

If I knew optimal Lagrange multipliers, then

$$x^* \in \operatorname{argmin}_x \mathcal{L}(x, \nu^*) \Leftarrow f_0(x) + \sum_i \nu_i^* h_i(x)$$

and turn this into an unconstrained problem. We can try searching for ν^* :

$$g(\nu) = \min_x \mathcal{L}(x, \nu)$$

and we try to maximize $g(\nu)$ /dual ascent. We can use gradient ascent:

$$\nu_{k+1} = \nu_k + \eta_k \cdot \nabla g(\nu_k)$$

Proposition 2.2.7

If $g(\nu) = \inf_x \mathcal{L}(x, \nu)$ and $x_\nu = \operatorname{argmin} \mathcal{L}(x, \nu)$ is a single point under some reasonable conditions,

$$\nabla g(\nu) = \nabla_\nu \mathcal{L}(x_\nu, \nu)$$

However, this problem might not have a minimizer unless ν is optimal (can go to negative infinity *e.g.* linear objective). Then there is no gradient. Augmented Lagrangian aims to fix this. The problem becomes:

$$\begin{aligned} (P_\mu) \quad & \min_x f_0(x) + \frac{\mu}{2} \sum_{i=1}^m h_i^2(x) \\ & \text{subject to } h_i(x) = 0 \end{aligned}$$

(P_μ) is equivalent to (P) , since the squared terms cannot make the objective smaller. Then the dual ascent becomes

$$\begin{aligned} & \max_\nu g(\nu) \\ g(\nu) &= \min_x \mathcal{L}_\mu(x, \nu) = \min_x f_0(x) + \sum_i \nu_i h_i(x) + \frac{\mu}{2} \sum_i h_i^2(x) \end{aligned}$$

The quadratic penalty term regularizes this objective so it won't go to negative infinity. Then the algorithm becomes

$$\begin{aligned} \text{principle update: } x_k &\in \operatorname{argmin}_x \mathcal{L}_\mu(x, \nu_k) \\ \text{dual update: } \nu_{k+1} &= \nu_k + \mu h_i(x_k) \quad \text{gradient ascent} \end{aligned}$$

In practice we often provide μ heuristically.

Notice that this is adding a regularization term to the gradient ascent, or we can understand it as adding a linear term and thus not requiring μ to go to infinity as the penalty method, which fixes the ill-conditioned problem of penalty method. The convergence result is however fairly weak.

What if we add inequality constraints? We have some tricks and we only mention one here:

- (1) slack variables: $f_i(x) \leq 0 \Leftrightarrow f_i(x) + s_i = 0, s_i \geq 0$ and we keep the simpler inequality constraint implicit in the domain (hard constraints). Then we hope the solver can solve it.

See the rest in Nocedal and Wright. LANCELOT is a software that uses this.

SQP

Sequential quadratic programming, *i.e.* fancy Newton. Packages use this are SNOPT, KNITRO, LANCELOT, TRON.

Again let's first consider only equality constraints:

$$\begin{aligned} \min \quad & f_0(x) \\ & H(x) = 0 \end{aligned}$$

The Lagrangian is

$$\mathcal{L}(x, \nu) = f_0(x) + \nu^T H(x)$$

Let's list the KKT conditions:

- (1) stationarity:

$$0 = \nabla_x \mathcal{L} = \nabla f_0(x) + \mathbf{J}(x) \nu$$

where $\mathbf{J}(x)$ is the Jacobian of $H(x)$.

- (2) primal feasibility: $H(x) = 0$.

There are no complementary slackness or dual feasibility conditions since λ doesn't exist.

So we have two vector equations and we can just stack them together as one vector equation $F(x, \nu) = 0$ and treat it as a root-finding problem using Newton's method.

$$\begin{pmatrix} x_{k+1} \\ \nu_{k+1} \end{pmatrix} = \begin{pmatrix} x_k \\ \nu_k \end{pmatrix} + (F'(x_k, \nu_k))^{-1} F(x_k, \nu_k)$$

where

$$F'(x, \nu) = \begin{pmatrix} \nabla_{xx}^2 \mathcal{L} & \mathbf{J}(x) \\ \mathbf{J}(x)^T & 0 \end{pmatrix}$$

Let $p = x_{k+1} - x_k$, then the problem becomes

$$\begin{aligned} (QP) \quad & \min \quad f_k + \langle \nabla f_k, p \rangle + \frac{1}{2} \langle p | \nabla_{xx}^2 \mathcal{L} | p \rangle \\ & \text{subject to} \quad \mathbf{J}(x_k)p + H(x_k) = 0 \end{aligned}$$

This is a primal-dual Newton method. We can also add in inequality constraints and solve it via Quadratic Programming. We can add in tricks:

- (1) active- sets
- (2) line search or trust region

2.2.3 Newton's Method revisited, [BV04] Ch.9

$$\begin{aligned} \Delta x_{nt} &= \underset{\Delta x}{\operatorname{argmin}} f(x_k) + \langle \nabla f(x_k), \Delta x \rangle + \frac{1}{2} \langle \Delta x | \nabla^2 f(x_k) | \Delta x \rangle \\ &= \nabla^2 f(x_k)^{-1} \nabla f(x_k) \quad \nabla^2 f(x_k) \succ 0 \\ x_{k+1} &= x_k + t \Delta x_{nt}, t \approx 1 \end{aligned}$$

Observe: this is affine invariant. That is, let $\tilde{x} = Ax + b$, A invertible. And the update step doesn't change. This allows us to precondition the Hessian. Thus, convergence ought to be independent of the condition number of Hessian.

old-fashioned analysis

Assume $\mu I \preceq \nabla^2 f(x) \preceq MI \forall x$, sometimes $\|\nabla^3 f(x)\| \leq L$. These are strong assumptions, depend on potentially unknown parameters, and not affine-invariant.

self-concordant analysis

Introduced by Nesterov and Nemirovski.

Definition 2.2.8 (self-concordant) — $f : \mathbb{R} \rightarrow \mathbb{R}$ is **self-concordant** (sc) if f is convex and

$$|f'''(x)| \leq 2(f''(x))^{\frac{3}{2}}$$

Note. $f'''(0) \Rightarrow \text{sc}$.

Example 2.2.9

$f(x) = -\log(x)$. Let's check if f is sc: $f''(x) = \frac{1}{x^2}$, $f'''(-\frac{2}{x^3})$, and it works!

Definition 2.2.10 — $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is sc if $\forall x, v, \phi(t) = f(x + tv)$ is sc.

Note. This is just sc for all the lines.

Proposition 2.2.11

If $f(x)$ is sc, so is $x \mapsto f(Ax + b)$, A is $n \times n$. This is affine-invariant.

Example 2.2.12

linear or quadratic, $-\log(\mathbf{x}) := \sum -\log(x_i)$ is sc. So is $-\log \det(X)$.

Proposition 2.2.13

$\nabla^2 f(x) \succ 0$ and f is sc $\Leftrightarrow f$ is strictly convex and f is sc.

Remark 2.2.14 Recall that $\nabla^2 f(x) \succ 0$ implies strong (thus strict) convexity but strict convexity doesn't imply positive-definite Hessian even if the Hessian exists. It only implies positive-semidefinite Hessian. This proposition allows us to make a stronger claim.

damped/guarded Newton method

suitable for strictly convex functions.

Let $\Delta x_{nt} = -(\nabla^2 f_k)^{-1} \nabla f_k$. $\|z\|_H := \sqrt{\langle z | H | z \rangle}$, $H \succ 0$.

$$\begin{aligned} \lambda_k^2 &:= \lambda^2(x_k) := \|\Delta x_{nt}\|_{\nabla^2 f_k}^2 && \text{Newton decrement} \\ &:= \langle \nabla f_k | \nabla^2 f_k^{-1} | \nabla f_k \rangle \end{aligned}$$

Remark 2.2.15 By the proposition above, we ensure $\nabla^2 f_k \succ 0$ defining a valid norm.

terminate if $\lambda_k^2/2 < \text{tol}$

backtracking linesearch to ensure

$$\begin{aligned} f(x_k + t\Delta x_{nt}) &< f_k + \underbrace{\alpha}_{\in (0,0.5)} \underbrace{t}_{t_0=1} \underbrace{\langle \nabla f_k, \Delta x_{nt} \rangle}_{-\lambda_k^2} \\ x_{k+1} &= x_k + t\Delta x_{nt} \end{aligned}$$

Proposition 2.2.16 (BV Ch. 9.6.3)

Let $p^* = \min_x f(x)$. If $\lambda(x) < 0.68$, then $f(x) - p^* \leq \lambda^2(x)$.

Theorem 2.2.17

If f is sc, strictly onvex, etc, there exists $0 < \eta < \frac{1}{4}$, $\gamma > 0$ s.t.

- (i) Damped Newton phase: $\lambda(x_k) > \eta$, then $f(x_{k+1}) - f(x_k) < -\gamma$
- (ii) Quadratic convergence phase: $\lambda(x_k) \leq \eta$, then $t = 1$ and $2\lambda_{k+1} \leq (2\lambda_k)^2$. Here we have the bound:

$$f(x_{k_0+\ell}) - p^* \leq \frac{1}{4} \left(\frac{1}{2} \right)^{2^{\ell-k_0+1}}$$

Note. Before we get to the second phase, it's a constant rate and could be bad if it's too far away, but once we hit the second phase and we get amazing accuracy very quickly (less than 6 iterations).

The total number of iterations to ε -solution is

$$\frac{f(x_0) - p^*}{\gamma} + \log_2 \left(\log_2 \left(\frac{1}{\varepsilon} \right) \right) \approx 375(f(x_0) - p^*) + 6$$

Newton's method with equality constraints

This is easy!

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & Ax = b \end{aligned}$$

Then

$$\mathcal{L}(x, \nu) = f(x) + \nu^T (Ax - b)$$

The KKT conditions are:

- (1) $0 = \nabla f(x) + A^T \nu$.
- (2) $Ax = b$.

These are already linearized. So the update step with Taylor expansion becomes:

$$\begin{aligned} 0 &= \nabla f_k + \nabla^2 f_k \Delta x + A^T \nu \\ b &= A(x + \Delta x) \end{aligned}$$

$$\underbrace{\begin{pmatrix} \nabla^2 f_k & A^T \\ A & 0 \end{pmatrix}}_{\text{saddle pt system}} \begin{pmatrix} \Delta x \\ \nu \end{pmatrix} = \begin{pmatrix} -\nabla f_k \\ -Ax_k \end{pmatrix}$$

Another way to think is that $x = Fz + x_p$, where $Fz \in \ker A$. Then

$$\min_z f(Fz + x_p)$$

Affine-invariant makes this easy using Newton. Then this problem is equivalent to the saddle point system via Schur's complement.

Newton with inequality constraints: IPM

$$\begin{aligned} \min \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0, i = 1, \dots, m \\ & Ax = b \end{aligned}$$

Using log barrier, we change the inequality constraints into solving the sc function

$$x^*(t) = \underset{x, Ax=b}{\operatorname{argmin}} f_0(x) + \underbrace{\sum_{i=1}^m -\frac{1}{t} \log(-f_i(x))}_{\phi_t(x), \text{ sc}}$$

Central path: $\{x^*(t)\}_{t \geq 0}$. As we increase t , we get closer to the boundary (think about the log barrier plot). If we choose t to be too large, it will take a long time. Instead we solve for an increasing sequence of t using warm-start using solution from previous t .

The primal-dual method take one step of Newton at each t , and might be more efficient.

$f(x_k) - p^* \leq f(x_k) - g(\lambda_k, \nu_k) = \frac{m}{t}$. We can prove convergence using this nicely. See BV04 Ch 11 for more details on IPM.

Remark 2.2.18 IPM are state-of-the-art on problems (used by cvxpy) that are

- (1) medium size or smaller (maybe 10000)
- (2) conic problems: LP, QP, SOCP, SDP:

$$\begin{aligned} \min \quad & \langle C, X \rangle \\ \text{subject to } & X \succeq 0 \\ & \mathcal{A}(x) = b \end{aligned}$$

and we can use $-\log \det(X)$ to satisfy $X \succeq 0$.

(Block) Coordinate Descent, Alternating Minimization, Gauss-Seidel

This method exploits certain structure of the problem. It's also a "column-action" methods.

Example 2.2.19 (Gauss-Siedel)

Consider solving the least square problem with $x \in \mathbb{R}^n$ and let G be the

Gram matrix. The normal equation becomes

$$\begin{aligned}
 Gx &= \tilde{b} \\
 \begin{pmatrix} g_1 & \dots & g_n \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} &= \tilde{b} \\
 g_i \alpha &= \tilde{b} - \left(\sum_{j < i} g_j x_j^{(k+1)} + \sum_{j > i} g_j x_j^{(k)} \right) \\
 x_i^{(k+1)} &= \alpha
 \end{aligned}$$

Remark 2.2.20 Jacobi only uses $x^{(k)}$ for each k , allowing parallelization and randomized order.

If we do this row-wise, it's called ART (algebraic reconstruction technique) or Kaczmarz algorithm, or POCS (projection onto convex sets).

Consider

$$\begin{aligned}
 \min \quad & f(x), x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}, x_i \in C_i \text{ can be blocks} \\
 x_i^{(k+1)} &\in \operatorname{argmin}_{\alpha \in C} f\left(x_1^{(k+1)}, \dots, x_{j-1}^{(k+1)}, \alpha, x_{i+1}^{(k)}, \dots, x_n^{(k)}\right) \text{ or} \\
 x_i^{(k+1)} &= x_i^{(k)} - \eta \frac{\partial f}{\partial x_i} \left(x_1^{(k+1)}, \dots, x_{i-1}^{(k+1)}, x_i^{(k)}, \dots, x_n^{(k)}\right)
 \end{aligned}$$

The last step is that if it's too hard to find argmin, we instead just take a gradient at that step.

If we have two variables $\min f(x, y)$, then

$$\begin{aligned}
 x^{(k+1)} &\in \operatorname{argmin}_x f(x, y^{(k)}) \\
 y^{(k+1)} &\in \operatorname{argmin}_y f(x^{(k+1)}, y)
 \end{aligned}$$

We can modify it to PALM (proximal alternating linearized minimization) for

non-convex problems:

$$\begin{aligned}x^{(k+1)} &\in \operatorname{argmin}_x f(x, y^{(k)}) + \frac{\mu}{2} \|x - x^{(k)}\|^2 \\y^{(k+1)} &\in \operatorname{argmin}_y f(x^{(k+1)}, y) + \frac{\mu}{2} \|y - y^{(k)}\|^2\end{aligned}$$

ADMM (Alternating Direction Method of Multipliers)

See 2011 Boyd et al monograph.

$$\begin{aligned}\min \quad & f(x) \\ \text{subject to} \quad & Ax = b\end{aligned}$$

Attempt 1: Let's try with dual ascent, using y as the dual variable:

$$\begin{aligned}\mathcal{L}(x, y) &= f(x) + \langle y, Ax - b \rangle \\ g(y) &= \inf_x \mathcal{L}(x, y) \\ x_{k+1} &\in \operatorname{argmin}_x \mathcal{L}(x, y_k) \\ y_{k+1} &= y_k + t \underbrace{(Ax_{k+1} - b)}_{\nabla g(y_k)}\end{aligned}$$

This allows us to exploit the separable structure of the original problem if available *e.g.* $f(x) = \sum f_i(x_i)$, since we need to relax the linear constraint in the original problem, and the dual allows us to make the Lagrangian separable *i.e.* $\langle y, Ax - b \rangle \Rightarrow \langle A^*y, x \rangle - \langle y, b \rangle$. However, the downside is that it may not converge.

Attempt 2: Let's try the augmented Lagrangian which is equivalent to the original problem:

$$\begin{aligned}\min \quad & f(x) + \frac{\rho}{2} \|Ax - b\|^2 \\ & Ax = b\end{aligned}$$

Unfortunately the Lagrangian is no longer separable due to the quadratic term:

$$\mathcal{L}(x, y) = f(x) + \langle y, Ax - b \rangle + \frac{\rho}{2} \|Ax - b\|^2$$

Would it be possible to combine the two methods?

Attempt 3 (ADMM): let $F(x) = \sum_{i=1}^n f_i(x_i)$ or $F(v) = f(x) + g(z)$ if $n = 2$.

$$\begin{aligned} \min \quad & f(x) + g(z) \\ & Ax + Bz = c \end{aligned}$$

The algorithm is:

$$\begin{aligned} x^{(k+1)} &\in \operatorname{argmin}_x \mathcal{L}_\rho \left(\begin{pmatrix} x \\ z^{(k)} \end{pmatrix}, y^{(k)} \right) \\ z^{(k+1)} &\in \operatorname{argmin}_z \mathcal{L}_\rho \left(\begin{pmatrix} x^{(k+1)} \\ z \end{pmatrix}, y^{(k)} \right) \\ \text{update } y^{(k+1)} &= y_k + \rho(Ax_{k+1} + Bz_{k+1} - c) \end{aligned}$$

Note. If we jointly minimize the first two lines, it becomes the augmented Lagrangian method.

What if $n > 2$, i.e. $\min_x \sum_{i=1}^n f_i(x)$, where x is a block vector of x_i ?

One idea is $\min_{x_i} \sum f_i(x_i)$ s.t. linear constraints enforces $x_i = x_j$. Naive generalization from $n = 2$ doesn't converge very well. Instead we use a consensus trick:

$$F(v) = G(x) + H(z)$$

where $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$, z has the same size as x_i , and $v = \begin{pmatrix} x \\ z \end{pmatrix}$.

$$\begin{aligned} \min_{x,z} \quad & F(x) + G(z) \\ & \begin{pmatrix} I & & -I \\ & I & -I \\ & & \ddots & \vdots \\ & & & I & -I \end{pmatrix} \begin{pmatrix} x \\ z \end{pmatrix} = 0 \end{aligned}$$

This enforces $x_i = z \Rightarrow x_i = x_j$. We see that $A = I$ and $B = \begin{pmatrix} -I \\ \vdots \\ -I \end{pmatrix}$ from the

$n = 2$ linear constraint. Now we see x_i is decoupled at each update step:

$$x_{k+1} \in \operatorname{argmin}_x \mathcal{L}_\rho(x, z, y_k) \quad \text{decoupled}$$

$$z_{k+1} \in \operatorname{argmin}_z \mathcal{L}_\rho(x_{k+1}, z, y_k) = \frac{1}{n} \sum_{i=1}^n x_i \quad \text{consensus}$$

update y_{k+1} as usual

Remark 2.2.21 This is a common trick in optimization. In a coupled system, we relax it to be decoupled first and let them recouple later.

Douglas-Rachford

It is equivalent to ADMM in certain senses. See [BC17 28.3].

Algorithm: $0 < \lambda < 2, \rho > 0, y_0$

$$x_k = \operatorname{prox}_{\rho g}(y_k)$$

$$z_k = \operatorname{prox}_{\rho f}(2x_k - y_k)$$

$$y_{k+1} = y_k + \lambda(z_k - y_k)$$

The proximity operator is equivalent to the Lagrangian in ADMM.

Notice

$$\min \sum_{i=1}^n f_i(x) \Leftrightarrow \min \sum_{i=1}^n f_i(x_i) \text{ s.t. } x_i = x_j \ \forall i, j$$

Remark 2.2.22 In signal processing, we can parallelize this algorithm and only require communications among all workers in the consensus step. We can also enforce consensus in different ways and achieve consensus faster in a graph.

2.2.4 Primal Dual Methods

ADMM has some issues: if we want to $\min_x g(x) + \tilde{h}(Ax)$, $h(x) = \tilde{h}(Ax)$ (DR form). Rewrite as $\min_{x,z} g(x) + \tilde{h}(z)$, $Ax - z = 0$ (ADMM form). Finding prox_h is often hard due to A . $\operatorname{prox}_{\tilde{h}}$ easy doesn't mean prox_h is easy due to A .

The trick is to use ADMM with a scaled norm in the quadratic term in augmented Lagrangian. A clever choice is

$$\|z\|_M^2 = \langle z | M | z \rangle, \quad M = \frac{1}{\sigma} I - A^T A, \quad \sigma < \frac{1}{\|A\|^2} \Rightarrow M \succ 0$$

Chambolle and Pock, primal-dual hybrid gradient, preconditioned ADMM

general primal-dual method (Condat)

Suppose f, g, h convex, proper, lsc,

$$\min_x \underbrace{f(x)}_{\text{smooth}, \nabla f} + \underbrace{g(x)}_{\text{easy prox}_g} + \underbrace{h(Ax)}_{\text{easy prox}_h}$$

Lemma 2.2.23

$$x = \text{prox}_h(x) + \text{prox}_{h^*}(x).$$

Since h is convex,

$$h(w) = h^{**}(w) = \sup_y \langle w, y \rangle - h^*(y)$$

We solve

$$\min_x \max_y f(x) + g(x) + \underbrace{\langle Ax, y \rangle}_{\text{links primal-dual}} - h^*(y) \quad \text{saddle pt problem}$$

Optimality: use Fenchel-Rockafellar.

Assume CQ hold,

$$\begin{aligned} 0 &\in \partial(f + g + h \circ A)(x) \\ 0 &\in \nabla f(x) + \partial g(x) + A^T \underbrace{\partial h(Ax)}_y \quad \text{CQ} \\ &\begin{cases} 0 \in \nabla f(x) + \partial g(x) + A^T y \\ Ax \in \partial h^*(y) \quad \text{since } y \in \partial h(Ax) \end{cases} \end{aligned}$$

Rewrite the two equations in matrix form (although they are operators)

$$\underbrace{-\begin{pmatrix} \nabla f & 0 \\ 0 & 0 \end{pmatrix}}_{T_2} \begin{pmatrix} x \\ y \end{pmatrix} \in \underbrace{\begin{pmatrix} \partial g & A^T \\ -A & \partial h^* \end{pmatrix}}_{T_1} \begin{pmatrix} x \\ y \end{pmatrix}$$

This yields

$$\begin{aligned}
-T_2 z &\in T_1 z, & z &= \begin{pmatrix} x \\ y \end{pmatrix} \\
z - T_2 z &\in z + T_1 z & & \text{add } z \text{ on both sides} \\
(I - T_2)z &\in (I + T_1)z
\end{aligned}$$

We will solve via forward-backward (proximal descent). WLOG assume T_2 is 1-Lipschitz, $(I + dg)^{-1} = \text{prox}_g$ easy and $(I + \partial h^*)^{-1}$ easy, then

$$z_{k+1} = \underbrace{(I + T_1)^{-1}}_{\text{backward/implicit}} \underbrace{(I - T_2)}_{\text{forward/explicit}} z_k$$

Instead of adding z , we do a trick and add Vz :

$$Vz - T_2 z \in Vz + T_1 z$$

where we choose V to be

$$V = \begin{pmatrix} \tau^{-1}I & -A^T \\ -A & \sigma^{-1}I \end{pmatrix}$$

This guarantees that $V \succ 0$ if $\sigma\tau > \|A\|^{-2}$.

$$\begin{aligned}
z_{k+1} &= (V + T_1)^{-1}(V - T_2)z \\
V + T_1 &= \begin{pmatrix} \tau^{-1}I + \partial g & 0 \\ -2A & \sigma^{-1}I + \partial h^* \end{pmatrix} & \text{upper triangular}
\end{aligned}$$

We can invert this using back substitution:

$$\begin{pmatrix} \tau^{-1}I + \partial g & 0 \\ -2A & \sigma^{-1}I + \partial h^* \end{pmatrix} \begin{pmatrix} x_{k+1} \\ y_{k+1} \end{pmatrix} = \begin{pmatrix} v \\ w \end{pmatrix}$$

$$x_{k+1} = (\tau^{-1}I + \partial g)^{-1}v \quad \text{via } \text{prox}_g$$

then solve for y_{k+1}

2.3 Linear Programs

2.3.1 Simplex Method

Definition 2.3.1 (face) —

$$\{x : a_i^T x = b_i, i \in I; a_i^T x \leq b_i, i \in I^c\}$$

Definition 2.3.2 (vertex) — A **vertex** of a polyhedral set $\{x : Ax \leq b\}$ is a face that has just 1 point.

There exists an optimal solution that is a vertex (optimal might not be unique *e.g.* when level set is parallel to an edge).

Simplex method hops from one vertex to another. We also call vertex a basic feasible point.

The constraints is $Ax = b, x \geq 0$, A is $m \times n$ matrix, and for some points the equality is achieved. Let B be the list of all indices, called basis, where $|B| = m$. And for $n - m$

We start at a basis B , use duality to choose 1 index to leave. Adjust until a new index enters.

Problems:

- (1) lots of linear algebra, A_B^{-1} needs to be updated/downdated. Usually we use LU. If A is sparse, it's more complicated.
- (2) there are many pivot rules
- (3) finding a starting point: 2-phase approach
- (4) degenerate bases and cycling
- (5) presolving
- (6) variants: dual simplex, self-dual

complexity

In CS, complexity means: time in input. What does the input mean?

- (1) real arithmetic, numbers are “cts”, encode real numbers. A unit storage cost for a \mathbb{R} number. Operations/flops have unit cost. Thus the cost depends only on the dimension.

- (2) rational model, or combinatorial model: $b_i \in \mathbb{Q}$. Now the cost of operations depends on the value of the integers. Larger value is more costly.

Practically simplex is great. However, Klee-Minty showed that in \mathbb{R}^n with 2^n vertices and simplex visited all. Thus simplex is not in P using the usual pivot rule. The open question is whether this is true for all pivot rules. If we prove polynomial Hirsch conjecture we will know the answer.

Example 2.3.3 (solving linear equations)

In real arithmetic, this is polynomial $\mathcal{O}(n^3)$. In rational model, it is also polynomial.

For LPs, in rational model, the answer is again yes. Khachiyan proposed ellipsoid method, a generalization of bisection method. Then Karmarkov with IPM which is practical.

What is the answer for real arithmetic model, is it polynomial time (strongly polynomial)? Are there other algorithms to solve LPs in polynomial time?

2.4 Bonus: find gradients

- (1) don't: DFO (derivate-free optimization).
- (2) finite differences
- (3) analytic
- (4) automatic:
 - (a) automatic-differentiation (AD)
 - (b) adjoint-state method

2.4.1 DFO

In small dimensions, the gradient doesn't much information than multiple function evaluations. In large dimensions DFO fails.

Let $x \in \mathbb{R}^n$. DFO gets $\{f(x_i)\}_{i=1}^k$ and gradient method gets $\{f(x_i), \nabla f(x_i)\}_{i=1}^k$.

Example 2.4.1

$$\min f(x) = \frac{1}{2}x^T Hx + q^T x$$

Consider the analogy of a game of guessing numbers but with an adversarial kid judging if your guesses are right or wrong. DFO only has k -degrees of freedom, but the function has n^2 -degree of freedom. So we need $k = n^2$ to get it right. Although in low-dimensions we can use a grid-search method (just like the number guessing game).

The usual setup of DFO: ∇f exists but hard to find or ∇f doesn't exist. This often happens when f is very noisy.

The algorithms are:

- (1) model-based: *e.g.* DFO-TR (trust-region) with $\dim \approx 200$. We interpolate a polynomial in a trust region iteratively. We usually do linear or quadratic polynomials in high-dimensions because we need too many degrees of freedom in cubics or more.
- (2) coordinate-descent, “pattern search”.
- (3) Nelder-Mead simplex-reflection (unrelated to symplex method)
- (4) implicit filtering: gradient descent with large-stepsizes finite-diff.
- (5) Bayesian optimization: use Gaussian processes.
- (6) Evolutionary search: pick a random direction.
- (7) particles swarm: meta-heuristics. They are good for finding global minimum but have poor accuracy.

2.4.2 Finite differences

Example 2.4.2

$$f(x) = \cos x$$

$$f'(x) = -\sin x$$

$$= \lim_{h \rightarrow 0} \frac{\cos(x+h) - \cos(x)}{h} \quad \text{forward-diff } \mathcal{O}(h)$$

$$= \lim_{h \rightarrow 0} \frac{\cos(x+h) - \cos(x-h)}{2h} \quad \text{central-diff } \mathcal{O}(h^2)$$

In \mathbb{R}^n ,

$$(\nabla f(x))_i \approx \frac{f(x + he_i) - f(x)}{h} \quad n+1 \text{ function eval}$$

$$\approx \frac{f(x + he_i) - f(x - he_i)}{2h} \quad 2n \text{ function eval}$$

This becomes expensive for more number of points at $\mathcal{O}(n)$.

Although often the error isn't a big deal, this is a bad idea if f is noisy. This is great for prototyping.

Is the expense necessary?

Example 2.4.3

$$f(x) = \frac{1}{2} \|Ax - b\|^2, A \text{ square} \quad \mathcal{O}(n^2)$$

$$\nabla f(x) = A^T(Ax - b) \quad \mathcal{O}(n^2)$$

But finite-difference here is $\mathcal{O}(n^3)$. So it's possible to beat finite-difference.

2.4.3 analytic solutions

Chain rule for vector fields:

Let $h = g \circ f$, where $f : \mathbb{R}^n \rightarrow \mathbb{R}^m, g : \mathbb{R}^m \rightarrow \mathbb{R}^p$, so $h : \mathbb{R}^n \rightarrow \mathbb{R}^p$.

$$[Jf(x)]_{ij} = \frac{[\partial f(x)]_{ij}}{\partial x_j}$$

$$Jh(x) = Jg(f(x))Jf(x)$$

Example 2.4.4

If $p = 1$, then

$$\nabla h(x) = (Jh(x))^T = (Jf(x))^T \nabla g(f(x))$$

Example 2.4.5

$$\begin{aligned} f(x) &= Ax + b, p = 1 & Jf(x) &= A \\ \nabla h(x) &= A^T \nabla g(Ax + b) \end{aligned}$$

2.4.4 Automatic differentiation

See hand-written notes. I'm too lazy to write up this one.

2.4.5 by hand**implicit differentiation**

$F(x, y) = 0$ often implicitly defines a function $y = f(x)$. Find dy/dx .

$F = 0 \Rightarrow dF/dx = d0/dx = 0$. Thus,

$$\begin{aligned} 0 &= \frac{dF}{dx} = \frac{\partial F}{\partial x} \frac{dx}{dx} + \frac{\partial F}{\partial y} \frac{dy}{dx} \\ 0 &= \frac{\partial F}{\partial x} + \frac{\partial F}{\partial y} \frac{dy}{dx} \\ \frac{dy}{dx} &= - \frac{\frac{\partial F}{\partial x}}{\frac{\partial F}{\partial y}} \end{aligned}$$

matrix variables

See BV04, or A Matrix Handbook for Statisticians Seber 08, or Matrix Cookbook, for examples.

Example 2.4.6

$$\nabla(\log \det(X)) = X^{-1}, X \succ 0$$

parametric functions

See notes for details.

$$f(x) = \max_{z \in Z} \phi(x, z)$$

Theorem 2.4.7 (Danskin)

Suppose Z compact, ϕ jointly continuous and $\phi(\cdot, z)$ convex. Define

$$Z(x) = \operatorname{argmax}_{z \in Z} \phi(x, z).$$

Then

- (1) The directional derivative D_d satisfies

$$D_d f(x) = \max_{z \in Z(x)} D_d \phi(x, z)$$

- (2) If $\phi(\cdot, z)$ is differentiable, $\nabla_x \phi$ is continuous, then

$$\partial f(x) = \operatorname{conv}\{\nabla_x \phi(x, z) : z \in Z(x)\}$$

and $Z(x)$ a singleton $\Rightarrow f$ is differentiable.

Note. This theorem doesn't apply to the discrete case.

Theorem 2.4.8 (Dubovitskii and Milyutin)

If $|Z|$ is finite, $\phi(\cdot, z)$ is convex $\forall z \in Z$, then

$$\partial f = \operatorname{conv} \left\{ \bigcup_{z \in Z(x)} \partial \phi(x, z) \right\}$$

Example 2.4.9

$f(x) = |x| = \max\{x, -x\}$. Then $\partial f(0) = \text{conv}\{1, -1\} = [-1, 1]$.

$$f(x) = \min_{z \in Z} \phi(x, z)$$

Theorem 2.4.10

Under the same conditions,

$$\partial f(x) = \partial \phi(x, z) \quad \forall z \in Z(x)$$

$$f(x) = \int \phi(x, z) dz$$

Theorem 2.4.11

Sometimes,

$$f'(x) = \int \frac{d}{dx} \phi(x, z) dz$$

2.4.6 Adjoint state method

(1) implicit differentiation and careful parentheses.

$$\min_p g(u(p), p) \text{ s.t. } f(u(p), p) = 0, u \in \mathcal{H}$$

Example 2.4.12

$u_{tt} = c^2(x)u_{xx}$. Let $p = c^2(x)$, e.g. varying speed of sound. Or p could be other parameters like ICs and BCs. Applications in inverses problems such as oil detection. This is often called “PDE-constrained

optimization problem". To solve this, we ignore the constraint first. Then we wish to solve dg/dp .

Example 2.4.13

$$\min g(u, p) \text{ s.t. } Au = b, A = V \operatorname{diag}(p) V^T, p \in \mathbb{R}^n, u \in \mathbb{R}^m, A \in \mathbb{R}^m \times \mathbb{R}^m$$

$$g(u, p) = \frac{1}{2} \sum_i (u_i - y_i)^2 + \frac{1}{2} \|p\|^2$$

Example 2.4.14

Let $\mathcal{H} = \mathbb{R}^m$. $f(u, p) = A(p) \cdot u - b(p) = 0$, linear in u , so $u(p) = A(p)^{-1}b(p)$. The goal is to find

$$\begin{aligned} (\nabla_p g)^T &= g_p + g_u u_p \\ 0 = f_p &= A_p u + A u_p - b_p \\ u_p &= A^{-1}(b_p - A_p u) \\ (\nabla_p g)^T &= g_p + g_u (A^{-1}(b_p - A_p u)) \\ &= g_p + (g_u A^{-1})(b_p - A_p u) \end{aligned}$$

Let $\lambda^* = g_u A^{-1} \Rightarrow \lambda = A^{-*} g_u^*$, then we have the adjoint-state equation

$$A^* \lambda = g_u^*$$

Now by clever grouping we are only solving one RHS instead of n RHS.

- (2) adjoints of (bounded/unbounded) linear operators. What about adjoints if $|\mathcal{H}| = \infty$?

Example 2.4.15

Let $L : \mathcal{H} \rightarrow \mathcal{H}$, $\mathcal{H} = L^2[0, T]$:

$$L(u) = 3u' + 4u, u(0) = 0, t \in [0, T].$$

Let $f(u, p) = L(u) - h(t)$. To get the formal adjoint (adjoint doesn't exist since L doesn't have full domain),

$$\begin{aligned} \int_0^T (3u' + 4u)v dt &= 3 \int_0^T u' v dt + 4 \int_0^T u v dt \\ \int_0^T u(-3v' + 4v) dt &= 3uv|_0^T - 3 \int_0^T uv' dt + 4 \int_0^T uv dt \quad \text{set } v(T) = 0 \end{aligned}$$

Thus,

$$L^*(v) = -3v' + 4v, v(T) = 0, L^*(v) = h$$

Remark 2.4.16

- (1) doesn't require linear PDE's
- (2) not always a good idea: memory issues, consistency from the order of applying discretization or optimization
- (3) software: Dolfin-Adjoint, FEnICS