

Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 3: Convex analysis and complexity

Laboratory for Information and Inference Systems (LIONS)
École Polytechnique Fédérale de Lausanne (EPFL)

EE-556 (Fall 2016)



License Information for Mathematics of Data Slides

- ▶ This work is released under a [Creative Commons License](#) with the following terms:
- ▶ **Attribution**
 - ▶ The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.
- ▶ **Non-Commercial**
 - ▶ The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor's permission.
- ▶ **Share Alike**
 - ▶ The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor's work.
- ▶ [Full Text of the License](#)

Outline

- ▶ This lecture
 - 1. Learning as an optimization problem
 - 2. Basic concepts in convex analysis
 - 3. Complexity theory review
 - ▶ Asymptotic notation
 - ▶ Computational complexity
 - ▶ Hardness result: certifying optimality in non-convex problems
- ▶ Next lecture
 - 1. Unconstrained convex optimization: the basics
 - 2. Gradient descent methods

Recommended reading

- ▶ Chapter 2 & 3 in S. Boyd, and L. Vandenberghe, *Convex Optimization*, Cambridge Univ. Press, 2009.
- ▶ Appendices A & B in D. Bertsekas, *Nonlinear Programming*, Athena Scientific, 1999.
- ▶ Chapter 3 & 34 in Cormen, Thomas H., et al. *Introduction to algorithms*. Vol. 2. Cambridge: MIT press, 2001.
- ▶ Sections 3.1, 3.2, 5.3, 6.3, 7.2-7.5 in Sipser, Michael. *Introduction to the Theory of Computation*. Cengage Learning, 2012.

Motivation

Motivation

- ▶ The first part of this recitation introduces basic notions in **convex analysis**.
- ▶ The second part is intended to help you understand some concepts in the **theory of computation** that you will encounter in discussions concerned with efficient computation, and some of the **notation** involved.

Practical Issues

Recall from Lecture 2:

Given an estimator $\hat{\mathbf{x}} \in \arg \min_{\mathbf{x} \in \mathcal{X}} \{F(\mathbf{x})\}$ of \mathbf{x}^\natural , we discussed two key questions:

1. Is the formulation **reasonable**?
2. What is the role of the **data size**?

Consider the estimation error in the ℓ_2 -norm: $\|\hat{\mathbf{x}} - \mathbf{x}^\natural\|_2$.

- ▶ Is $\|\hat{\mathbf{x}} - \mathbf{x}^\natural\|_2$ enough to evaluate the performance of the estimator $\hat{\mathbf{x}}$?

Practical Issues

No, because in general we can only *numerically approximate* the solution of

$$\hat{\mathbf{x}} \in \arg \min_{\mathbf{x} \in \mathbb{R}^p} \{F(\mathbf{x})\}.$$

Implementation

How do we *numerically approximate* $\hat{\mathbf{x}}$?

Practical performance

Denote the numerical approximation by \mathbf{x}_ϵ^* . The practical performance is governed by

$$\|\mathbf{x}_\epsilon^* - \mathbf{x}^\natural\|_2 \leq \underbrace{\|\mathbf{x}_\epsilon^* - \hat{\mathbf{x}}\|_2}_{\text{approximation error}} + \underbrace{\|\hat{\mathbf{x}} - \mathbf{x}^\natural\|_2}_{\text{statistical error}}.$$

How do we evaluate $\|\mathbf{x}_\epsilon^* - \hat{\mathbf{x}}\|_2$?

- ▶ The ϵ -approximation solution \mathbf{x}_ϵ^* will be defined rigorously in the later lectures.

Practical issues

How do we *numerically approximate* $\hat{\mathbf{x}} \in \arg \min_{\mathbf{x} \in \mathbb{R}^p} \{F(\mathbf{x})\}$ for a given F ?

General idea of an optimization algorithm

Guess a solution, and then *refine* it based on *oracle information*.

Repeat the procedure until the result is *good enough*.

How do we evaluate the approximation error $\|\mathbf{x}_\epsilon^* - \hat{\mathbf{x}}\|_2$?

General concept about the approximation error

It depends on the *characteristics* of the function F and the chosen numerical *optimization algorithm*.

Need for convex analysis

General idea of an optimization algorithm

Guess a solution, and then *refine* it based on *oracle information*.

Repeat the procedure until the result is *good enough*.

General concept about the approximation error

It depends on the *characteristics* of the function F and the chosen numerical *optimization algorithm*.

Role of convexity

Convexity provides a key optimization framework in obtaining numerical approximations at theoretically well-understood computational costs.

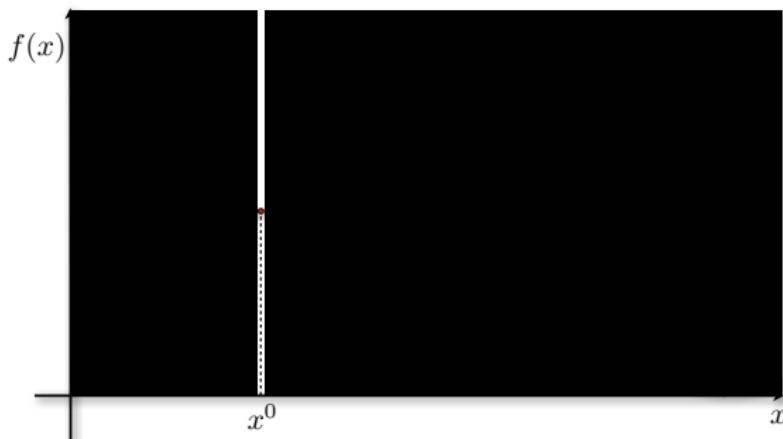
To precisely understand these ideas, we need to understand basics of *convex analysis*.

Challenges for an iterative optimization algorithm

Problem

Find the minimum x^* of $f(x)$, given starting point x^0 based on only local information.

- ▶ Fog of war

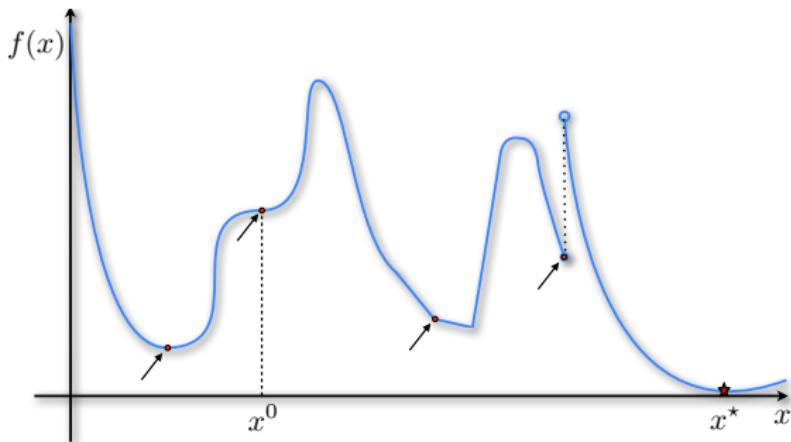


Challenges for an iterative optimization algorithm

Problem

Find the minimum x^* of $f(x)$, given starting point x^0 based on only local information.

- ▶ Fog of war, non-differentiability, discontinuities, local minima, stationary points...

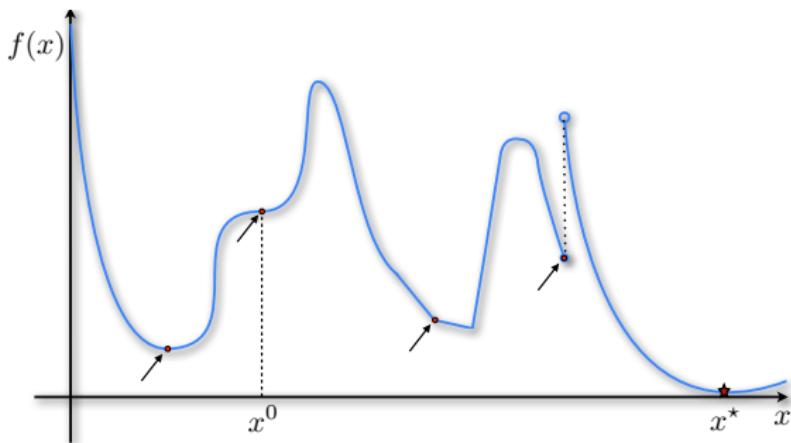


Challenges for an iterative optimization algorithm

Problem

Find the minimum x^* of $f(x)$, given starting point x^0 based on only local information.

- ▶ Fog of war, non-differentiability, discontinuities, local minima, stationary points...



We need a key structure on the function local minima: **Convexity**.

Basics of functions

Definition (Function)

A function f with domain $\mathcal{Q} \subseteq \mathbb{R}^p$ and codomain $\mathcal{U} \subseteq \mathbb{R}$ is denoted as:

$$f : \mathcal{Q} \rightarrow \mathcal{U}.$$

The domain \mathcal{Q} represents the set of values in \mathbb{R}^p on which f is defined and is denoted as $\text{dom}(f) \equiv \mathcal{Q} = \{\mathbf{x} : -\infty < f(\mathbf{x}) < +\infty\}$. The codomain \mathcal{U} is the set of function values of f for any input in \mathcal{Q} .

Continuity in functions

Definition (Continuity)

Let $f : \mathcal{Q} \rightarrow \mathbb{R}$ where $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, f is a continuous function over its domain \mathcal{Q} if and only if

$$\lim_{\mathbf{x} \rightarrow \mathbf{y}} f(\mathbf{x}) = f(\mathbf{y}), \quad \forall \mathbf{y} \in \mathcal{Q},$$

i.e., the limit of f —as \mathbf{x} approaches \mathbf{y} —exists and is equal to $f(\mathbf{y})$.

Definition (Class of continuous functions)

We denote the class of continuous functions f over the domain \mathcal{Q} as $f \in \mathcal{C}(\mathcal{Q})$.

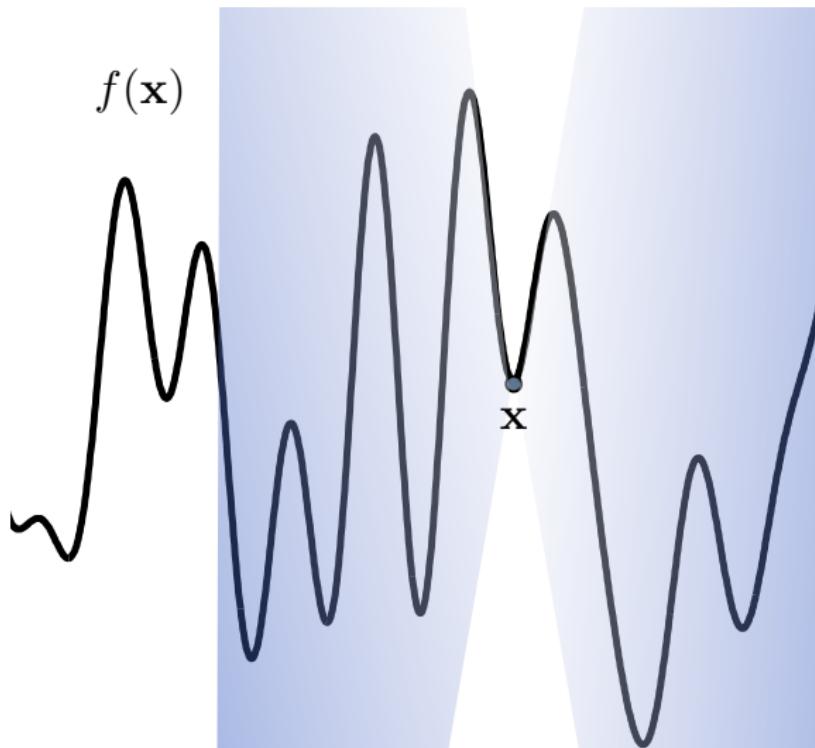
Definition (Lipschitz continuity)

Let $f : \mathcal{Q} \rightarrow \mathbb{R}$ where $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, f is called Lipschitz continuous if there exists a constant value $K \geq 0$ such that:

$$|f(\mathbf{y}) - f(\mathbf{x})| \leq K \|\mathbf{y} - \mathbf{x}\|_2, \quad \forall \mathbf{x}, \mathbf{y} \in \mathcal{Q}.$$

- ▶ "Small" changes in the input result into "small" changes in the function values.

Continuity in functions



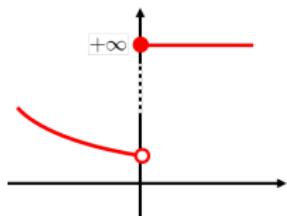
Lower semi-continuity

Definition

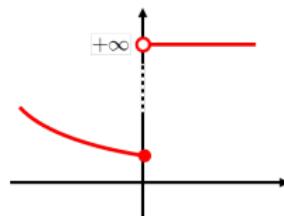
A function $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is lower semi-continuous (l.s.c.) if

$$\liminf_{x \rightarrow y} f(x) \geq f(y), \text{ for any } y \in \text{dom}(f).$$

$$f(x) = \begin{cases} e^{-x}, & \text{if } x < 0 \\ +\infty, & \text{if } x \geq 0 \end{cases}$$



$$f(x) = \begin{cases} e^{-x}, & \text{if } x \leq 0 \\ +\infty, & \text{if } x > 0 \end{cases}$$



Unless stated otherwise, we only consider l.s.c. functions.

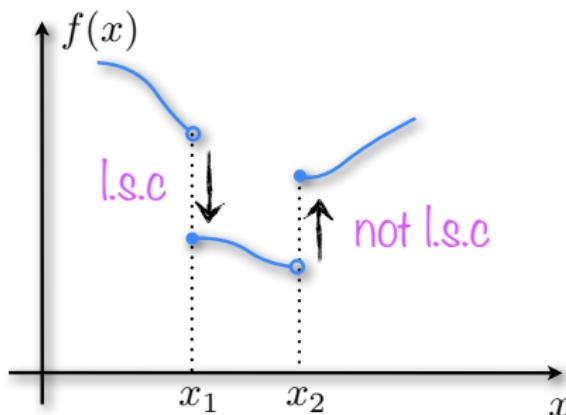
Lower semi-continuity

Definition

A function $f : \mathbb{R}^n \rightarrow \mathbb{R} \cup \{+\infty\}$ is lower semi-continuous (l.s.c.) if

$$\liminf_{\mathbf{x} \rightarrow \mathbf{y}} f(\mathbf{x}) \geq f(\mathbf{y}), \text{ for any } \mathbf{y} \in \text{dom}(f).$$

- ▶ **Intuition:** A lower semi-continuous function *only jumps down*.



Differentiability in functions

- We use $\nabla f(\mathbf{x})$ to denote the *gradient* of f at $\mathbf{x} \in \mathbb{R}^p$ such that:

$$\nabla f(\mathbf{x}) = \sum_{i=1}^p \frac{\partial f}{\partial x_i} \mathbf{e}_i = \left[\frac{\partial f}{\partial x_1}, \dots, \frac{\partial f}{\partial x_p} \right]^T$$

Example: $f(\mathbf{x}) = \|\mathbf{b} - \mathbf{Ax}\|_2^2$

$$\nabla f(\mathbf{x}) = -2\mathbf{A}^T (\mathbf{b} - \mathbf{Ax}).$$

Definition (Differentiability)

Let $f \in \mathcal{C}(\mathcal{Q})$ where $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, f is a k -times continuously differentiable on \mathcal{Q} if its partial derivatives up to k -th order exist and are continuous $\forall \mathbf{x} \in \mathcal{Q}$.

Definition (Class of differentiable functions)

We denote the class of k -times continuously differentiable functions f on \mathcal{Q} as $f \in \mathcal{C}^k(\mathcal{Q})$.

- In the special case of $k = 2$, we dub $\nabla^2 f(\mathbf{x})$ the **Hessian** of $f(\mathbf{x})$, where $[\nabla^2 f(\mathbf{x})]_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$.
- We have $\mathcal{C}^q(\mathcal{Q}) \subseteq \mathcal{C}^k(\mathcal{Q})$ where $q \leq k$. For example, a twice differentiable function is also once differentiable.
- For the case of complex-valued matrices, we refer to the Matrix Cookbook online.

Differentiability in functions

- ▶ Some examples:

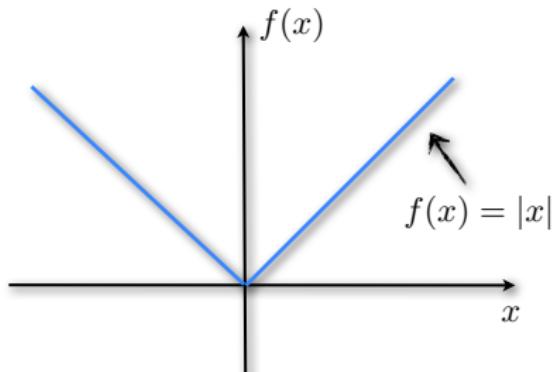
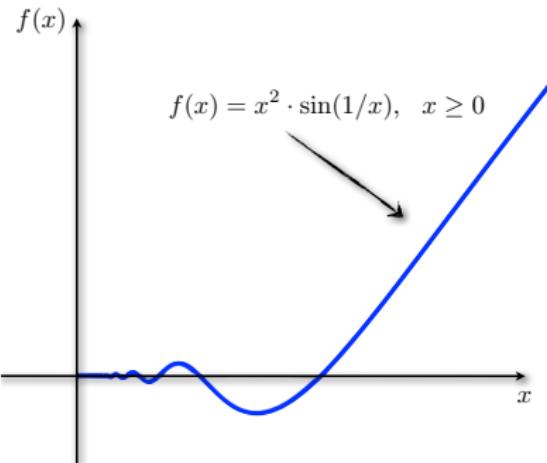


Figure: (Left panel) ∞ -times continuously differentiable function in \mathbb{R} . (Right panel) Non-differentiable $f(x) = |x|$ in \mathbb{R} .

Stationary points of differentiable functions

Definition (Stationary point)

A point \bar{x} is called a stationary point of a twice differentiable function $f(x)$ if

$$\nabla f(\bar{x}) = \mathbf{0}.$$

Definition (Local minima, maxima, and saddle points)

Let \bar{x} be a stationary point of a twice differentiable function $f(x)$.

- ▶ If $\nabla^2 f(\bar{x}) \succ 0$, then the point \bar{x} is called a local minimum.
- ▶ If $\nabla^2 f(\bar{x}) \prec 0$, then the point \bar{x} is called a local maximum.
- ▶ If $\nabla^2 f(\bar{x}) = 0$, then the point \bar{x} can be a saddle point depending on the sign change.

Stationary points of smooth functions contd.

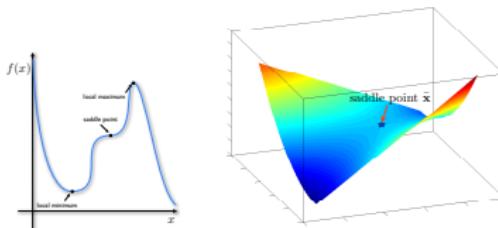
Intuition

Recall Taylor's theorem for the function f around $\bar{\mathbf{x}}$ for all \mathbf{y} that satisfy $\|\mathbf{y} - \bar{\mathbf{x}}\|_2 \leq r$ in a local region with radius r as follows

$$f(\mathbf{y}) = f(\bar{\mathbf{x}}) + \langle \nabla f(\bar{\mathbf{x}}), \mathbf{y} - \bar{\mathbf{x}} \rangle + \frac{1}{2} (\mathbf{y} - \bar{\mathbf{x}})^T \nabla^2 f(\mathbf{z}) (\mathbf{y} - \bar{\mathbf{x}}),$$

where \mathbf{z} is a point between $\bar{\mathbf{x}}$ and \mathbf{y} . When $r \rightarrow 0$, the second-order term becomes $\nabla^2 f(\mathbf{z}) \rightarrow \nabla^2 f(\bar{\mathbf{x}})$. Since $\nabla f(\bar{\mathbf{x}}) = 0$, Taylor's theorem leads to

- ▶ $f(\mathbf{y}) > f(\bar{\mathbf{x}})$ when $\nabla^2 f(\bar{\mathbf{x}}) \succ 0$. Hence, the point $\bar{\mathbf{x}}$ is a local minimum.
- ▶ $f(\mathbf{y}) < f(\bar{\mathbf{x}})$ when $\nabla^2 f(\bar{\mathbf{x}}) \prec 0$. Hence, the point $\bar{\mathbf{x}}$ is a local maximum.
- ▶ $f(\mathbf{y}) \geq f(\bar{\mathbf{x}})$ when $\nabla^2 f(\bar{\mathbf{x}}) = 0$. Hence, the point $\bar{\mathbf{x}}$ can be a saddle point (i.e., $f(x) = x^3$ at $\bar{x} = 0$), a local minima (i.e., $f(x) = x^4$ at $\bar{x} = 0$) or a local maxima (i.e., $f(x) = -x^4$ at $\bar{x} = 0$).



Convexity

Definition

A function $f : \mathcal{Q} \rightarrow \mathbb{R} \cup \{+\infty\}$ is called convex on its domain \mathcal{Q} if, for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q}$ and $\alpha \in [0, 1]$, we have:

$$f(\alpha \mathbf{x}_1 + (1 - \alpha) \mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1 - \alpha) f(\mathbf{x}_2).$$

- If $-f(\mathbf{x})$ is convex, then $f(\mathbf{x})$ is called concave.

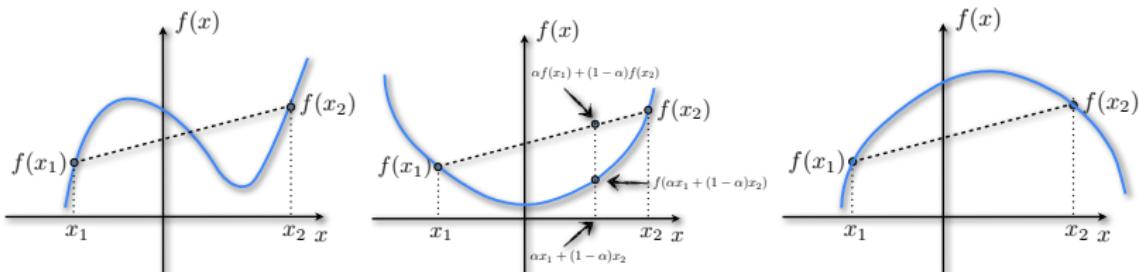


Figure: (Left) Non-convex (Middle) Convex (Right) Concave

Convexity

Definition

A function $f : \mathcal{Q} \rightarrow \mathbb{R} \cup \{+\infty\}$ is called convex on its domain \mathcal{Q} if, for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q}$ and $\alpha \in [0, 1]$, we have:

$$f(\alpha \mathbf{x}_1 + (1 - \alpha) \mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1 - \alpha) f(\mathbf{x}_2).$$

- ▶ Additional terms that you will encounter in the literature

Definition (Proper)

A convex function f is called proper if its domain satisfies $\text{dom}(f) \neq \emptyset$ and, $f(\mathbf{x}) > -\infty, \forall \mathbf{x} \in \text{dom}(f)$.

Definition (Extended real-valued convex functions)

We define the extended real-valued convex functions f as

$$f(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \text{if } \mathbf{x} \in \text{dom}(f) \\ +\infty & \text{if otherwise} \end{cases}$$

To denote this concept, we use $f : \text{dom}(f) \rightarrow \mathbb{R} \cup \{+\infty\}$. (Note how l.s.c. might be useful)

Convexity

Definition

A function $f : \mathcal{Q} \rightarrow \mathbb{R} \cup \{+\infty\}$ is called convex on its domain \mathcal{Q} if, for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q}$ and $\alpha \in [0, 1]$, we have:

$$f(\alpha \mathbf{x}_1 + (1 - \alpha) \mathbf{x}_2) \leq \alpha f(\mathbf{x}_1) + (1 - \alpha) f(\mathbf{x}_2).$$

Example

Function	Example	Attributes
ℓ_p vector norms, $p \geq 1$	$\ \mathbf{x}\ _2, \ \mathbf{x}\ _1, \ \mathbf{x}\ _\infty$	convex
ℓ_p matrix norms, $p \geq 1$	$\ \mathbf{X}\ _* = \sum_{i=1}^{\text{rank}(\mathbf{X})} \sigma_i$	convex
Square root function	\sqrt{x}	concave, nondecreasing
Maximum of functions	$\max\{x_1, \dots, x_n\}$	convex, nondecreasing
Minimum of functions	$\min\{x_1, \dots, x_n\}$	concave, nondecreasing
Sum of convex functions	$\sum_{i=1}^n f_i, f_i$ convex	convex
Logarithmic functions	$\log(\det(\mathbf{X}))$	concave, assumes $\mathbf{X} \succ 0$
Affine/linear functions	$\sum_{i=1}^n X_{ii}$	both convex and concave
Eigenvalue functions	$\lambda_{\max}(\mathbf{X})$	convex, assumes $\mathbf{X} = \mathbf{X}^T$

Strict convexity

Definition

A function $f : \mathcal{Q} \rightarrow \mathbb{R} \cup \{+\infty\}$ is called *strictly convex* on its domain \mathcal{Q} if and only if for any $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q}$ and $\alpha \in [0, 1]$ we have:

$$f(\alpha \mathbf{x}_1 + (1 - \alpha) \mathbf{x}_2) < \alpha f(\mathbf{x}_1) + (1 - \alpha) f(\mathbf{x}_2).$$

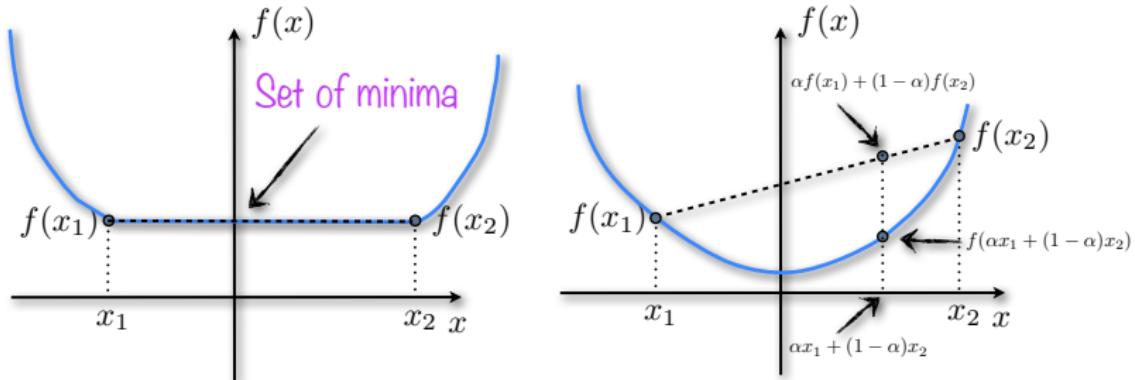


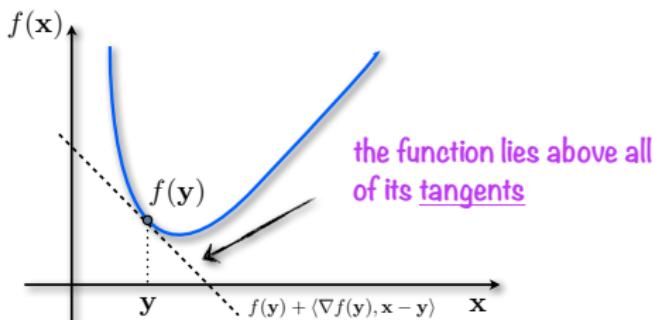
Figure: (Left panel) Convex function. (Right panel) Strictly convex function.

Revisiting: Alternative definitions of function convexity II

Definition

A function $f \in \mathcal{C}^1(\mathcal{Q})$ is called convex on its domain if for any $x, y \in \mathcal{Q}$:

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



Definition

A function $f \in \mathcal{C}^1(\mathcal{Q})$ is called convex on its domain if for any $x, y \in \mathcal{Q}$:

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

* That is, if its gradient is a monotone operator.

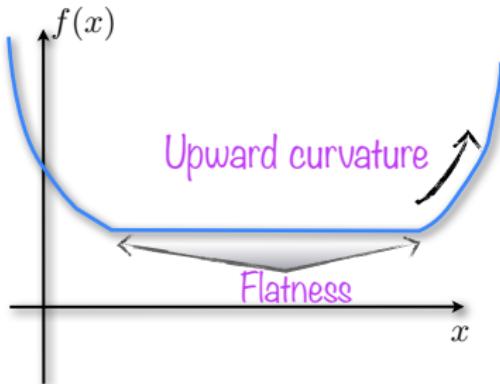
Revisiting: Alternative definitions of function convexity III

Definition

A function $f \in \mathcal{C}^2(\mathbb{R}^p)$ is called convex on its domain if for any $\mathbf{x}, \mathbf{y} \in \mathbb{R}^p$:

$$\nabla^2 f(\mathbf{x}) \succeq 0.$$

- ▶ Geometrical interpretation: the graph of f has zero or positive (upward) curvature.
- ▶ However, this does not exclude flatness of f .
- ▶ $\nabla^2 f(\mathbf{x}) \succ 0$ is a sufficient condition for *strict* convexity.



Stationary points and convexity

Lemma

Let f be a **smooth convex** function, i.e., $f \in \mathcal{F}^1$. Then, any stationary point of f is also a global minimum.

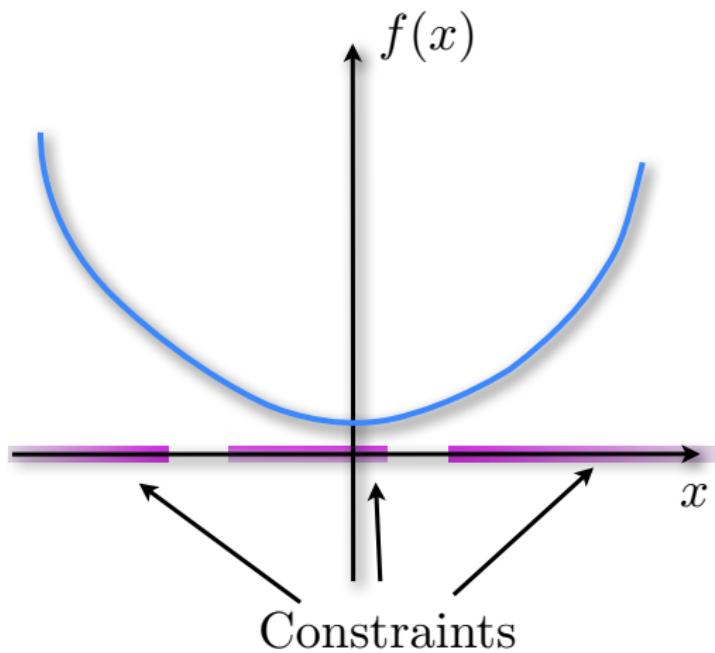
Proof.

Let \mathbf{x}^* be a stationary point, i.e., $\nabla f(\mathbf{x}^*) = 0$. By convexity, we have:

$$f(\mathbf{x}) \geq f(\mathbf{x}^*) + \langle \nabla f(\mathbf{x}^*), \mathbf{x} - \mathbf{x}^* \rangle \stackrel{\nabla f(\mathbf{x}^*)=0}{=} f(\mathbf{x}^*) \quad \text{for all } \mathbf{x} \in \mathbb{R}^p.$$

□

Is convexity of f enough for an iterative optimization algorithm?



Convexity over sets

Definition

- $\mathcal{Q} \subseteq \mathbb{R}^p$ is a **convex set** if $x_1, x_2 \in \mathcal{Q} \Rightarrow \forall \alpha \in [0, 1], \alpha x_1 + (1 - \alpha)x_2 \in \mathcal{Q}$.
- $\mathcal{Q} \subseteq \mathbb{R}^p$ is a **strictly convex set** if
 $x_1, x_2 \in \mathcal{Q} \Rightarrow \forall \alpha \in (0, 1), \alpha x_1 + (1 - \alpha)x_2 \in \text{interior}(\mathcal{Q})$.

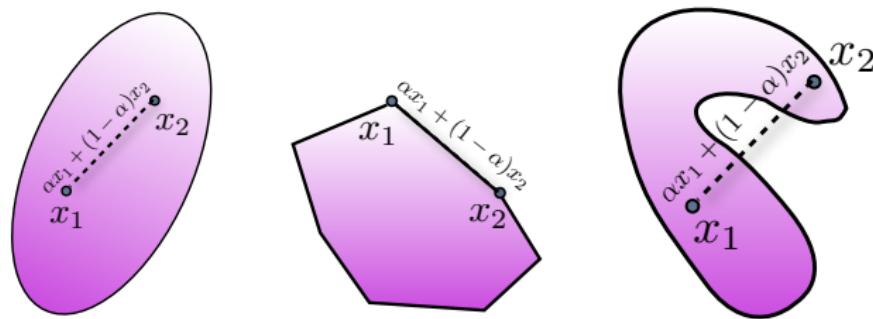


Figure: (Left) Strictly convex (Middle) Convex (Right) Non-convex

Convexity over sets

Definition

- ▶ $\mathcal{Q} \subseteq \mathbb{R}^p$ is a convex set if $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q} \Rightarrow \forall \alpha \in [0, 1], \alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \in \mathcal{Q}$.
- ▶ $\mathcal{Q} \subseteq \mathbb{R}^p$ is a *strictly convex* set if $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q} \Rightarrow \forall \alpha \in (0, 1), \alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \in \text{interior}(\mathcal{Q})$.

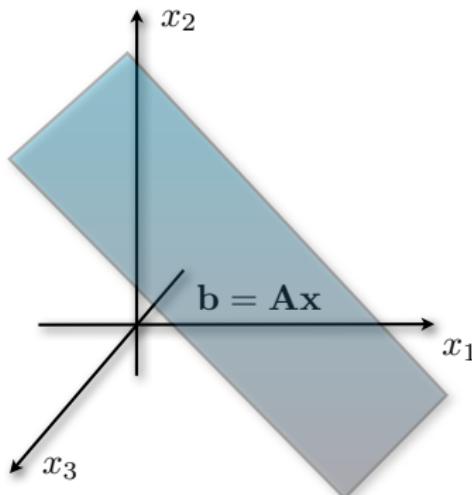


Figure: A linear set of equations $\mathbf{b} = \mathbf{Ax}$ defines an affine (thus convex) set.

Convexity over sets

Definition

- ▶ $\mathcal{Q} \subseteq \mathbb{R}^p$ is a convex set if $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q} \Rightarrow \forall \alpha \in [0, 1], \alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \in \mathcal{Q}$.
- ▶ $\mathcal{Q} \subseteq \mathbb{R}^p$ is a *strictly* convex set if
 $\mathbf{x}_1, \mathbf{x}_2 \in \mathcal{Q} \implies \forall \alpha \in (0, 1), \alpha\mathbf{x}_1 + (1 - \alpha)\mathbf{x}_2 \in \text{interior}(\mathcal{Q})$.



Why is this also important/useful?

- ▶ convex sets \Leftrightarrow convex optimization constraints

minimize
$$_{\mathbf{x}} f_0(\mathbf{x})$$

subject to constraints

Some basic notions on sets I

Definition (Closed set)

A set is called *closed* if it contains all its limit points.

Definition (Closure of a set)

Let $\mathcal{Q} \subseteq \mathbb{R}^p$ be a given open set, i.e., the limit points on the boundaries of \mathcal{Q} do not belong into \mathcal{Q} . Then, the closure of \mathcal{Q} , denoted as $\text{cl}(\mathcal{Q})$, is the smallest set in \mathbb{R}^p that includes \mathcal{Q} with its boundary points.



Figure: (Left panel) Closed set \mathcal{Q} . (Middle panel) Open set \mathcal{Q} and its closure $\text{cl}(\mathcal{Q})$ (Right panel).

Some basic notions on sets II

Definition (Interior)

Let $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, a point $\mathbf{x} \in \mathbb{R}^p$ is an *interior* of \mathcal{Q} if a neighborhood with radius r of \mathbf{x} is also included in \mathcal{Q} . That is, there exists $r > 0$, such that $\{\mathbf{y} : \|\mathbf{y} - \mathbf{x}\|_2 \leq r\} \in \mathcal{Q}$. The set of all interior points is denoted as $\text{int}(\mathcal{Q})$.

Example

- ▶ The interior of an open set is the set itself.
- ▶ The interior of the set $\{\mathbf{x} : \|\mathbf{y} - \mathbf{x}\|_2 \leq r\}$ is the open set $\{\mathbf{x} : \|\mathbf{y} - \mathbf{x}\|_2 < r\}$.

Some basic notions on sets II

Definition (Interior)

Let $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, a point $x \in \mathbb{R}^p$ is an *interior* of \mathcal{Q} if a neighborhood with radius r of x is also included in \mathcal{Q} . That is, there exists $r > 0$, such that $\{y : \|y - x\|_2 \leq r\} \in \mathcal{Q}$. The set of all interior points is denoted as $\text{int}(\mathcal{Q})$.

Example

- ▶ The interior of an open set is the set itself.
- ▶ The interior of the set $\{x : \|y - x\|_2 \leq r\}$ is the open set $\{x : \|y - x\|_2 < r\}$.

Definition (Relative interior)

Let $\mathcal{Q} \subseteq \mathbb{R}^p$. Then, a point $x \in \mathbb{R}^p$ is a *relative interior* of \mathcal{Q} if \mathcal{Q} contains the intersection of a neighborhood with radius r around x with the intersection of all affine sets containing \mathcal{Q} , i.e., $\text{aff}(\mathcal{Q})$. The set of all relative interior points is denoted as $\text{relint}(\mathcal{Q})$.

Example

The interior of the affine set $\mathcal{X} = \{x : Ax = b\}$ is empty. However, its relative interior is itself, i.e., $\text{relint}(\mathcal{X}) = \mathcal{X}$.

Convex hull

Definition (Convex hull)

Let $\mathcal{V} \subseteq \mathbb{R}^p$ be a set. The convex hull of \mathcal{V} , i.e., $\text{conv}(\mathcal{V})$, is the *smallest* convex set that contains \mathcal{V} .

Definition (Convex hull of points)

Let $\mathcal{V} \subseteq \mathbb{R}^p$ be a finite set of points with cardinality $|\mathcal{V}|$. The convex hull of \mathcal{V} is the set of all convex combinations of its points, i.e.,

$$\text{conv}(\mathcal{V}) = \left\{ \sum_{i=1}^{|\mathcal{V}|} \alpha_i \mathbf{x}_i : \sum_{i=1}^{|\mathcal{V}|} \alpha_i = 1, \alpha_i \geq 0, \forall i, \mathbf{x}_i \in \mathcal{V} \right\}.$$

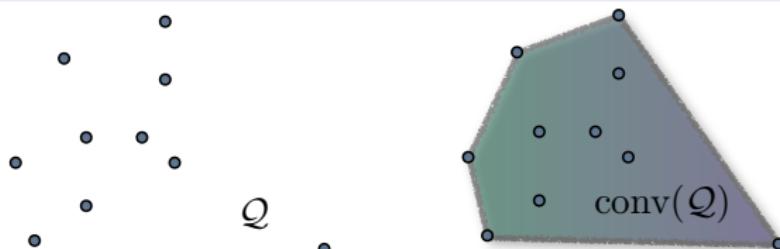


Figure: (Left) Discrete set of points \mathcal{V} . (Right) Convex hull $\text{conv}(\mathcal{V})$.

Revisiting: Alternative definitions of function convexity IV

Definition

The epigraph of a function $f : \mathcal{Q} \rightarrow \mathbb{R}$, $\mathcal{Q} \subseteq \mathbb{R}^p$ is the subset of \mathbb{R}^{p+1} given by:

$$\text{epi}(f) = \{(\mathbf{x}, w) : \mathbf{x} \in \mathcal{Q}, w \in \mathbb{R}, f(\mathbf{x}) \leq w\}.$$

Lemma

A function $f : \mathcal{Q} \rightarrow \mathbb{R}$ is convex if and only if its epigraph, i.e, the region above its graph, is a convex set.

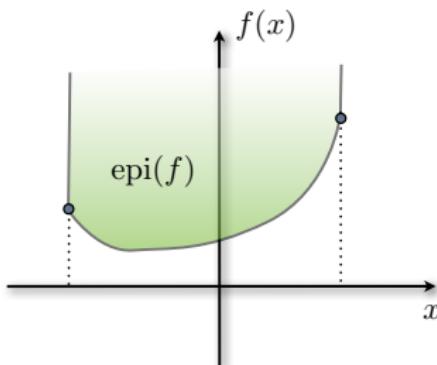


Figure: Epigraph — the region in green above graph $f(\cdot)$.

Unfortunately, convexity does not imply tractability

But first...

1. How do we define **tractability**?
2. How do we classify **running times** of algorithms?

Asymptotic Notation

What is this notation?

- ▶ Asymptotic Notation (Big-Oh Notation, Landau's notation) describes asymptotic growth of functions.
- ▶ It is usually used to describe:
 - ▶ Running time of an algorithm
 - ▶ Memory storage required by an algorithm
 - ▶ Error achieved by an approximation
- ▶ Exact computation of the running time, memory, or error is usually not important: For large inputs, multiplicative constants and lower-order terms “do not matter.”

Examples

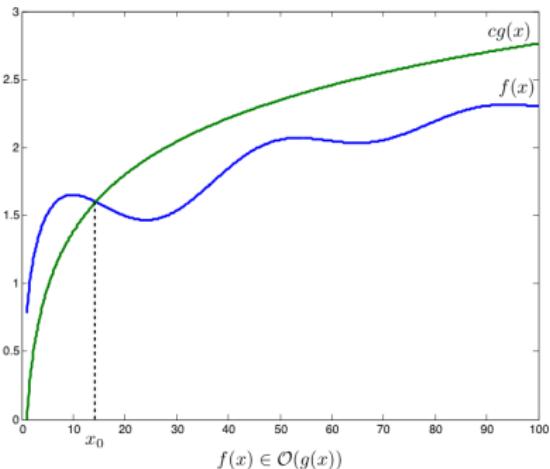
- ▶ Binary search’s running time in a sorted list of n elements. [1]: $O(\log(n))$
- ▶ Number of comparisons required for sorting a list of n elements [1]: $\Omega(n \log(n))$

Asymptotic Notation: Big-Oh

Definition (Big-Oh)

Let f, g be two functions defined on some subset of the real numbers:

$$f(x) \in O(g(x)) \text{ iff } \exists c > 0, \exists x_0, \text{ such that } |f(x)| \leq c|g(x)|, \forall x \geq x_0$$



- In computer science, the definition is taken over positive integers.

Example

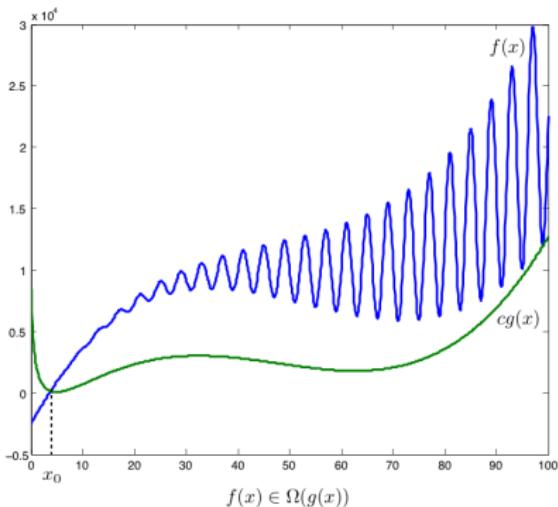
- $x \in O(x^2)$
- $\log(n!) \in O(n \log(n))$ [cf., lab 1]
- $n^{1+\sin(n)} \in O(n^2)$

Asymptotic Notation: Big-Omega

Definition (Big-Omega)

Let f, g be two functions defined on some subset of the real numbers:

$$f(x) \in \Omega(g(x)) \text{ iff } \exists c > 0, \exists x_0, \text{ such that } |f(x)| \geq c|g(x)|, \forall x \geq x_0$$



- ▶ **Intuition:** g is a lower bound of f iff f is an upper bound of g .
- ▶ $f(x) \in \Omega(g(x)) \Leftrightarrow g(x) \in O(f(x)).$

Example

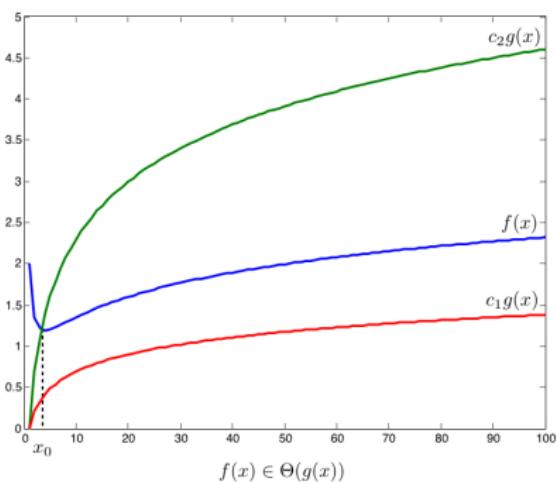
- ▶ $x^2 \in \Omega(x)$
- ▶ $\log(n!) \in \Omega(n \log(n))$ [cf., lab 1]
- ▶ $n^{1+\sin(n)} \in \Omega(1)$

Asymptotic Notation: Theta

Definition (Theta)

Let f, g be two functions defined on some subset of the real numbers:

$$f(x) \in \Theta(g(x)) \text{ iff } \exists c_1, c_2 > 0, \exists x_0, \text{ such that } c_1 \leq \frac{|f(x)|}{|g(x)|} \leq c_2, \forall x \geq x_0$$



- ▶ **Intuition:** g is a tight bound for f iff it is both an upper and a lower bound of it.
- ▶ $f(x) \in \Theta(g(x))$ iff $f(x) \in O(g(x))$ and $f(x) \in \Omega(g(x))$.
- ▶ $f(x) \in \Theta(g(x))$ iff $g(x) \in \Theta(f(x))$.

Example

- ▶ $\sin(x) \in \Theta(1)$
- ▶ $x + \log(x) \in \Theta(x)$
- ▶ $\log(n!) \in \Theta(n \log(n))$ [cf., lab 1]
- ▶ **Stirling's approximation:**
 $n! \in \sqrt{2\pi n} \left(\frac{n}{e}\right)^n \left(1 + \Theta\left(\frac{1}{n}\right)\right)$

Asymptotic Notation: small-oh and small-omega

Definition (small-oh, small-omega)

Let f, g be two functions defined on some subset of the real numbers:

$f(x) \in o(g(x))$ iff $\forall c > 0, \exists x_0$, such that $|f(x)| \leq c|g(x)|, \forall x \geq x_0$,

or equivalently $\lim_{x \rightarrow \infty} \frac{|f(x)|}{|g(x)|} = 0$.

$f(x) \in \omega(g(x))$ iff $\forall c > 0, \exists x_0$, such that $|f(x)| \geq c|g(x)|, \forall x \geq x_0$,

or equivalently $\lim_{x \rightarrow \infty} \frac{|f(x)|}{|g(x)|} = \infty$.

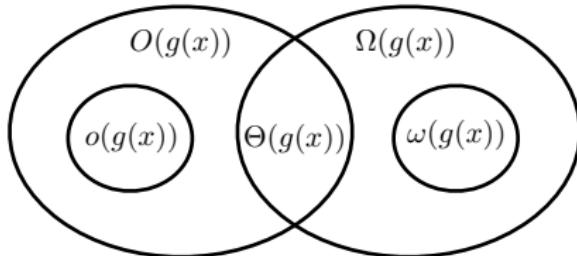
- These are **non-tight** upper/lower bounds.
- $g(x) \in o(f(x))$: g is dominated by f asymptotically.
- $f(x) \in \omega(g(x))$: f dominates g asymptotically.
- $f(x) \in \omega(g(x)) \Leftrightarrow g(x) \in o(f(x))$.

Example

- $\frac{1}{x} \in o(1)$
- $5 \in \omega(\frac{1}{x})$
- $n! \in o(n^n)$ [cf., lab 1]
- $n! \in \omega(2^n)$ [cf., lab 1]

Hierarchy of asymptotic notation classes

- Relation between the different asymptotic notations:



- Analogy with real numbers comparison:

Aymptotic function comparison	Real numbers comparison
$f(x) = O(g(x))$	$a \leq b$
$f(x) = \Omega(g(x))$	$a \geq b$
$f(x) = \Theta(g(x))$	$a = b$
$f(x) = o(g(x))$	$a < b$
$f(x) = \omega(g(x))$	$a > b$

- Difference from real numbers comparison: Not all functions are **asymptotically comparable**, e.g., n , $n^{1+\sin(n)}$.

Asymptotic Notation: some remarks

Some notation abuse:

- ▶ Use of equality: $f(x) = O(g(x))$

Some variations:

- ▶ Soft-Oh: $\tilde{O}(\cdot)$ notation ignores log terms, i.e., $O(x^c \log^k(x)) = \tilde{O}(x^c)$.
- ▶ Asymptotic notation can also describe limiting behavior as $x \rightarrow a$, e.g., $e^x = 1 + x + \frac{x^2}{2} + o(x^2), x \rightarrow 0$ (by Taylor's theorem).

Computational complexity: Complexity of deciding



Decision problems: “yes” or “no” answers.

How hard are these problems?

- ▶ **Shortest path:** Is there a path from point a to point b shorter than d ?
- ▶ **Subset sum problem:** Is there a subset of some given integers that sums up to d ?

For $d = 5$?

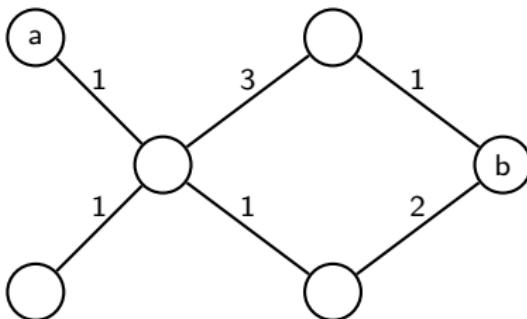


Figure: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

Applications:

- ▶ Driving directions in google maps.
- ▶ Minimum delay path for data packets in networking.

Computational complexity: Complexity of deciding



Decision problems: “yes” or “no” answers.

How hard are these problems?

- ▶ Shortest path: Is there a path from point a to point b shorter than d ?
- ▶ Subset sum problem: Is there a subset of some given integers that sums up to d ?

For $d = 5$? yes.

Shortest path can be computed by Dijkstra in $O(|\mathcal{V}|^2)$ [1]

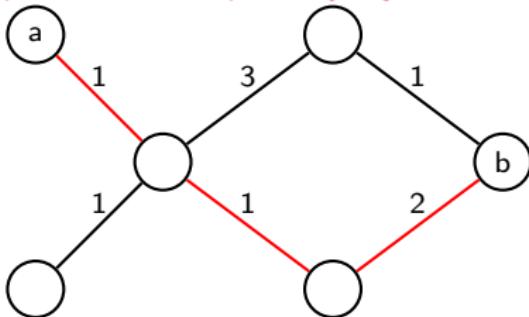


Figure: Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$.

Applications:

- ▶ Driving directions in google maps.
- ▶ Minimum delay path for data packets in networking.

Computational complexity: Complexity of deciding



Decision problems: “yes” or “no” answers.

How hard are these problems?

- ▶ Shortest path: Is there a path from point a to point b shorter than d ?
- ▶ Subset sum problem: Is there a subset of some given integers that sums up to d ?

For $d = 0$?

Consider these two lists of integers:

$$A : -2, 5, 4, 9, 19, -11$$

$$B : -2, 5, 4, 9, 19, -6$$

Applications:

- ▶ In cryptography:
public key system,
computer passwords,
message verification.

Computational complexity: Complexity of deciding



Decision problems: “yes” or “no” answers.

How hard are these problems?

- ▶ Shortest path: Is there a path from point a to point b shorter than d ?
- ▶ Subset sum problem: Is there a subset of some given integers that sums up to d ?

For $d = 0$?

Consider these two lists of integers:

$$A : -2, 5, 4, 9, 19, -11$$

Yes

$$B : -2, 5, 4, 9, 19, -6$$

No

Applications:

- ▶ In cryptography:
public key system,
computer passwords,
message verification.

Computational complexity: Complexity of deciding



Decision problems: “yes” or “no” answers.

How hard are these problems?

- ▶ Shortest path: Is there a path from point a to point b shorter than d ?
- ▶ Subset sum problem: Is there a subset of some given integers that sums up to d ?

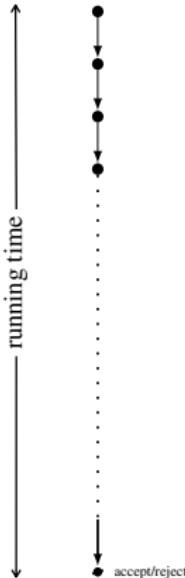
No known “efficient” algorithm can decide if a general list has a d -subset sum

-84	348	422	818	-364	44	-843	222	978	-934	452	83	-819	-100	474	179	-134
-951	515	-790	779	912	291	-560	585	-30	-255	693	-803	-932	-675	668	746	-323
46	-636	918	980	-943	22	-30	676	-766	-287	108	715	-352	530	673	-92	-91
979	239	-153	-353	-206	324	31	375	717	-451	685	-915	-668	967	-934	968	540
447	540	120	975	-445	-538	678	-838	-316	-471	-903	-716	997	-714	825	-480	512
-206	-158	-803	228	-276	966	-174	613	-367	489	518	-565	-169	656	-811	382	985
182	-122	-34	304	561	-829	-196	-731	-245	-737	352	555	509	763	702	194	-594
146	-22	-693	203	-109	-175	906	140	740	606	749	-285	387	-860	44	-442	-249
15	-567	-313	596	-886	971	-330	149	63	-391	-338	-950	542	-44	-938	-442	418
-709	666	-739	221	-904	-29	-733	-966	-950	-314	37	668	825	478	-742	366	-673
-418	3	-313	-246	62	224	391	870	-504	319	-92	-274	-379	468	186	623	-352
2	-795	326	-651	534	-978	846	230	448	-923	-641	577	-591	633	444	-848	618
-717	-271	32	-881	229	537	-25	337	-135	545	695	760	-855	67	-926	-261	-562
-187	642	594	-696	865	483	11	-157	-477	380	-908	353	174	122	-648	613	-343
186	-755	153	-233	563	-566	-991	406	-621	855	47	91	85	-307	-535	917	-76
619	299	-867	-974	510	-205	-623	-59	-571	-876	-535	798	288	102	430	984	-295
-59	473	-503	931	-348	-475	560	-182	-10	375	807	-801	196	-958	541	-149	-853

Applications:

- ▶ In cryptography:
public key system,
computer passwords,
message verification.

Computational complexity: Class \mathcal{P}



Definition (Class \mathcal{P})

\mathcal{P} (polynomial time): decision problems solvable in polynomial time.

Definition (Turing machine)

(Deterministic) Turing machine (DTM): mathematical computational model, think of it as your regular computer. For formal definition, refer to [4].

- ▶ Problems in \mathcal{P} can be solved by a DTM in polynomial time.
- ▶ Polynomial time means $O(n^k)$ time for some $k \in \mathbb{N}$, where n is the size of the input.

Example (Shortest path problem)

Shortest path can be computed by Dijkstra in $O(|\mathcal{V}|^2)$ [1]

Figure: Deterministic Turing machine

Computational complexity: Class \mathcal{NP}

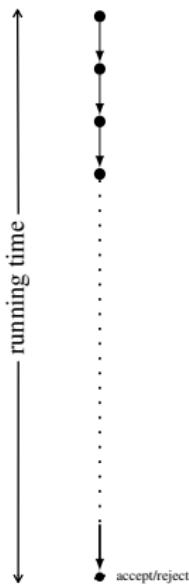


Figure: Deterministic Turing machine

Definition (Class \mathcal{NP})

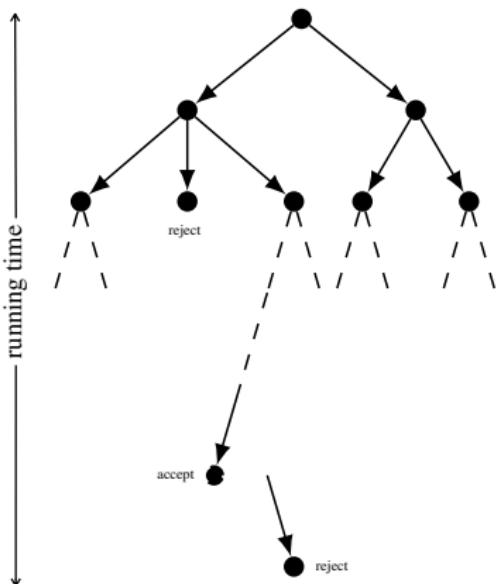
\mathcal{NP} (nondeterministic polynomial time): decision problems such that “yes” answer can be “checked” in poly-time via a deterministic Turing machine, i.e., there exists a certificate (proof) whose correctness can be verified in poly-time.

- ▶ Polynomial time means $O(n^k)$ time for some $k \in \mathbb{N}$, where n is the size of the input.
- ▶ It follows then that the certificate should be of **polynomial length**.
- ▶ $\mathcal{P} \subseteq \mathcal{NP}$

Example

- ▶ Subset sum problem:
 - ▶ Proof: Subset of integers that do sum up to d .
 - ▶ Verification: Addition of can be done in polynomial time.
- ▶ Shortest path problem.

Computational complexity: Class \mathcal{NP}



Definition (Class \mathcal{NP})

\mathcal{NP} (nondeterministic polynomial time): decision problems such that “yes” answer can be “checked” in poly-time via a deterministic Turing machine, i.e., there exists a certificate (proof) whose correctness can be verified in poly-time.

Definition (Non-deterministic TM)

Non-deterministic Turing machine (NTM): A fictional “super” computer than can “clone” itself every time it reaches a decision, each clone continue with one of the possible choices. For formal definition, refer to [4].

- ▶ Problems in \mathcal{NP} can be solved by a **NTM** in **polynomial** time.

Figure: Non-Deterministic (decider)
Turing machine

Computational complexity: Open Problem

Open problem: \mathcal{P} vs \mathcal{NP}

- ▶ $\mathcal{P} \stackrel{?}{=} \mathcal{NP}$: Is generating a proof as easy as checking it ?
- ▶ One of the 7 Millennium Prize Problems by the Clay Mathematics Institute.
- ▶ Conjecture: $\mathcal{P} \neq \mathcal{NP}$.
- ▶ Many other open problems in complexity.

Computational complexity: Reductions

Definition (Polynomial time reducibility)

- ▶ **Reduction** is a way of converting one problem to another such that the solution to the second problem can be used to solve the first problem.
 - ▶ **Polynomial time reduction (mapping reducibility) [4]**: We say a problem A is poly-time mapping reducible to B , denoted by $A \leq_p B$, iff there exists polynomial time computable function f that converts any input of A into an input of B , such that the transformed problem has the same output as the original problem.
 - ▶ A 's answer is yes on input $w \Leftrightarrow B$'s answer is yes on input $f(w)$.
-
- ▶ If A is poly-time reducible to B , then the existence of a polynomial algorithm of B would imply the existence of a polynomial algorithm for A also.



Computational complexity: Hardness

Definition (\mathcal{NP} -Hard)

- ▶ **\mathcal{NP} -Hard:** Problems (not necessarily decision) that are at least as hard as the hardest problems in \mathcal{NP} .
- ▶ $B \in \mathcal{NP}\text{-Hard} \Leftrightarrow \forall A \in \mathcal{NP}, A \leq_p B$.
- ▶ Examples: search version of subset sum, candy crush.

Definition (\mathcal{NP} -Complete)

- ▶ **\mathcal{NP} -Complete:** Decision problems in \mathcal{NP} , that are at least as hard as the hardest problems in \mathcal{NP} .
- ▶ $\mathcal{NP}\text{-Complete} = \mathcal{NP} \cap \mathcal{NP}\text{-Hard}$.
- ▶ Subset sum, Karp's 21 NP-complete problems [2].

Computational complexity: Class coNP

Definition (Class coNP)

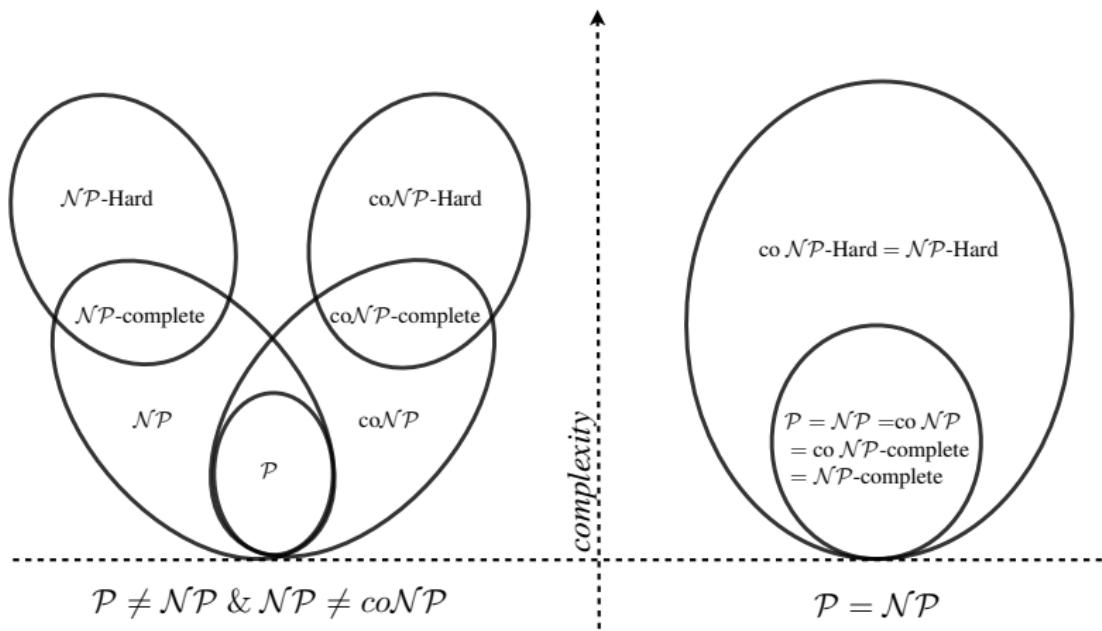
coNP (complement nondeterministic polynomial time): decision problems such that “no” answer can be “checked” in poly-time, i.e., there exists a certificate (or a counter-example) whose correctness can be verified in poly-time.

Example (subset sum $\in \text{coNP}$)

Does every non-empty subset sums up to a non-zero sum?

- ▶ $A \in \text{coNP}$ iff $\overline{A} \in \text{NP}$
- ▶ $\mathcal{P} \in \text{coNP} \cap \text{NP}$
- ▶ $B \in \text{coNP-Hard} \Leftrightarrow \forall A \in \text{coNP}, A \leq_p B$
- ▶ coNP-Complete : $\text{coNP} \cap \text{coNP-Hard}$.

Two of the possible worlds



Hardness result: Certifying optimality in mathematical programming

Problem (Smooth constrained optimization problems)

We consider *smooth constrained optimization problems*:

$$\min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x}) \text{ such that } g_i(\mathbf{x}) \leq 0, \forall i \in [1, \dots, m]$$

Smooth: we assume that f and all g_i 's are infinitely differentiable.

How *hard* is it to *check* that a given solution $\mathbf{x} \in \mathbb{R}^p$ is optimal?

Why should we care ?

- ▶ Optimization is ubiquitous: applications in control, estimation, signal processing, electronics design, communications, finance, ...
- ▶ Emphasize the importance of convexity: smoothness alone is not enough.

Hardness result: Certifying optimality in mathematical programming

Problem (Smooth constrained optimization problems)

We consider *smooth constrained* optimization problems:

$$\min_{\mathbf{x} \in \mathbb{R}^p} f(\mathbf{x}) \text{ such that } g_i(\mathbf{x}) \leq 0, \forall i \in [1, \dots, m]$$

Smooth: we assume that f and all g_i 's are infinitely differentiable.

How *hard* is it to *check* that a given solution $\mathbf{x} \in \mathbb{R}^p$ is optimal?

Example (Subset sum problem revisited)

Subset Sum problem: Given integers $d_1 \dots d_p$, is there a non-empty subset that sums up to d_0 ?

Equivalent smooth constrained optimization problem:

$$\min_{\mathbf{y} \in \mathbb{R}^p} \left(\sum_{i=1}^p d_i y_i - d_0 \right)^2 + \sum_{i=1}^p y_i(1 - y_i) \text{ such that } 0 \leq y_i \leq 1, \forall i \in [1, \dots, p]$$

Hardness result: certifying optimality in mathematical programming

Proposition

Checking if a point is the global minimum of a smooth constrained optimization problem is NP-Hard [3] in general.

Proof.

Reduce an NP-complete problem to our problem:

- ▶ B: “Compute the global minimum of a smooth constrained optimization problem”.
- ▶ Subset sum problem (A) is known to be NP-complete.
- ▶ It has an equivalent formulation:

$$\min_{\mathbf{y} \in \mathbb{R}^p} \left(\sum_{i=1}^p d_i y_i - d_0 \right)^2 + \sum_{i=1}^p y_i (1-y_i) \text{ such that } 0 \leq y_i \leq 1, \forall i \in [1, \dots, p] \quad (1)$$

- ▶ Zero is the global minimum objective value of (1) iff there exists a subset of $[d_1, \dots, d_p]$ with sum d_0 .
- ▶ We showed that $A \leq_p B$, and $A \in \mathcal{NP}$ -complete, then $B \in \mathcal{NP}$ -Hard.



Hardness result: Certifying optimality in mathematical programming

Proposition

Checking if a point is a local minimum of a smooth constrained optimization problem is coNP-Hard [3] in general.

We need a structure beyond smoothness that avoids such problems: Convexity?

Unfortunately, convexity does not imply tractability

Consider the following NP-Hard problem:

Problem (Maximum Cut)

Given a graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, such that $n = |\mathcal{V}|, m = |\mathcal{E}|$, the maximum cut problem is the problem of finding a cut (i.e., a partition of the vertices of a graph into two disjoint subsets S and \bar{S}) with a cut-set (edges between S and \bar{S}) of maximum weight.

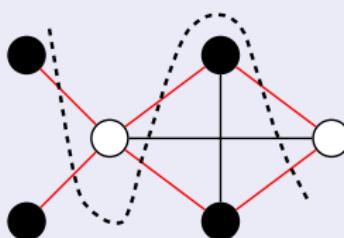


Figure: The set S of black nodes corresponds to the cut-set $\delta(S)$ of red edges.

Max-Cut problem can be formulated as: $\max_{S \subseteq \mathcal{V}} \mathbf{w}^T \delta(S)$, where $\mathbf{w} \in \mathbb{R}^m$ denote the edge weights.

Unfortunately, convexity does not imply tractability

Example (Cut polytope)

Consider the following smooth **convex** constrained optimization problem:

$$\max_{\mathbf{x} \subseteq \text{Cut}_n} \mathbf{w}^T \mathbf{x} \quad (2)$$

where Cut_n is the convex hull of the characteristic vectors of cut sets, i.e.,
 $\text{Cut}_n = \text{conv}(\{\mathbf{1}_{\mathcal{S}}, \mathcal{S} \in \mathcal{V}\})$. It is called the cut polytope. Problem (2) is **NP-Hard**,
since Max-Cut problem can be reformulated as (2).

Convexity is still helpful

- ▶ Convexity does not imply tractability in general.
- ▶ Convexity implies that finding a local minimum is enough to find a global minimum.

References |

- [1] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein.
Introduction To Algorithms.
MIT Press, 2001.
- [2] Richard M Karp.
Reducibility among combinatorial problems.
Springer, 1972.
- [3] Katta G Murty and Santosh N Kabadi.
Some np-complete problems in quadratic and nonlinear programming.
Mathematical programming, 39(2):117–129, 1987.
- [4] Michael Sipser.
Introduction to the Theory of Computation.
Cengage Learning, 2012.