Introduction to Optimization

Romain Couillet and Ronald Phlypo

December 3, 2020

Outline

Outline

Main objective

Objective of the class: solve the problem

$$\text{Find } x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} f(x) \tag{1}$$

for some function $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$.

Main objective

Objective of the class: solve the problem

$$\text{Find } x^\star \in \operatorname{argmin}_{x\in\Omega\subset\mathcal{X}} f(x) \tag{1}$$

for some function $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$.

Remark

$\operatorname{argmin}_{x\in\Omega\subset\mathcal{X}} f(x)$ *is a subset of* $\mathcal{X}$ *(may be empty, a singleton, a discrete set, an uncountable set).*

Main objective

Objective of the class: solve the problem

$$\text{Find } x^\star \in \operatorname{argmin}_{x\in\Omega\subset\mathcal{X}} f(x) \tag{1}$$

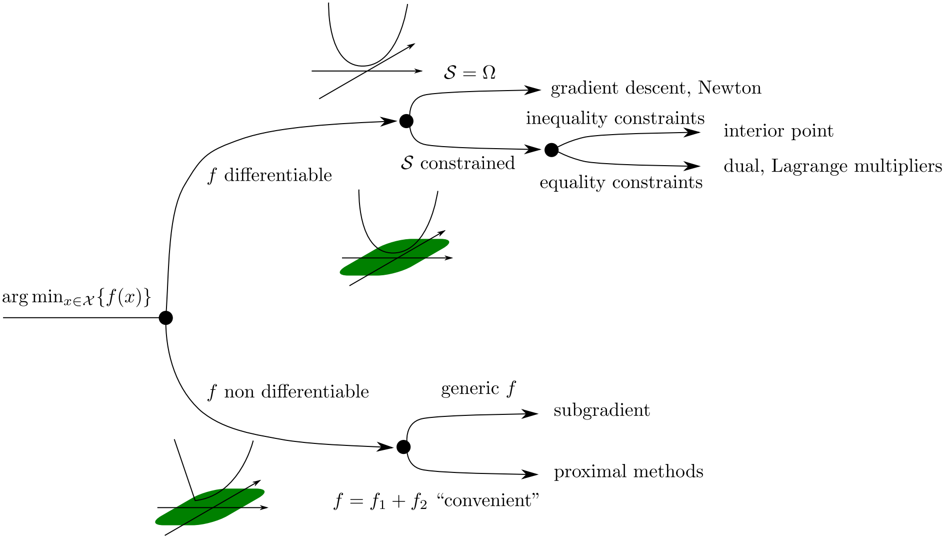
for some function $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$.

Remark

$\operatorname{argmin}_{x\in\Omega\subset\mathcal{X}} f(x)$ *is a subset of $\mathcal{X}$ (may be empty, a singleton, a discrete set, an uncountable set).*

- f is the *cost*, *penalty*, or *objective* function;
- $\Omega = \mathcal{S} \cap \mathcal{X}$ is the set of constraints $\mathcal{S}$ restricted to $\mathcal{X}$.

Specifying f



Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Setting:

- n assets;
- at time t, return $[x_t]_i$ for asset i, with $\mathbb{E}[x_t] = \mu$ and $\mathrm{Cov}[x_t] = C$;
- investment of wealth 1 across assets $[w]_1, \ldots, [w]_n$, $\sum_{i=1}^n [w]_i = 1$.

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Setting:

- n assets;
- at time t, return $[x_t]_i$ for asset i, with $\mathbb{E}[x_t] = \mu$ and $\mathrm{Cov}[x_t] = C$;
- investment of wealth 1 across assets $[w]_1, \ldots, [w]_n$, $\sum_{i=1}^n [w]_i = 1$.

Objective:

- Optimal expected gain:

$$\operatorname{argmax}_{w \in \mathbb{R}^n} \mathbb{E}[w^\mathsf{T} x_t] = w^\mathsf{T} \mu, \text{ such that } \sum_{i=1}^n [w]_i = 1.$$

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Setting:

- n assets;
- at time t, return $[x_t]_i$ for asset i, with $\mathbb{E}[x_t] = \mu$ and $\mathrm{Cov}[x_t] = C$;
- investment of wealth 1 across assets $[w]_1, \ldots, [w]_n$, $\sum_{i=1}^n [w]_i = 1$.

Objective:

- Optimal expected gain:

$$\mathrm{argmax}_{w \in \mathbb{R}^n} \mathbb{E}[w^\mathsf{T} x_t] = w^\mathsf{T} \mu, \text{ such that } \sum_{i=1}^n [w]_i = 1.$$

- Risk minimization:

$$\mathrm{argmin}_{w \in \mathbb{R}^n} \mathbb{E}[|w^\mathsf{T}(x_t - \mu)|^2], \text{ such that } \sum_{i=1}^n [w]_i = 1.$$

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Setting:

- n assets;
- at time t, return $[x_t]_i$ for asset i, with $\mathbb{E}[x_t] = \mu$ and $\mathrm{Cov}[x_t] = C$;
- investment of wealth 1 across assets $[w]_1, \ldots, [w]_n$, $\sum_{i=1}^n [w]_i = 1$.

Objective:

- Optimal expected gain:

$$\operatorname{argmax}_{w \in \mathbb{R}^n} \mathbb{E}[w^\mathsf{T} x_t] = w^\mathsf{T} \mu, \text{ such that } \sum_{i=1}^n [w]_i = 1.$$

- Risk minimization:

$$\operatorname{argmin}_{w \in \mathbb{R}^n} \mathbb{E}[|w^\mathsf{T}(x_t - \mu)|^2], \text{ such that } \sum_{i=1}^n [w]_i = 1.$$

- Risk minimization under constrained expected gain g:

$$\operatorname{argmin}_{w \in \mathbb{R}^n} \mathbb{E}[|w^\mathsf{T}(x_t - \mu)|^2], \text{ such that } \sum_{i=1}^n [w]_i = 1 \text{ and } \mathbb{E}[w^\mathsf{T} x_t] \geq g.$$

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Objective:

- Risk minimization with non-negativity constraint:

$$\mathrm{argmin}_{w \in \mathbb{R}^n} \mathbb{E}[|w^\mathsf{T}(x_t - \mu)|^2], \text{ such that } \sum_{i=1}^{n} [w]_i = 1 \text{ and } \forall i,\ [w]_i \geq 0.$$

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Objective:

- Risk minimization with non-negativity constraint:

$$\operatorname{argmin}_{w\in\mathbb{R}^n} \mathbb{E}[|w^\top(x_t-\mu)|^2], \text{ such that } \sum_{i=1}^n [w]_i = 1 \text{ and } \forall i,\ [w]_i \geq 0.$$

Overview:

- Without inequality constraint, Lagrange multipliers give the solution:

$$w^\star = \frac{C^{-1}1_n}{1_n^\top C^{-1}1_n}.$$

Examples: the Lab Sessions

Example (1. Portfolio Optimization)

Objective:

- Risk minimization with non-negativity constraint:

$$\operatorname{argmin}_{w\in\mathbb{R}^n} \mathbb{E}[|w^\top(x_t-\mu)|^2], \text{ such that } \sum_{i=1}^n [w]_i = 1 \text{ and } \forall i,\ [w]_i \geq 0.$$

Overview:

- Without inequality constraint, Lagrange multipliers give the solution:

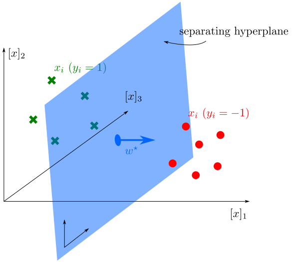
$$w^\star = \frac{C^{-1}1_n}{1_n^\top C^{-1}1_n}.$$

- With inequality constraint, interior point method (Lab Session 1), or proximal point method (Lab Session 2).

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.

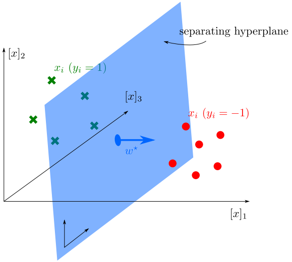


Examples: the Lab Sessions

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.



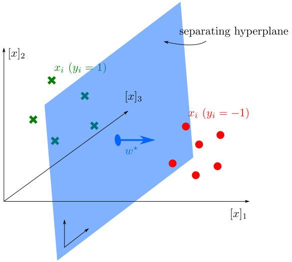
Objective: Maximize hyperplane "margin",

Examples: the Lab Sessions

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.



Objective: Maximize hyperplane "margin", or equivalently

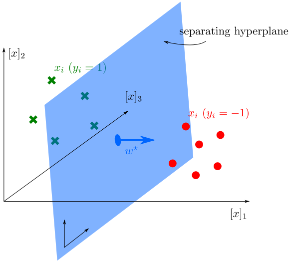
$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^n} \left\{ \|w\|^2 \right\} \text{ such that } y_i(w^\mathsf{T} x_i + b) \geq 1.$$

Examples: the Lab Sessions

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.



Objective: Maximize hyperplane "margin", or equivalently

$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^n} \left\{ \|w\|^2 \right\} \text{ such that } y_i(w^\mathsf{T} x_i + b) \geq 1.$$

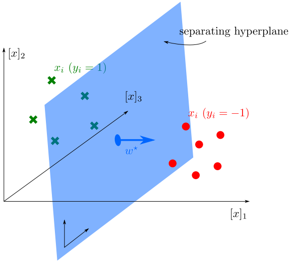
Why? Distance between "supporting" hyperplanes $\mathcal{H}_{\pm 1} : x^\mathsf{T} w^\star + b^\star = \pm 1$ for all $\|x_{+1} - x_{-1}\|$, $x_{\pm 1} \in \mathcal{H}_{\pm 1}$: implies $(x_{+1} - x_{-1})^\mathsf{T} w^\star = 2$. Distance max for $\|w^\star\|$ min.

Examples: the Lab Sessions

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.



Objective: Maximize hyperplane "margin", or equivalently

$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^n} \left\{\|w\|^2\right\} \text{ such that } y_i(w^\mathsf{T} x_i + b) \geq 1.$$

Why? Distance between "supporting" hyperplanes $\mathcal{H}_{\pm 1} : x^\mathsf{T} w^\star + b^\star = \pm 1$ for all $\|x_{+1} - x_{-1}\|$, $x_{\pm 1} \in \mathcal{H}_{\pm 1}$: implies $(x_{+1} - x_{-1})^\mathsf{T} w^\star = 2$. Distance max for $\|w^\star\|$ min.

But argmin can be empty! Relaxation to "soft-margin" SVM:

$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^p} \left\{ \frac{1}{m} \sum_{i=1}^{m} \max(0, 1 - y_i[w^\mathsf{T} x_i + b]) + \lambda \|w\|^2 \right\}$$

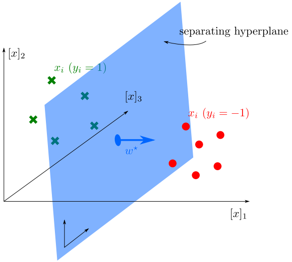
for some $\lambda > 0$.

Examples: the Lab Sessions

Example (2. Support Vector Machines)

Setting:

- Data points and labels $(x_1, y_1), \ldots, (x_m, y_m) \in \mathbb{R}^n \times \{\pm 1\}$;
- Separating hyperplane of $\mathbb{R}^n$ of the form $\mathcal{H} = \{x \mid x^\mathsf{T} w^\star + b^\star = 0\}$.



Objective: Maximize hyperplane "margin", or equivalently

$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^n} \left\{ \|w\|^2 \right\} \text{ such that } y_i(w^\mathsf{T} x_i + b) \geq 1.$$

Why? Distance between "supporting" hyperplanes $\mathcal{H}_{\pm 1} : x^\mathsf{T} w^\star + b^\star = \pm 1$ for all $\|x_{+1} - x_{-1}\|$, $x_{\pm 1} \in \mathcal{H}_{\pm 1}$: implies $(x_{+1} - x_{-1})^\mathsf{T} w^\star = 2$. Distance max for $\|w^\star\|$ min.

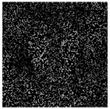
But argmin can be empty! Relaxation to "soft-margin" SVM:

$$(w^\star, b^\star) \in \operatorname{argmin}_{w,b \in \mathbb{R}^p} \left\{ \frac{1}{m} \sum_{i=1}^{m} \max(0, 1 - y_i[w^\mathsf{T} x_i + b]) + \lambda \|w\|^2 \right\}$$

for some $\lambda > 0$.

Solution: Interior point or proximal methods.

Examples: the Lab Sessions

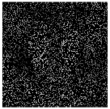




Example (3. Compressive Sensing)

Setting:

- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

Examples: the Lab Sessions

Example (3. Compressive Sensing)

Setting:

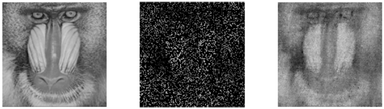
- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

Objective: Maximize sparsity via "ℓ_1-relaxation"

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \|x\|_1 \text{ such that } y = Ax$$

with $\|x\|_1 = \sum_{i=1}^n |[x]_i|$.

Examples: the Lab Sessions



Example (3. Compressive Sensing)

Setting:

- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

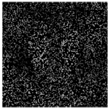
Objective: Maximize sparsity via "ℓ_1-relaxation"

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \|x\|_1 \text{ such that } y = Ax$$

with $\|x\|_1 = \sum_{i=1}^n |[x]_i|$.

Remark 1: $\|\cdot\|_1$ is not differentiable.

Examples: the Lab Sessions

Example (3. Compressive Sensing)

Setting:

- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

Objective: Maximize sparsity via "ℓ_1-relaxation"

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \|x\|_1 \text{ such that } y = Ax$$

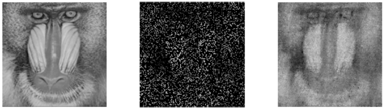
with $\|x\|_1 = \sum_{i=1}^n |[x]_i|$.

Remark 1: $\|\cdot\|_1$ is not differentiable.

Remark 2: Denoting $\imath_\Omega(x) = 0$ if $x \in \Omega$ and $\imath_\Omega(x) = +\infty$ if $x \notin \Omega$,

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{ \|x\|_1 + \imath_{\{y=Ax\}} \right\}$$

Examples: the Lab Sessions



Example (3. Compressive Sensing)

Setting:

- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

Objective: Maximize sparsity via "ℓ_1-relaxation"

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \|x\|_1 \text{ such that } y = Ax$$

with $\|x\|_1 = \sum_{i=1}^n |[x]_i|$.

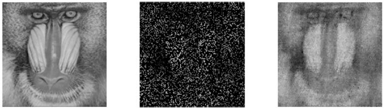
Remark 1: $\|\cdot\|_1$ is not differentiable.

Remark 2: Denoting $\imath_\Omega(x) = 0$ if $x \in \Omega$ and $\imath_\Omega(x) = +\infty$ if $x \notin \Omega$,

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{ \|x\|_1 + \imath_{\{y=Ax\}} \right\} \equiv \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{ f_1(x) + f_2(x) \right\}$$

with f_1, f_2 convex non-differentiable.

Examples: the Lab Sessions



Example (3. Compressive Sensing)

Setting:

- retrieve $x \in \mathbb{R}^n$ from $y = Ax \in \mathbb{R}^p$, $p \ll n$, with x a *sparse* vector;

Objective: Maximize sparsity via "ℓ_1-relaxation"

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \|x\|_1 \text{ such that } y = Ax$$

with $\|x\|_1 = \sum_{i=1}^n |[x]_i|$.

Remark 1: $\|\cdot\|_1$ is not differentiable.

Remark 2: Denoting $\imath_\Omega(x) = 0$ if $x \in \Omega$ and $\imath_\Omega(x) = +\infty$ if $x \notin \Omega$,

$$x^\star \in \operatorname{argmin}_{x \in \mathbb{R}^n} \left\{\|x\|_1 + \imath_{\{y=Ax\}}\right\} \equiv \operatorname{argmin}_{x \in \mathbb{R}^n} \{f_1(x) + f_2(x)\}$$

with f_1, f_2 convex non-differentiable.

Solution: Proximal methods and the *Douglas-Rachford splitting* algorithm.

Outline

Outline

Convex Sets

Definition (Convex Set)

$\mathcal{C} \subset \mathcal{X}$ convex iif $\forall x, y \in \mathcal{C}$ and $\forall \lambda \in [0, 1]$,

$$(1-\lambda)x + \lambda y = x + \lambda(y - x) \in \mathcal{C}.$$

Convex Sets

Definition (Convex Set)

$\mathcal{C} \subset \mathcal{X}$ convex iif $\forall x, y \in \mathcal{C}$ and $\forall \lambda \in [0, 1]$,

$$(1-\lambda)x + \lambda y = x + \lambda(y - x) \in \mathcal{C}.$$

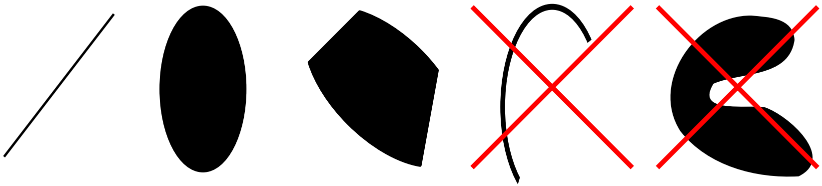


Figure: Convex sets and non-convex sets (stroke out).

Convex Sets: basic properties

Remark (Ensemble manipulations on convex sets)

For convex sets $\mathcal{C}_1$, $\mathcal{C}_2$,

- $\mathcal{C}_i$ *can be open, closed, bounded, unbounded.*
- $\mathcal{C}_1 \cap \mathcal{C}_2$ *is convex.*
- $\mathcal{C}_1 \cup \mathcal{C}_2$ *is not necessarily convex.*

Convex Sets: basic properties

Remark (Ensemble manipulations on convex sets)

For convex sets $\mathcal{C}_1$, $\mathcal{C}_2$,

- $\mathcal{C}_i$ *can be open, closed, bounded, unbounded.*
- $\mathcal{C}_1 \cap \mathcal{C}_2$ *is convex.*
- $\mathcal{C}_1 \cup \mathcal{C}_2$ *is not necessarily convex.*

Remark (List of convex sets)

The following ensembles are convex:

- *line, segment, half-line,* $\mathbb{R}^n$
- *a vector subspace*
- *hyperplanes* $\{x,\ x^\mathsf{T} a = b\}$, *half-spaces* $\{x,\ x^\mathsf{T} a \leq b\}$
- *balls* $\mathcal{B}(x_c; r) \equiv \{x,\ \|x - x_c\| \leq r\}$ *and ellipsoids* $\{x,\ (x - x_c)^\mathsf{T} P^{-1}(x - x_c) \leq r\}$.

Convex Sets: basic properties

Exercise (1. Ball convexity)

Show that $\mathcal{B}(x_c; r) \equiv \{x,\ \|x - x_c\| \leq r\}$ *is convex.*

Convex Sets: basic properties

Exercise (1. Ball convexity)

Show that $\mathcal{B}(x_c; r) \equiv \{x, \ \|x - x_c\| \leq r\}$ *is convex.*

Proof of ball convexity.

Let $x, y \in \mathcal{B}(x_c; r)$. Then,

$$\|\lambda x + (1-\lambda) y - x_c\| = \|\lambda(x - x_c) + (1-\lambda)(y - x_c)\|$$

Convex Sets: basic properties

Exercise (1. Ball convexity)

Show that $\mathcal{B}(x_c; r) \equiv \{x,\ \|x - x_c\| \le r\}$ *is convex.*

Proof of ball convexity.

Let $x, y \in \mathcal{B}(x_c; r)$. Then,

$$\|\lambda x + (1-\lambda)y - x_c\| = \|\lambda(x - x_c) + (1-\lambda)(y - x_c)\| \le \lambda\|x - x_c\| + (1-\lambda)\|y - x_c\| \le r.$$

□

Exercise (1. Ball convexity)

Show that $\mathcal{B}(x_c; r) \equiv \{x,\ \|x - x_c\| \leq r\}$ *is convex.*

Proof of ball convexity.

Let $x, y \in \mathcal{B}(x_c; r)$. Then,

$$\|\lambda x + (1-\lambda)y - x_c\| = \|\lambda(x - x_c) + (1-\lambda)(y - x_c)\| \leq \lambda\|x - x_c\| + (1-\lambda)\|y - x_c\| \leq r.$$

$\square$

Exercise (2. Polyhedron convexity)

For $A \in \mathbb{R}^{l\times n}$, $B \in \mathbb{R}^{m\times n}$ *and* $b \in \mathbb{R}^l$, $d \in \mathbb{R}^m$, *show the convexity of polyhedron*

$$\mathcal{P} = \{x,\ Ax \leq b,\ Cx = d\}\,.$$

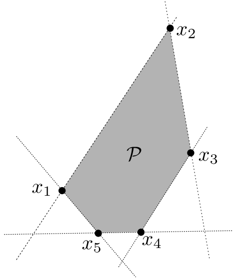


Figure: A polyhedron.

Basic properties

Definition (Convex combinations)

The set of convex combinations of $x_1, \ldots, x_k \in \mathcal{S}$ is the set

$$\left\{ \theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0 \right\}.$$

Basic properties

Definition (Convex combinations)

The set of convex combinations of $x_1, \ldots, x_k \in \mathcal{S}$ is the set

$$\left\{\theta_1 x_1 + \ldots + \theta_k x_k \;\middle|\; \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0\right\}.$$

This is a convex set.

Basic properties

Definition (Convex combinations)

The set of convex combinations of $x_1, \ldots, x_k \in \mathcal{S}$ is the set

$$\left\{ \theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0 \right\}.$$

This is a convex set.

The polyhedron (Figure 2) is the set of convex combinations of $x_1, \ldots, x_5$.

Basic properties

Definition (Convex combinations)

The set of convex combinations of $x_1, \ldots, x_k \in \mathcal{S}$ is the set

$$\left\{\theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0\right\}.$$

This is a convex set.

The polyhedron (Figure 2) is the set of convex combinations of $x_1, \ldots, x_5$.

Definition (Convex hull)

The convex hull $\mathrm{conv}(\mathcal{X})$ is the set of all convex combinations of points in $\mathcal{X}$,

$$\mathrm{conv}(\mathcal{X}) = \left\{\theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0,\ x_1, \ldots, x_k \in \mathcal{X},\ k \geq 0\right\}.$$

Basic properties

Definition (Convex combinations)

The set of convex combinations of $x_1, \ldots, x_k \in \mathcal{S}$ is the set

$$\left\{ \theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0 \right\}.$$

This is a convex set.

The polyhedron (Figure 2) is the set of convex combinations of $x_1, \ldots, x_5$.

Definition (Convex hull)

The convex hull $\operatorname{conv}(\mathcal{X})$ is the set of all convex combinations of points in $\mathcal{X}$,

$$\operatorname{conv}(\mathcal{X}) = \left\{ \theta_1 x_1 + \ldots + \theta_k x_k \mid \sum_{i=1}^{k} \theta_i = 1,\ \theta_1, \ldots, \theta_k \geq 0,\ x_1, \ldots, x_k \in \mathcal{X},\ k \geq 0 \right\}.$$

Property (Convex sets and convex hulls)

$\operatorname{conv}(\mathcal{X})$ *is the smallest convex set containing* $\mathcal{X}$*:* $\mathcal{X}$ *is convex iif* $\mathcal{X} = \operatorname{conv}(\mathcal{X})$.

Outline

Convex Function

Definition (Epigraph of a function)

The epigraph of $f : \mathcal{X} \to \mathbb{R}$ is the set

$$\mathrm{epi}(f) = \{(x, c) \in \mathcal{X} \times \mathbb{R},\ f(x) \leq c\}\,.$$

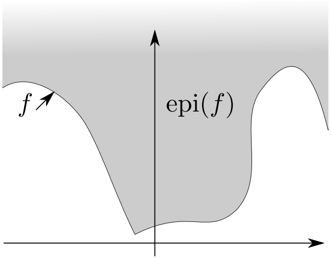


Figure: Epigraph of a function $f : \mathbb{R} \to \mathbb{R}$.

Convex Function

Definition (Epigraph of a function)

The epigraph of $f : \mathcal{X} \to \mathbb{R}$ is the set

$$\mathrm{epi}(f) = \{(x, c) \in \mathcal{X} \times \mathbb{R},\ f(x) \leq c\}\,.$$

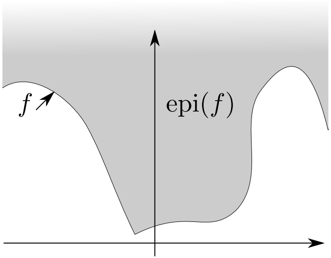


Figure: Epigraph of a function $f : \mathbb{R} \to \mathbb{R}$.

Definition (Convex function)

A function $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif $\mathrm{epi}(f)$ is a convex set.

Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0, 1]$,

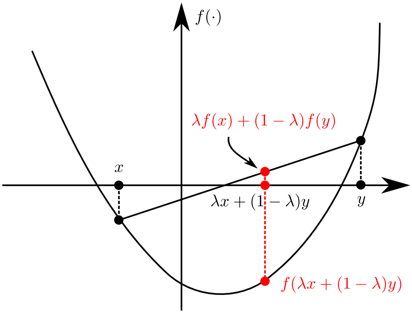
$$f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$,

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

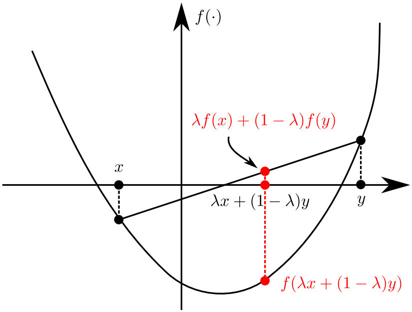


Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0, 1]$,

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



Proof.

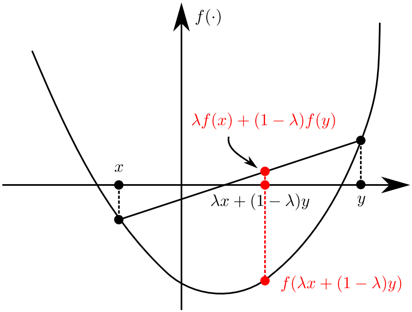
$\Rightarrow$ Let $x, y \in \mathcal{X}$. Then $(x, f(x)), (y, f(y)) \in \mathrm{epi}(f)$.

Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0, 1]$,

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



Proof.

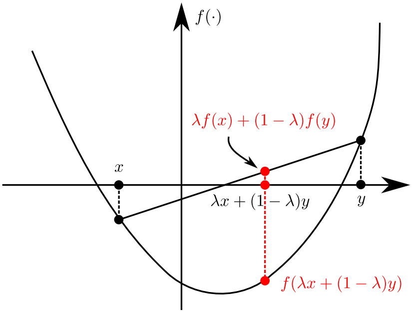
$\Rightarrow$ Let $x, y \in \mathcal{X}$. Then $(x, f(x)), (y, f(y)) \in \mathrm{epi}(f)$.
Thus so is $(\lambda x + (1-\lambda)y, \lambda f(x) + (1-\lambda)f(y))$.

Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0, 1]$,

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



Proof.

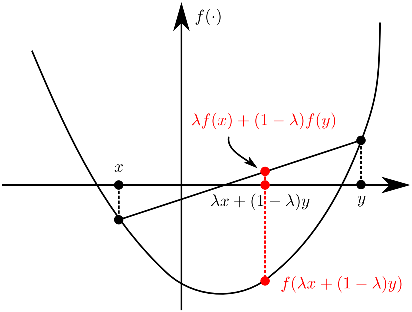
$\Rightarrow$ Let $x, y \in \mathcal{X}$. Then $(x, f(x)), (y, f(y)) \in \mathrm{epi}(f)$.
Thus so is $(\lambda x + (1-\lambda)y, \lambda f(x) + (1-\lambda)f(y))$.
By definition of $\mathrm{epi}(f)$, this implies $\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$.

Convex Function

Property (Convex function)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is convex iif, for all $x, y \in \mathcal{X}$ and $\lambda \in [0,1]$,

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



Proof.

$\Rightarrow$ Let $x, y \in \mathcal{X}$. Then $(x, f(x)), (y, f(y)) \in \mathrm{epi}(f)$.
Thus so is $(\lambda x + (1-\lambda)y, \lambda f(x) + (1-\lambda)f(y))$.
By definition of $\mathrm{epi}(f)$, this implies $\lambda f(x) + (1-\lambda)f(y) \geq f(\lambda x + (1-\lambda)y)$.

$\Leftarrow$ For $x, y \in \mathcal{X}$, $(\lambda x + (1-\lambda)y, \lambda f(x) + (1-\lambda)f(y)) \in \mathrm{epi}(f)$ and so $\mathrm{epi}(f)$ is convex.

Differentiable convex function

Reminder. For f *differentiable at* x, $\nabla f(x) = \left\{ \frac{\partial f}{\partial x_i}(x) \right\}_{i=1}^{n}$.

Differentiable convex function

Reminder. For f *differentiable at* x, $\nabla f(x) = \left\{ \frac{\partial f}{\partial x_i}(x) \right\}_{i=1}^{n}$.

Definition (Domain of a function)

The domain of $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is the set $\mathrm{dom}(f) = \{x, f(x) < +\infty\}$.

Differentiable convex function

Reminder. For f *differentiable at* x, $\nabla f(x) = \left\{ \frac{\partial f}{\partial x_i}(x) \right\}_{i=1}^{n}$.

Definition (Domain of a function)

The domain of $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is the set $\mathrm{dom}(f) = \{x, f(x) < +\infty\}$.

Theorem (First order conditions)

For $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *differentiable in its domain,* f *convex iif,* $\forall x, y \in \mathrm{dom}(f)$*,*

$$f(y) \geq f(x) + \nabla f(x)^{\mathsf{T}}(y - x).$$

Differentiable convex function

Reminder. For f *differentiable at* x, $\nabla f(x) = \left\{ \frac{\partial f}{\partial x_i}(x) \right\}_{i=1}^{n}$.

Definition (Domain of a function)

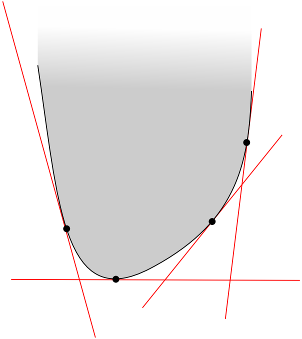
The domain of $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ is the set $\mathrm{dom}(f) = \{x, f(x) < +\infty\}$.

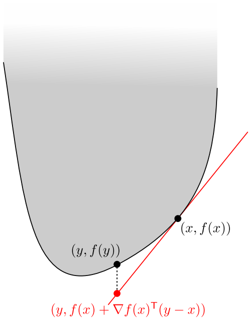
Theorem (First order conditions)

For $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *differentiable in its domain,* f *convex iif,* $\forall x, y \in \mathrm{dom}(f)$,

$$f(y) \geq f(x) + \nabla f(x)^\mathsf{T}(y - x).$$

Differentiable f: f convex iif all tangent hyperplanes of $\mathrm{epi}(f)$ are below the epigraph.





Differentiable convex function

Proof.

$\Rightarrow$ f convex implies, for $\lambda \in [0,1]$, $x, y \in \mathcal{X}$,

$$f(\lambda x + (1-\lambda)y) = f(y + \lambda(x-y)) \leq \lambda f(x) + (1-\lambda)f(y) = \lambda(f(x) - f(y)) + f(y)$$

Differentiable convex function

Proof.

$\Rightarrow$ f convex implies, for $\lambda \in [0, 1]$, $x, y \in \mathcal{X}$,

$$f(\lambda x + (1-\lambda)y) = f(y + \lambda(x-y)) \leq \lambda f(x) + (1-\lambda)f(y) = \lambda(f(x) - f(y)) + f(y)$$

or equivalently

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

Differentiable convex function

Proof.

$\Rightarrow$ f convex implies, for $\lambda \in [0,1]$, $x, y \in \mathcal{X}$,

$$f(\lambda x + (1-\lambda)y) = f(y + \lambda(x-y)) \leq \lambda f(x) + (1-\lambda)f(y) = \lambda(f(x) - f(y)) + f(y)$$

or equivalently

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

Taking the limit $y \downarrow 0$ (for this: $g(\lambda) = f(y + \lambda(x-y)) \Rightarrow g'(\lambda) = \sum_{i=1}^{n} \frac{\partial f}{\partial [x]_i} \frac{d[x]_i}{d\lambda}$),

$$\nabla f(y)^{\mathsf{T}}(x-y) \leq f(x) - f(y).$$

Differentiable convex function

Proof.

$\Rightarrow$ f convex implies, for $\lambda \in [0,1]$, $x, y \in \mathcal{X}$,

$$f(\lambda x + (1-\lambda)y) = f(y + \lambda(x-y)) \leq \lambda f(x) + (1-\lambda)f(y) = \lambda(f(x) - f(y)) + f(y)$$

or equivalently

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

Taking the limit $y \downarrow 0$ (for this: $g(\lambda) = f(y + \lambda(x-y)) \Rightarrow g'(\lambda) = \sum_{i=1}^{n} \frac{\partial f}{\partial [x]_i} \frac{d[x]_i}{d\lambda}$),

$$\nabla f(y)^{\mathsf{T}}(x-y) \leq f(x) - f(y).$$

$\Leftarrow$ For $z = \lambda x + (1-\lambda)y$,

$$\begin{aligned} (*) \quad & f(x) \geq f(z) + \nabla f(z)^{\mathsf{T}}(x-z) \\ (**) \quad & f(y) \geq f(z) + \nabla f(z)^{\mathsf{T}}(y-z). \end{aligned}$$

Differentiable convex function

Proof.

$\Rightarrow$ f convex implies, for $\lambda \in [0,1]$, $x, y \in \mathcal{X}$,

$$f(\lambda x + (1-\lambda)y) = f(y + \lambda(x-y)) \leq \lambda f(x) + (1-\lambda)f(y) = \lambda(f(x) - f(y)) + f(y)$$

or equivalently

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

Taking the limit $y \downarrow 0$ (for this: $g(\lambda) = f(y + \lambda(x-y)) \Rightarrow g'(\lambda) = \sum_{i=1}^{n} \frac{\partial f}{\partial [x]_i} \frac{d[x]_i}{d\lambda}$),

$$\nabla f(y)^{\mathsf{T}}(x-y) \leq f(x) - f(y).$$

$\Leftarrow$ For $z = \lambda x + (1-\lambda)y$,

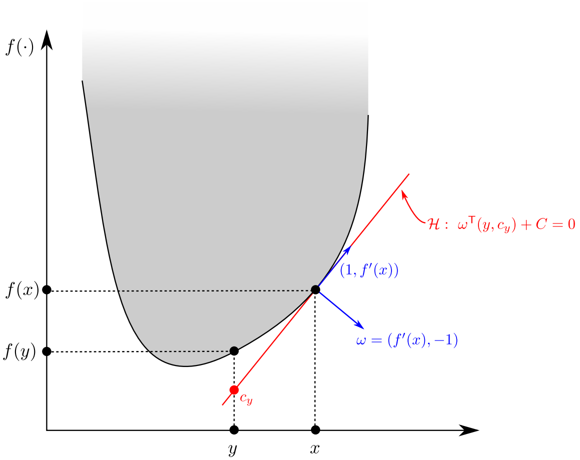
$$\begin{aligned} (*) \quad & f(x) \geq f(z) + \nabla f(z)^{\mathsf{T}}(x-z) \\ (**) \quad & f(y) \geq f(z) + \nabla f(z)^{\mathsf{T}}(y-z). \end{aligned}$$

Then $\lambda(*) + (1-\lambda)(**)$ gives

$$\lambda f(x) + (1-\lambda)f(y) \geq f(z) = f(\lambda x + (1-\lambda)y).$$

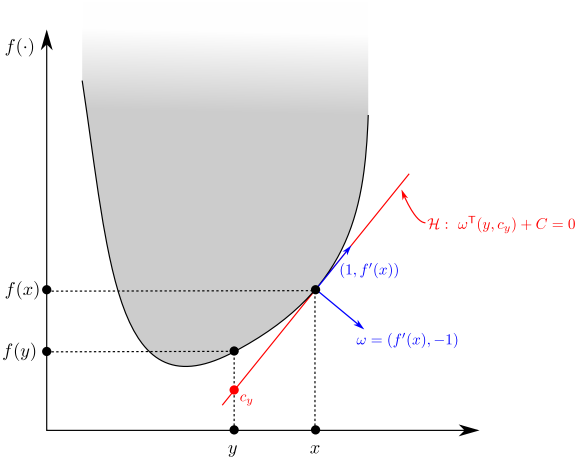
$\square$

Differentiable convex function



Detailed derivation of the first order conditions for $n = 1$:

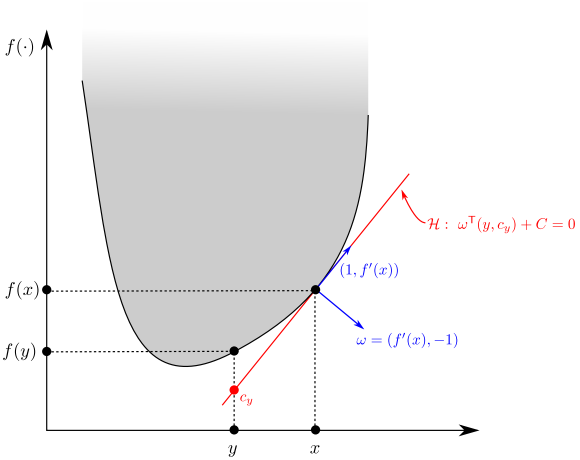
Differentiable convex function



Detailed derivation of the first order conditions for $n = 1$:

- hyperplane $\mathcal{H}$ equation given by $\omega^{\mathsf{T}}(y, c_y) + C = 0$, with $(x, f(x)) \in \mathcal{H}$

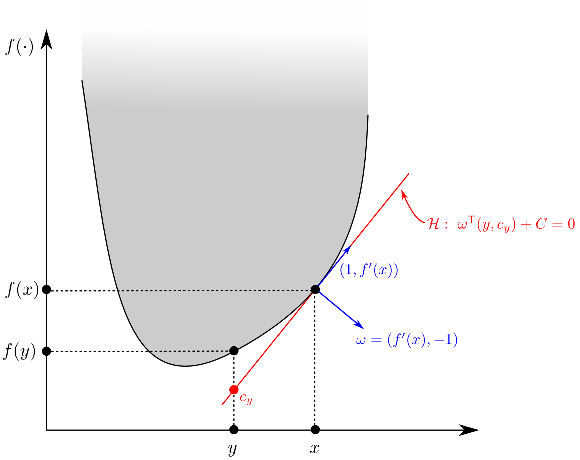
Differentiable convex function



Detailed derivation of the first order conditions for $n = 1$:

- hyperplane $\mathcal{H}$ equation given by $\omega^{\mathsf{T}}(y, c_y) + C = 0$, with $(x, f(x)) \in \mathcal{H}$
- hence $C = f(x) - f'(x)x$ (because $(f'(x), -1)^{\mathsf{T}}(x, f(x)) + C = 0$)

Differentiable convex function



Detailed derivation of the first order conditions for $n = 1$:

- hyperplane $\mathcal{H}$ equation given by $\omega^{\mathsf{T}}(y, c_y) + C = 0$, with $(x, f(x)) \in \mathcal{H}$
- hence $C = f(x) - f'(x)x$ (because $(f'(x), -1)^{\mathsf{T}}(x, f(x)) + C = 0$)
- using $c_y \leq f(y)$, one retrieves the first order conditions.

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ *minimizes* $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *convex iif* $\nabla f(x^\star) = 0$.

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ *minimizes* $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *convex iif* $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ minimizes $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex iif $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$f(x^\star + \epsilon h) = f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2)$$
$$f(x^\star - \epsilon h) = f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2).$$

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ *minimizes* $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *convex iif* $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$\begin{aligned} f(x^\star + \epsilon h) &= f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2) \\ f(x^\star - \epsilon h) &= f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2). \end{aligned}$$

If $\nabla f(x^\star)^\mathsf{T} h \neq 0$, contradiction as $\epsilon \to 0$!

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ *minimizes* $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *convex iif* $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$f(x^\star + \epsilon h) = f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2)$$
$$f(x^\star - \epsilon h) = f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2).$$

If $\nabla f(x^\star)^\mathsf{T} h \neq 0$, contradiction as $\epsilon \to 0$!
So $\nabla f(x^\star)^\mathsf{T} h = 0$.

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ minimizes $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex iif $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$
\begin{aligned}
f(x^\star + \epsilon h) &= f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2) \\
f(x^\star - \epsilon h) &= f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2).
\end{aligned}
$$

If $\nabla f(x^\star)^\mathsf{T} h \neq 0$, contradiction as $\epsilon \to 0$!
So $\nabla f(x^\star)^\mathsf{T} h = 0$.
True for all h: this implies $\nabla f(x^\star) = 0$.

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ minimizes $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex iif $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$f(x^\star + \epsilon h) = f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2)$$
$$f(x^\star - \epsilon h) = f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2).$$

If $\nabla f(x^\star)^\mathsf{T} h \neq 0$, contradiction as $\epsilon \to 0$!
So $\nabla f(x^\star)^\mathsf{T} h = 0$.
True for all h: this implies $\nabla f(x^\star) = 0$.

$\Leftarrow$ If $\nabla f(x^\star) = 0$ with f convex, $\forall x \in \mathcal{X}$,

$$f(x) \geq f(x^\star) + \nabla f(x^\star)^\mathsf{T}(x - x^\star) = f(x^\star)$$

Differentiable convex function

Important consequence: *Fermat's rule*,

Theorem (Fermat's rule)

$x^\star \in \mathcal{X}$ minimizes $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex iif $\nabla f(x^\star) = 0$.

Proof.

$\Rightarrow$ Assume $\nabla f(x^\star) \neq 0$.
Then, for $h \in \mathcal{X}$ and $\epsilon > 0$,

$$f(x^\star + \epsilon h) = f(x^\star) + \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2)$$
$$f(x^\star - \epsilon h) = f(x^\star) - \epsilon \nabla f(x^\star)^\mathsf{T} h + O(\epsilon^2).$$

If $\nabla f(x^\star)^\mathsf{T} h \neq 0$, contradiction as $\epsilon \to 0$!
So $\nabla f(x^\star)^\mathsf{T} h = 0$.
True for all h: this implies $\nabla f(x^\star) = 0$.

$\Leftarrow$ If $\nabla f(x^\star) = 0$ with f convex, $\forall x \in \mathcal{X}$,

$$f(x) \geq f(x^\star) + \nabla f(x^\star)^\mathsf{T}(x - x^\star) = f(x^\star)$$

so $x^\star$ minimizes f. $\square$

Twice-differentiable convex function

Reminder: For f twice-differentiable at x, Hessian $\nabla^2 f(x) = \{\frac{\partial^2 f}{\partial[x]_i \partial[x]_j}\}_{i,j=1}^n$.

Twice-differentiable convex function

Reminder: For f twice-differentiable at x, Hessian $\nabla^2 f(x) = \{\frac{\partial^2 f}{\partial [x]_i \partial [x]_j}\}_{i,j=1}^n$.

Theorem (Second order conditions)

For $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ twice differentiable, f is convex on its domain iif $\nabla^2 f(x)$ is semi-definite positive for all $x \in \mathrm{dom}(f)$.

Twice-differentiable convex function

Reminder: For f twice-differentiable at x, Hessian $\nabla^2 f(x) = \{\frac{\partial^2 f}{\partial[x]_i \partial[x]_j}\}_{i,j=1}^n$.

Theorem (Second order conditions)

For $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ twice differentiable, f is convex on its domain iif $\nabla^2 f(x)$ is semi-definite positive for all $x \in \mathrm{dom}(f)$.

Remark (Case $n = 1$)

For $n = 1$, $\nabla^2 f(x) = f''(x)$.

Twice-differentiable convex function

Reminder: For f twice-differentiable at x, Hessian $\nabla^2 f(x) = \{\frac{\partial^2 f}{\partial[x]_i \partial[x]_j}\}_{i,j=1}^n$.

Theorem (Second order conditions)

For $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ twice differentiable, f is convex on its domain iif $\nabla^2 f(x)$ is semi-definite positive for all $x \in \mathrm{dom}(f)$.

Remark (Case $n = 1$)

For $n = 1$, $\nabla^2 f(x) = f''(x)$. Thus, f convex iif $f''(x) > 0$ (or equivalently $f'(x)$ non-decreasing).

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon), \; f(x + \epsilon h) = f(x) + \epsilon h^\mathsf{T} \nabla f(x) + \epsilon^2 h^\mathsf{T} \nabla^2 f(x + \gamma h) h$$

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\ f(x + \epsilon h) = f(x) + \epsilon h^\top \nabla f(x) + \epsilon^2 h^\top \nabla^2 f(x + \gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x + \epsilon h)$.

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\ f(x + \epsilon h) = f(x) + \epsilon h^\mathsf{T} \nabla f(x) + \epsilon^2 h^\mathsf{T} \nabla^2 f(x + \gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x + \epsilon h)$.

But by convexity,

$$f(x + \epsilon h) \geq f(x) + \epsilon \nabla f(x)^\mathsf{T} h$$

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\ f(x + \epsilon h) = f(x) + \epsilon h^\mathsf{T} \nabla f(x) + \epsilon^2 h^\mathsf{T} \nabla^2 f(x + \gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x + \epsilon h)$.

But by convexity,

$$f(x + \epsilon h) \geq f(x) + \epsilon \nabla f(x)^\mathsf{T} h \quad \Rightarrow \quad \forall h \in \mathcal{X},\ h^\mathsf{T} \left[\nabla^2 f(x + \gamma h)\right] h \geq 0.$$

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\; f(x+\epsilon h) = f(x) + \epsilon h^{\mathsf{T}} \nabla f(x) + \epsilon^2 h^{\mathsf{T}} \nabla^2 f(x+\gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x+\epsilon h)$.

But by convexity,

$$f(x+\epsilon h) \geq f(x) + \epsilon \nabla f(x)^{\mathsf{T}} h \quad \Rightarrow \quad \forall h \in \mathcal{X},\; h^{\mathsf{T}} \left[\nabla^2 f(x+\gamma h)\right] h \geq 0.$$

With $\epsilon \downarrow 0$, we obtain $\forall h \in \mathcal{X},\; h^{\mathsf{T}} \left[\nabla^2 f(x)\right] h \geq 0$, i.e., $\nabla^2 f \succeq 0$.

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0,\, \epsilon),\; f(x+\epsilon h) = f(x) + \epsilon h^\mathsf{T} \nabla f(x) + \epsilon^2 h^\mathsf{T} \nabla^2 f(x+\gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x+\epsilon h)$.

But by convexity,

$$f(x+\epsilon h) \geq f(x) + \epsilon \nabla f(x)^\mathsf{T} h \quad \Rightarrow \quad \forall h \in \mathcal{X},\; h^\mathsf{T} \left[\nabla^2 f(x+\gamma h)\right] h \geq 0.$$

With $\epsilon \downarrow 0$, we obtain $\forall h \in \mathcal{X},\; h^\mathsf{T} \left[\nabla^2 f(x)\right] h \geq 0$, i.e., $\nabla^2 f \succeq 0$.

$\Leftarrow$ Define $g : [0,1] \to \mathbb{R} \cup \{+\infty\}$, $g(t) = f(tx + (1-t)y)$.

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0,\, \epsilon),\; f(x+\epsilon h) = f(x) + \epsilon h^\top \nabla f(x) + \epsilon^2 h^\top \nabla^2 f(x+\gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x+\epsilon h)$.
But by convexity,

$$f(x+\epsilon h) \geq f(x) + \epsilon \nabla f(x)^\top h \quad \Rightarrow \quad \forall h \in \mathcal{X},\; h^\top \left[\nabla^2 f(x+\gamma h)\right] h \geq 0.$$

With $\epsilon \downarrow 0$, we obtain $\forall h \in \mathcal{X},\; h^\top \left[\nabla^2 f(x)\right] h \geq 0$, i.e., $\nabla^2 f \succeq 0$.

$\Leftarrow$ Define $g : [0,1] \to \mathbb{R} \cup \{+\infty\}$, $g(t) = f(tx + (1-t)y)$.
By chain rule ($g'(t) = \sum_{i=1}^n \frac{\partial f}{\partial [z]_i} \frac{d[z]_i(t)}{dt}$ with $g(t) \equiv f(z(t))$, and similarly for $g''(t)$)

$$g''(t) = (x-y)^\top \left[\nabla^2 f(tx + (1-t)y)\right] (x-y) \geq 0 \quad (\text{since } \nabla^2 f \succeq 0).$$

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\ f(x+\epsilon h) = f(x) + \epsilon h^\top \nabla f(x) + \epsilon^2 h^\top \nabla^2 f(x+\gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x+\epsilon h)$.
But by convexity,

$$f(x+\epsilon h) \geq f(x) + \epsilon \nabla f(x)^\top h \quad \Rightarrow \quad \forall h \in \mathcal{X},\ h^\top \left[\nabla^2 f(x+\gamma h)\right] h \geq 0.$$

With $\epsilon \downarrow 0$, we obtain $\forall h \in \mathcal{X}$, $h^\top \left[\nabla^2 f(x)\right] h \geq 0$, i.e., $\nabla^2 f \succeq 0$.

$\Leftarrow$ Define $g : [0,1] \to \mathbb{R} \cup \{+\infty\}$, $g(t) = f(tx + (1-t)y)$.
By chain rule ($g'(t) = \sum_{i=1}^n \frac{\partial f}{\partial [z]_i} \frac{d[z]_i(t)}{dt}$ with $g(t) \equiv f(z(t))$, and similarly for $g''(t)$)

$$g''(t) = (x-y)^\top \left[\nabla^2 f(tx + (1-t)y)\right] (x-y) \geq 0 \quad (\text{since } \nabla^2 f \succeq 0).$$

By Taylor-Lagrange, we then have, for some $\zeta_x, \zeta_y \in [0,1]$,

$$(*)\ f(y) = g(0) = g(t) + (0-t)g'(t) + \frac{1}{2}t^2 g''(\zeta_y) \geq g(t) - tg'(t)$$

$$(**)\ f(x) = g(1) = g(t) + (1-t)g'(t) + \frac{1}{2}t^2 g''(\zeta_x) \geq g(t) + (1-t)g'(t).$$

Twice-differentiable convex function

Proof.

$\Rightarrow$ By Taylor-Lagrange, $\forall h \in \mathcal{X}$ and $\forall \epsilon > 0$,

$$\exists \gamma \in (0, \epsilon),\ f(x + \epsilon h) = f(x) + \epsilon h^{\mathsf{T}} \nabla f(x) + \epsilon^2 h^{\mathsf{T}} \nabla^2 f(x + \gamma h) h$$

Why? 1D Taylor-Lagrange by differentiating $g : \epsilon \mapsto f(x + \epsilon h)$.
But by convexity,

$$f(x + \epsilon h) \geq f(x) + \epsilon \nabla f(x)^{\mathsf{T}} h \quad \Rightarrow \quad \forall h \in \mathcal{X},\ h^{\mathsf{T}} \left[\nabla^2 f(x + \gamma h)\right] h \geq 0.$$

With $\epsilon \downarrow 0$, we obtain $\forall h \in \mathcal{X}$, $h^{\mathsf{T}} \left[\nabla^2 f(x)\right] h \geq 0$, i.e., $\nabla^2 f \succeq 0$.

$\Leftarrow$ Define $g : [0, 1] \to \mathbb{R} \cup \{+\infty\}$, $g(t) = f(tx + (1 - t)y)$.
By chain rule $\left(g'(t) = \sum_{i=1}^{n} \frac{\partial f}{\partial [z]_i} \frac{d[z]_i(t)}{dt}\right.$ with $g(t) \equiv f(z(t))$, and similarly for $\left.g''(t)\right)$

$$g''(t) = (x - y)^{\mathsf{T}} \left[\nabla^2 f(tx + (1 - t)y)\right] (x - y) \geq 0 \quad (\text{since } \nabla^2 f \succeq 0).$$

By Taylor-Lagrange, we then have, for some $\zeta_x, \zeta_y \in [0, 1]$,

$$(*)\ f(y) = g(0) = g(t) + (0 - t)g'(t) + \frac{1}{2} t^2 g''(\zeta_y) \geq g(t) - t g'(t)$$

$$(**)\ f(x) = g(1) = g(t) + (1 - t)g'(t) + \frac{1}{2} t^2 g''(\zeta_x) \geq g(t) + (1 - t) g'(t).$$

Using $(1 - t)(*) + t(**)$, we conclude $t f(x) + (1 - t) f(y) \geq g(t) = f(tx + (1 - t)y)$.

Outline

Outline

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Reminder: our objective is to solve

$$x^\star \in \mathrm{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \dots$ with x_{k+1} a function of x_k.

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x\in\Omega\subset\mathcal{X}}\{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \mathrm{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

- $\|x_{k+1} - x_k\| < \epsilon$: the algorithm no longer progresses in $\mathcal{X}$;

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

- $\|x_{k+1} - x_k\| < \epsilon$: the algorithm no longer progresses in $\mathcal{X}$;
- $|f(x_{k+1}) - f(x_k)| < \epsilon$: the *cost* no longer progresses

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

- $\|x_{k+1} - x_k\| < \epsilon$: the algorithm no longer progresses in $\mathcal{X}$;
- $|f(x_{k+1}) - f(x_k)| < \epsilon$: the *cost* no longer progresses ($\not\Rightarrow$ x_k converges!);

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x\in\Omega\subset\mathcal{X}}\{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

- $\|x_{k+1} - x_k\| < \epsilon$: the algorithm no longer progresses in $\mathcal{X}$;
- $|f(x_{k+1}) - f(x_k)| < \epsilon$: the *cost* no longer progresses ($\not\Rightarrow$ x_k converges!);
- $\|\nabla f(x_k)\| < \epsilon$: cost almost flat

Convex optimization algorithms: the unconstrained differentiable case

Reminder: our objective is to solve

$$x^\star \in \operatorname{argmin}_{x \in \Omega \subset \mathcal{X}} \{f(x)\}.$$

Assumption (Unconstrained Ω, differentiable f)

- *f differentiable everywhere on $\mathcal{X}$;*
- *Ω unbounded.*

Definition (Iterative algorithms)

Sequentially evaluate f at positions $x_1, x_2, \ldots$ with x_{k+1} a function of x_k.
Algorithm terminates when either:

- $\|x_{k+1} - x_k\| < \epsilon$: the algorithm no longer progresses in $\mathcal{X}$;
- $|f(x_{k+1}) - f(x_k)| < \epsilon$: the *cost* no longer progresses ($\not\Rightarrow x_k$ converges!);
- $\|\nabla f(x_k)\| < \epsilon$: cost almost flat (close to $\nabla f(x^*) = 0$ but maybe far from $x^\star$).

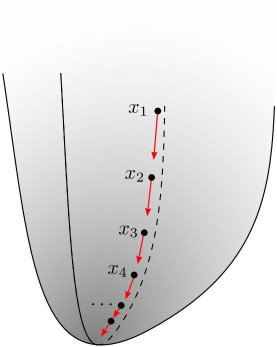
Convex optimization algorithms: descent methods

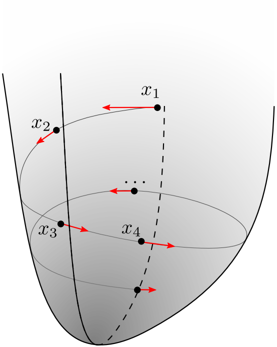
Definition (Descent Method)

Descent method is an algorithm outputing $x_1, x_2, \ldots \in \mathcal{X}$ of the form

$$x_{k+1} = x_k + t_k \Delta x_k, \quad \textit{step size} \quad t_k > 0, \quad \textit{increment} \quad \Delta x_k$$

such that $f(x_{k+1}) < f(x_k)$ if $x_k \notin \operatorname{argmin} f$ and $f(x_{k+1}) = f(x_k)$ if $x_k \in \operatorname{argmin} f$.





Remark (Convergence (or not) of descent algorithms)

For f with non-empty set of minima, descent algorithms converge, however not necessarily to local minimum:

Convex optimization algorithms: descent methods

Remark (Convergence (or not) of descent algorithms)

For f with non-empty set of minima, descent algorithms converge, however not necessarily to local minimum:

- *too slow descent: we may have* $\lim_k f(x_k) > f(x^\star)$;

Convex optimization algorithms: descent methods

Remark (Convergence (or not) of descent algorithms)

For f with non-empty set of minima, descent algorithms converge, however not necessarily to local minimum:

- *too slow descent: we may have* $\lim_k f(x_k) > f(x^\star)$*;*
- $f(x_k) \to f(x^\star)$ *does not imply that* x_k *converges at all (periodic behavior of* x_k*!).*

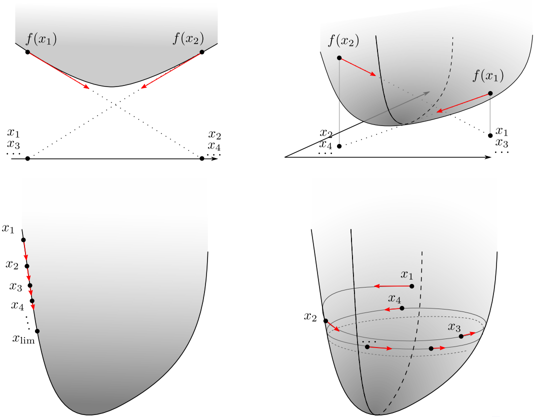
Convex optimization algorithms: descent methods

Remark (Convergence (or not) of descent algorithms)

For f with non-empty set of minima, descent algorithms converge, however not necessarily to local minimum:

- *too slow descent: we may have* $\lim_k f(x_k) > f(x^\star)$*;*
- $f(x_k) \to f(x^\star)$ *does not imply that* x_k *converges at all (periodic behavior of* x_k*!).*

Descent sequences either **not converging** (top) or **not reaching minimum** (bottom).



Convex optimization algorithms: descent methods

Important property: for $x_k, x_{k+1} \in \mathcal{X}$, by first order condition

$$f(x_k + t_k \Delta x_k) \geq f(x_k) + t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k.$$

Convex optimization algorithms: descent methods

Important property: for $x_k, x_{k+1} \in \mathcal{X}$, by first order condition

$$f(x_k + t_k \Delta x_k) \geq f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

As such, letting $x_1, x_2, \ldots$ defined by

$$x_{k+1} = x_k + t_k \Delta x_k,$$

we have

$$f(x_{k+1}) \geq f(x_k) + \nabla f(x_k)^\mathsf{T} (x_{k+1} - x_k) = f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Convex optimization algorithms: descent methods

Important property: for $x_k, x_{k+1} \in \mathcal{X}$, by first order condition

$$f(x_k + t_k \Delta x_k) \geq f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

As such, letting $x_1, x_2, \ldots$ defined by

$$x_{k+1} = x_k + t_k \Delta x_k,$$

we have

$$f(x_{k+1}) \geq f(x_k) + \nabla f(x_k)^\mathsf{T}(x_{k+1} - x_k) = f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

and thus $x_1, x_2, \ldots$ cannot be a descent method sequence unless $\nabla f(x_k)^\mathsf{T} \Delta x_k \leq 0$.

Convex optimization algorithms: descent methods

Important property: for $x_k, x_{k+1} \in \mathcal{X}$, by first order condition

$$f(x_k + t_k \Delta x_k) \geq f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

As such, letting $x_1, x_2, \ldots$ defined by

$$x_{k+1} = x_k + t_k \Delta x_k,$$

we have

$$f(x_{k+1}) \geq f(x_k) + \nabla f(x_k)^\mathsf{T}(x_{k+1} - x_k) = f(x_k) + t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

and thus $x_1, x_2, \ldots$ cannot be a descent method sequence unless $\nabla f(x_k)^\mathsf{T} \Delta x_k \leq 0$.

Property (Descent direction)

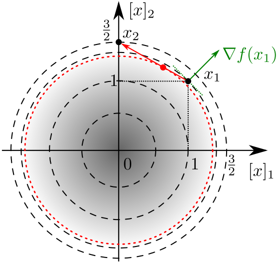
Necessary condition for $x_1, x_2, \ldots$ to be a descent sequence,

$$\nabla f(x_k)^\mathsf{T} \Delta x_k \leq 0$$

where $\Delta x_k = x_{k+1} - x_k$, and equality reached iif $x_k \in \arg\min f$.

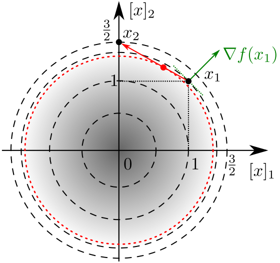
Convex optimization algorithms: descent methods

The condition is not sufficient!



Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

The condition is not sufficient!



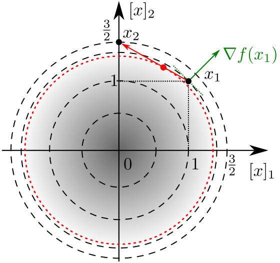
Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

Although $\Delta x_1 = [-1,\ 1/2]$ has acute angle with $-\nabla f(x_1)$,

$$x_2 = [0, 3/2] = x_1 + \Delta x_1$$

increases rather than decreases f.

The condition is not sufficient!



Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

Although $\Delta x_1 = [-1, 1/2]$ has acute angle with $-\nabla f(x_1)$,

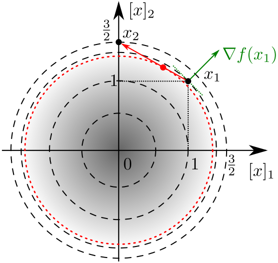
$$x_2 = [0, 3/2] = x_1 + \Delta x_1$$

increases rather than decreases f.

Yet, for small t, $x_1 + t\Delta x_1$ is descent direction (red circle).

Convex optimization algorithms: descent methods

The condition is not sufficient!



Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

Although $\Delta x_1 = [-1,\ 1/2]$ has acute angle with $-\nabla f(x_1)$,

$$x_2 = [0, 3/2] = x_1 + \Delta x_1$$

increases rather than decreases f.

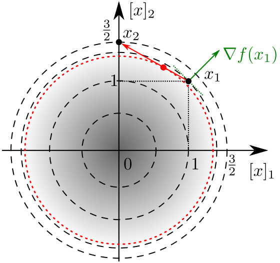
Yet, for small t, $x_1 + t\Delta x_1$ is descent direction (red circle).

The condition is "locally sufficient" with **small steps** *and* f locally twice-differentiable; indeed, by Taylor

$$f(x_{k+1}) = f(x_k) + t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k + O(t_k^2 \|\Delta x_k\|^2)$$

Convex optimization algorithms: descent methods

The condition is not sufficient!



Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

Although $\Delta x_1 = [-1, 1/2]$ has acute angle with $-\nabla f(x_1)$,

$$x_2 = [0, 3/2] = x_1 + \Delta x_1$$

increases rather than decreases f.

Yet, for small t, $x_1 + t\Delta x_1$ is descent direction (red circle).

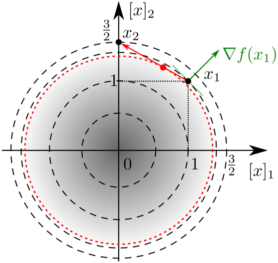
The condition is "locally sufficient" with **small steps** *and* f locally twice-differentiable; indeed, by Taylor

$$f(x_{k+1}) = f(x_k) + t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k + O(t_k^2 \|\Delta x_k\|^2)$$

so that, $\forall t_k > 0$ **small**, $f(x_{k+1}) < f(x_k)$.

Convex optimization algorithms: descent methods

The condition is not sufficient!



Function $f(x) = [x]_1^2 + [x]_2^2$.
Initialized at $x_1 = [1, 1]$.

Although $\Delta x_1 = [-1, 1/2]$ has acute angle with $-\nabla f(x_1)$,

$$x_2 = [0, 3/2] = x_1 + \Delta x_1$$

increases rather than decreases f.

Yet, for small t, $x_1 + t\Delta x_1$ is descent direction (red circle).

The condition is "locally sufficient" with **small steps** *and* f locally twice-differentiable; indeed, by Taylor

$$f(x_{k+1}) = f(x_k) + t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k + O(t_k^2 \|\Delta x_k\|^2)$$

so that, $\forall t_k > 0$ **small**, $f(x_{k+1}) < f(x_k)$.

$\Rightarrow$ **Careful control of step sizes needed!**

Convex optimization algorithms: descent methods

Remark: Still in small step size limit, gain $|f(x_{k+1}) - f(x_k)|$ maximal when $\nabla f(x_k)^\mathsf{T} \Delta x_k$ **both negative and of maximal absolute value**.

Convex optimization algorithms: descent methods

Remark: Still in small step size limit, gain $|f(x_{k+1}) - f(x_k)|$ maximal when $\nabla f(x_k)^{\mathsf{T}}\Delta x_k$ **both negative and of maximal absolute value**.

For $\|\Delta x_k\| = 1$, optimal when

$$\Delta x_k = -\frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}.$$

Remark: Still in small step size limit, gain $|f(x_{k+1}) - f(x_k)|$ maximal when $\nabla f(x_k)^{\mathsf{T}} \Delta x_k$ **both negative and of maximal absolute value**.

For $\|\Delta x_k\| = 1$, optimal when

$$\Delta x_k = -\frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}.$$

Leads to popular *gradient descent* algorithm.

Definition (Gradient Descent Algorithm)

$x_1 \in \mathcal{X}$ and, for all $k \geq 1$,

$$x_{k+1} = x_k - t_k \nabla f(x_k), \quad t_1, t_2, \ldots > 0.$$

Remark: Still in small step size limit, gain $|f(x_{k+1}) - f(x_k)|$ maximal when $\nabla f(x_k)^{\mathsf{T}} \Delta x_k$ **both negative and of maximal absolute value**.

For $\|\Delta x_k\| = 1$, optimal when

$$\Delta x_k = -\frac{\nabla f(x_k)}{\|\nabla f(x_k)\|}.$$

Leads to popular *gradient descent* algorithm.

Definition (Gradient Descent Algorithm)

$x_1 \in \mathcal{X}$ and, for all $k \geq 1$,

$$x_{k+1} = x_k - t_k \nabla f(x_k), \quad t_1, t_2, \ldots > 0.$$

Remark: Often, *constant step*, i.e., $t_k = t$ constant:

- easy: does not request fine-tuning of t_k,
- but suboptimal.

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^{\mathsf{T}} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k.$$

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^{\mathsf{T}} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^{\mathsf{T}} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^\mathsf{T} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Remark (On step size)

- **[Line search]**

$$t_k \in \mathrm{argmin}_{t>0} f(x_k + t\Delta x_k)$$

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^\mathsf{T} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Remark (On step size)

- **[Line search]**

$$t_k \in \operatorname{argmin}_{t>0} f(x_k + t\Delta x_k)$$

But can be expensive (second optimization or full line search).

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^\mathsf{T} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Remark (On step size)

- **[Line search]**

$$t_k \in \operatorname{argmin}_{t>0} f(x_k + t\Delta x_k)$$

But can be expensive (second optimization or full line search).

- **[Backtracking]** *simplified line search:* $t^{(0)} = 1$ *and, for some* $0 < \alpha, \beta < 1$, $t^{(j+1)} = \beta t^{(j)}$ *until*

$$f(x_k + t^{(j+1)} \Delta x_k) < f(x_k) + \alpha t^{(j+1)} \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^\mathsf{T} \Delta x_k < 0$, and $\alpha \in (0,1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Remark (On step size)

- **[Line search]**

$$t_k \in \mathrm{argmin}_{t>0}\, f(x_k + t\Delta x_k)$$

  But *can be expensive* *(second optimization or full line search).*

- **[Backtracking]** *simplified line search:* $t^{(0)} = 1$ *and, for some* $0 < \alpha, \beta < 1$, $t^{(j+1)} = \beta t^{(j)}$ *until*

$$f(x_k + t^{(j+1)} \Delta x_k) < f(x_k) + \alpha t^{(j+1)} \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

  Remark: *meets Armijo-Goldstein condition!*

Convex optimization algorithms: descent methods

Definition (Armijo-Goldstein condition)

Given Δx_k with $\|\Delta x_k\| = 1$ and $\nabla f(x_k)^\mathsf{T} \Delta x_k < 0$, and $\alpha \in (0, 1)$, t_k satisfies Armijo-Goldstein condition if

$$f(x_k + t_k \Delta x_k) < f(x_k) + \alpha t_k \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Remark: a descent sequence $x_1, x_2, \ldots$.

Remark (On step size)

- **[Line search]**

  $$t_k \in \operatorname{argmin}_{t>0} f(x_k + t\Delta x_k)$$

  But *can be expensive* *(second optimization or full line search).*

- **[Backtracking]** *simplified line search:* $t^{(0)} = 1$ *and, for some* $0 < \alpha, \beta < 1$, $t^{(j+1)} = \beta t^{(j)}$ *until*

  $$f(x_k + t^{(j+1)} \Delta x_k) < f(x_k) + \alpha t^{(j+1)} \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

  Remark: *meets Armijo-Goldstein condition!*
  Always achievable: as $t^{(j)} \to 0$,

  $$f(x_k + t^{(j+1)} \Delta x_k) \simeq f(x_k) + t^{(j)} \nabla f(x_k)^\mathsf{T} \Delta x_k < f(x_k) + \alpha t^{(j+1)} \nabla f(x_k)^\mathsf{T} \Delta x_k.$$

Convex optimization algorithms: convergence of gradient descent

Theorem (Convergence of Gradient Descent with Constant Step Size)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ *convex, twice continuously differentiable, with L-Lipschitz* ∇f*:*

$$\forall x, y \in \mathcal{X} \qquad \|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|.$$

Then gradient descent with constant step size $t \leq \frac{1}{L}$ *convergences to a minimum of* f*:*

$$x_k \to x^\star \in \operatorname{argmin}_x f(x).$$

Proof.

1. Prelim. Lipschitz condition on ∇f implies $\nabla^2 f(x) \preceq L I_n$:

Proof.

1. Prelim. Lipschitz condition on ∇f implies $\nabla^2 f(x) \preceq LI_n$: for $x, u \in \mathcal{X}$

$$f(x+\epsilon u) = f(x) + \epsilon \nabla f(x)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x) u + o(\epsilon^2)$$

$$f(x) = f(x+\epsilon u) - \epsilon \nabla f(x+\epsilon u)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x+\epsilon u) u + o(\epsilon^2).$$

Convex optimization algorithms: convergence of gradient descent

Proof.

1. Prelim. Lipschitz condition on ∇f implies $\nabla^2 f(x) \preceq LI_n$: for $x, u \in \mathcal{X}$

$$f(x+\epsilon u) = f(x) + \epsilon \nabla f(x)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x) u + o(\epsilon^2)$$

$$f(x) = f(x+\epsilon u) - \epsilon \nabla f(x+\epsilon u)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x+\epsilon u) u + o(\epsilon^2).$$

Summing and dividing by ϵ^2:

$$\frac{(\nabla f(x+\epsilon u) - \nabla f(x))^\mathsf{T} u}{\epsilon} = \frac{1}{2} u^\mathsf{T} (\nabla^2 f(x) + \nabla^2 f(x+\epsilon u)) u + o(1).$$

Convex optimization algorithms: convergence of gradient descent

Proof.

1. Prelim. Lipschitz condition on ∇f implies $\nabla^2 f(x) \preceq L I_n$: for $x, u \in \mathcal{X}$

$$f(x + \epsilon u) = f(x) + \epsilon \nabla f(x)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x) u + o(\epsilon^2)$$

$$f(x) = f(x + \epsilon u) - \epsilon \nabla f(x + \epsilon u)^\mathsf{T} u + \frac{1}{2}\epsilon^2 u^\mathsf{T} \nabla^2 f(x + \epsilon u) u + o(\epsilon^2).$$

Summing and dividing by ϵ^2:

$$\frac{(\nabla f(x + \epsilon u) - \nabla f(x))^\mathsf{T} u}{\epsilon} = \frac{1}{2} u^\mathsf{T} (\nabla^2 f(x) + \nabla^2 f(x + \epsilon u)) u + o(1).$$

By Cauchy-Schwarz and the Lipschitz condition,

$$\frac{1}{2} u^\mathsf{T} (\nabla^2 f(x) + \nabla^2 f(x + \epsilon u)) u + o(1) \le \frac{\|\nabla f(x + \epsilon u) - \nabla f(x)\| \|u\|}{\epsilon} \le L \|u\|^2.$$

Convex optimization algorithms: convergence of gradient descent

Proof.

1. Prelim. Lipschitz condition on ∇f implies $\nabla^2 f(x) \preceq LI_n$: for $x, u \in \mathcal{X}$

$$f(x+\epsilon u) = f(x) + \epsilon \nabla f(x)^{\mathsf{T}} u + \frac{1}{2}\epsilon^2 u^{\mathsf{T}} \nabla^2 f(x) u + o(\epsilon^2)$$

$$f(x) = f(x+\epsilon u) - \epsilon \nabla f(x+\epsilon u)^{\mathsf{T}} u + \frac{1}{2}\epsilon^2 u^{\mathsf{T}} \nabla^2 f(x+\epsilon u) u + o(\epsilon^2).$$

Summing and dividing by ϵ^2:

$$\frac{(\nabla f(x+\epsilon u) - \nabla f(x))^{\mathsf{T}} u}{\epsilon} = \frac{1}{2} u^{\mathsf{T}} (\nabla^2 f(x) + \nabla^2 f(x+\epsilon u)) u + o(1).$$

By Cauchy-Schwarz and the Lipschitz condition,

$$\frac{1}{2} u^{\mathsf{T}} (\nabla^2 f(x) + \nabla^2 f(x+\epsilon u)) u + o(1) \le \frac{\|\nabla f(x+\epsilon u) - \nabla f(x)\| \|u\|}{\epsilon} \le L\|u\|^2.$$

So, as $\epsilon \to 0$,

$$u^{\mathsf{T}} \nabla^2 f(x) u \le L\|u\|^2, \quad \forall u \in \mathcal{X}.$$

$\square$

Convex optimization algorithms: convergence of gradient descent

Proof.

2. Core of Proof. Since f convex $(*)$ and $\nabla^2 f(x) \preceq LI_n$ $(**)$, for $x, y \in \mathcal{X}$,

$$(*)\ f(y) \geq f(x) + \nabla f(x)^{\mathsf{T}}(y-x)$$

$$\begin{aligned}(**)\ f(y) &= f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}(y-x)^{\mathsf{T}}\nabla^2 f(\zeta)(y-x) \\ &\leq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}L\|y-x\|^2\end{aligned}$$

($\zeta = x + \lambda(y-x)$ for some $\lambda \in [0,1]$).

Convex optimization algorithms: convergence of gradient descent

Proof.

2. Core of Proof. Since f convex $(*)$ and $\nabla^2 f(x) \preceq LI_n$ $(**)$, for $x, y \in \mathcal{X}$,

$$
\begin{aligned}
(*)\ f(y) &\geq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) \\
(**)\ f(y) &= f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}(y-x)^{\mathsf{T}}\nabla^2 f(\zeta)(y-x) \\
&\leq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}L\|y-x\|^2
\end{aligned}
$$

($\zeta = x + \lambda(y-x)$ for some $\lambda \in [0,1]$).
From $(**)$,

$$
\begin{aligned}
f(x_{k+1}) &\leq f(x_k) + \nabla f(x_k)^{\mathsf{T}}(x_{k+1}-x_k) + \frac{1}{2}L\|x_{k+1}-x_k\|^2 \\
&= f(x_k) - t\|\nabla f(x_k)\|^2 + \frac{1}{2}Lt^2\|\nabla f(x_k)\|^2 \\
&= f(x_k) - \left(1 - \frac{1}{2}Lt\right) t\|\nabla f(x_k)\|^2.
\end{aligned}
$$

Proof.

2. Core of Proof. Since f convex $(*)$ and $\nabla^2 f(x) \preceq LI_n$ $(**)$, for $x, y \in \mathcal{X}$,

$$
\begin{aligned}
(*)\ f(y) &\geq f(x) + \nabla f(x)^{\mathsf{T}}(y - x) \\
(**)\ f(y) &= f(x) + \nabla f(x)^{\mathsf{T}}(y - x) + \frac{1}{2}(y - x)^{\mathsf{T}}\nabla^2 f(\zeta)(y - x) \\
&\leq f(x) + \nabla f(x)^{\mathsf{T}}(y - x) + \frac{1}{2}L\|y - x\|^2
\end{aligned}
$$

($\zeta = x + \lambda(y - x)$ for some $\lambda \in [0, 1]$).
From $(**)$,

$$
\begin{aligned}
f(x_{k+1}) &\leq f(x_k) + \nabla f(x_k)^{\mathsf{T}}(x_{k+1} - x_k) + \frac{1}{2}L\|x_{k+1} - x_k\|^2 \\
&= f(x_k) - t\|\nabla f(x_k)\|^2 + \frac{1}{2}Lt^2\|\nabla f(x_k)\|^2 \\
&= f(x_k) - \left(1 - \frac{1}{2}Lt\right) t\|\nabla f(x_k)\|^2.
\end{aligned}
$$

We now use $t \leq 1/L$:

$$
f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 (\leq f(x_k)) \tag{2}
$$

with equality iif $\nabla f(x_k) = 0$

Convex optimization algorithms: convergence of gradient descent

Proof.

2. Core of Proof. Since f convex $(*)$ and $\nabla^2 f(x) \preceq LI_n$ $(**)$, for $x, y \in \mathcal{X}$,

$$(*)\ f(y) \geq f(x) + \nabla f(x)^{\mathsf{T}}(y-x)$$

$$\begin{aligned}(**)\ f(y) &= f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}(y-x)^{\mathsf{T}}\nabla^2 f(\zeta)(y-x) \\ &\leq f(x) + \nabla f(x)^{\mathsf{T}}(y-x) + \frac{1}{2}L\|y-x\|^2\end{aligned}$$

($\zeta = x + \lambda(y-x)$ for some $\lambda \in [0,1]$).
From $(**)$,

$$\begin{aligned}f(x_{k+1}) &\leq f(x_k) + \nabla f(x_k)^{\mathsf{T}}(x_{k+1}-x_k) + \frac{1}{2}L\|x_{k+1}-x_k\|^2 \\ &= f(x_k) - t\|\nabla f(x_k)\|^2 + \frac{1}{2}Lt^2\|\nabla f(x_k)\|^2 \\ &= f(x_k) - \left(1 - \frac{1}{2}Lt\right) t\|\nabla f(x_k)\|^2.\end{aligned}$$

We now use $t \leq 1/L$:

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 (\leq f(x_k)) \qquad (2)$$

with equality iif $\nabla f(x_k) = 0 \Rightarrow$ gradient descent *is* a descent algorithm.

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x)$$

Convex optimization algorithms: convergence of gradient descent

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

Convex optimization algorithms: convergence of gradient descent

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

So in particular, from (2), $(f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2)$

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2$$

Convex optimization algorithms: convergence of gradient descent

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

So in particular, from (2), $\left(f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2\right)$

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 \leq f(x^\star) + \nabla f(x_k)^\mathsf{T}(x_k - x^\star) - \frac{t}{2}\|\nabla f(x_k)\|^2.$$

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

So in particular, from (2), $\left(f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2\right)$

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 \leq f(x^\star) + \nabla f(x_k)^\mathsf{T}(x_k - x^\star) - \frac{t}{2}\|\nabla f(x_k)\|^2.$$

We need to relate $\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$ to $t\|\nabla f(x_k)\|^2$:

$$\|x_k - x^\star - t\nabla f(x_k)\|^2 = \|x_k - x^\star\|^2 + t^2\|\nabla f(x_k)\|^2 - 2t\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$$

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

So in particular, from (2), $\left(f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2\right)$

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 \leq f(x^\star) + \nabla f(x_k)^\mathsf{T}(x_k - x^\star) - \frac{t}{2}\|\nabla f(x_k)\|^2.$$

We need to relate $\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$ to $t\|\nabla f(x_k)\|^2$:

$$\|x_k - x^\star - t\nabla f(x_k)\|^2 = \|x_k - x^\star\|^2 + t^2\|\nabla f(x_k)\|^2 - 2t\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$$

which yields

$$f(x_{k+1}) \leq f(x^\star) + \frac{1}{2t}\left(\|x_k - x^\star\|^2 - \|\underbrace{x_k - t\nabla f(x_k)}_{x_{k+1}} - x^\star\|^2\right).$$

Convex optimization algorithms: convergence of gradient descent

Proof.

3. Convergence to minimum. From $(*)$, for any $x^\star \in \operatorname{argmin} f$ and $x \in \mathcal{X}$,

$$f(x^\star) \geq f(x) + \nabla f(x)^\mathsf{T}(x^\star - x) \quad \Leftrightarrow \quad f(x) \leq f(x^\star) + \nabla f(x)^\mathsf{T}(x - x^\star).$$

So in particular, from (2), $\left(f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2\right)$

$$f(x_{k+1}) \leq f(x_k) - \frac{t}{2}\|\nabla f(x_k)\|^2 \leq f(x^\star) + \nabla f(x_k)^\mathsf{T}(x_k - x^\star) - \frac{t}{2}\|\nabla f(x_k)\|^2.$$

We need to relate $\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$ to $t\|\nabla f(x_k)\|^2$:

$$\|x_k - x^\star - t\nabla f(x_k)\|^2 = \|x_k - x^\star\|^2 + t^2\|\nabla f(x_k)\|^2 - 2t\nabla f(x_k)^\mathsf{T}(x_k - x^\star)$$

which yields

$$f(x_{k+1}) \leq f(x^\star) + \frac{1}{2t}\left(\|x_k - x^\star\|^2 - \|\underbrace{x_k - t\nabla f(x_k)}_{x_{k+1}} - x^\star\|^2\right).$$

Summing for $k = 1, \ldots, K$, RHS *telescopes*:

$$\underbrace{\sum_{k=1}^{K} f(x_{k+1}) - f(x^\star)}_{\geq K(f(x_K) - f(x^\star))} \leq \frac{1}{2t}\left(\|x_1 - x^\star\|^2 - \|x_K - x^\star\|^2\right) \leq \frac{1}{2t}\|x_1 - x^\star\|^2.$$

Proof.

So finally, as $K \to \infty$

$$f(x_K) - f(x^\star) \leq \frac{1}{2Kt}\|x_1 - x^\star\|^2 \to 0$$

Convex optimization algorithms: convergence of gradient descent

Proof.
So finally, as $K \to \infty$

$$f(x_K) - f(x^\star) \leq \frac{1}{2Kt}\|x_1 - x^\star\|^2 \to 0$$

x_K may not converge, but $f(x_K) \to f(x^\star)$. $\square$

Remark (Advantages/limitations of gradient descent)

- *simple to implement*: *for f not easily differentiable, gradient approximation* $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ *with* $[e_i]_j = \delta_i^j$ *i-th canonical vector;*

Convex optimization algorithms: convergence of gradient descent

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j = \delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k+\epsilon e_i)-f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j=\delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*
- *ensured convergence for fixed steps: "no" step size adaptation required;*

Convex optimization algorithms: convergence of gradient descent

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j = \delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*
- *ensured convergence for fixed steps: "no" step size adaptation required;*
- **BUT** *requires small steps ($< 1/L$): in most cases, difficult to evaluate;*

Convex optimization algorithms: convergence of gradient descent

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j = \delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*
- *ensured convergence for fixed steps: "no" step size adaptation required;*
- **BUT** *requires small steps ($< 1/L$): in most cases, difficult to evaluate;*
- *strong constraints on f: bounded $\nabla^2 f$ bounded (f cannot be super-quadratic), risk of "bouncing or diverging steps";*

Convex optimization algorithms: convergence of gradient descent

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j = \delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*
- *ensured convergence for fixed steps: "no" step size adaptation required;*
- **BUT** *requires small steps ($< 1/L$): in most cases, difficult to evaluate;*
- *strong constraints on f: bounded $\nabla^2 f$ bounded (f cannot be super-quadratic), risk of "bouncing or diverging steps";*
- *f needs be everywhere differentiable for gradient to be evaluated;*

Convex optimization algorithms: convergence of gradient descent

Remark (Advantages/limitations of gradient descent)

- *simple to implement: for f not easily differentiable, gradient approximation $\{(f(x_k + \epsilon e_i) - f(x_k))/\epsilon\}_{i=1}^n$ with $[e_i]_j = \delta_i^j$ i-th canonical vector;*
- *quite flexible, generalizes in many ways: when f not perfectly known, stochastic gradient descent (averages well on the long run);*
- *ensured convergence for fixed steps: "no" step size adaptation required;*
- **BUT** *requires small steps ($< 1/L$): in most cases, difficult to evaluate;*
- *strong constraints on f: bounded $\nabla^2 f$ bounded (f cannot be super-quadratic), risk of "bouncing or diverging steps";*
- *f needs be everywhere differentiable for gradient to be evaluated;*
- *needs unbounded Ω: $x_k + t\nabla f(x_k)$ remains within the domain of f.*

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \leq \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

Convex optimization algorithms: convergence speed of gradient descent

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \leq \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

i.e., 100 steps lead to 1% error:

Convex optimization algorithms: convergence speed of gradient descent

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \le \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

i.e., 100 steps lead to 1% error: **this is quite slow!**, called **sublinear** convergence rate.

Convex optimization algorithms: convergence speed of gradient descent

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \leq \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

i.e., 100 steps lead to 1% error: **this is quite slow!**, called **sublinear** convergence rate.

We can do much better!

Convex optimization algorithms: convergence speed of gradient descent

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \leq \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

i.e., 100 steps lead to 1% error: **this is quite slow!**, called **sublinear** convergence rate.

We can do much better!

Theorem (Linear Convergence of Gradient Descent)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex, twice continuously differentiable, and $\forall x \in \mathcal{X}$,

$$lI_n \preceq \nabla^2 f(x) \preceq LI_n, \quad \text{for some } L \geq l > 0.$$

Then, ofr gradient descent algorithm with step size $t \leq \frac{1}{L}$,

$$f(x_k) - f(x^\star) \leq \alpha C^k, \quad C < 1.$$

Convex optimization algorithms: convergence speed of gradient descent

Remark: From the proof, convergence speed satisfies at least

$$f(x_k) - f(x^\star) \leq \frac{1}{2kt}\|x_1 - x^\star\|^2.$$

i.e., 100 steps lead to 1% error: **this is quite slow!**, called **sublinear** convergence rate.

We can do much better!

Theorem (Linear Convergence of Gradient Descent)

$f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$ convex, twice continuously differentiable, and $\forall x \in \mathcal{X}$,

$$lI_n \preceq \nabla^2 f(x) \preceq LI_n, \quad \text{for some } L \geq l > 0.$$

Then, ofr gradient descent algorithm with step size $t \leq \frac{1}{L}$,

$$f(x_k) - f(x^\star) \leq \alpha C^k, \quad C < 1.$$

Convergence is said **linear**.

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2} t \|\nabla f(x_k)\|^2$$

Convex optimization algorithms: convergence speed of gradient descent

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2} t \|\nabla f(x_k)\|^2$$

from which

$$f(x_{k+1}) - f(x^\star) \leq f(x_k) - f(x^\star) - \frac{1}{2} t \|\nabla f(x_k)\|^2. \tag{3}$$

Convex optimization algorithms: convergence speed of gradient descent

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2} t \|\nabla f(x_k)\|^2$$

from which

$$f(x_{k+1}) - f(x^\star) \leq f(x_k) - f(x^\star) - \frac{1}{2} t \|\nabla f(x_k)\|^2. \tag{3}$$

Also, by Taylor expansion: $\forall x, y \in \mathcal{X}$,

$$f(y) = f(x) + \nabla f(x)^\mathsf{T}(y - x) + \frac{1}{2}(y - x)^\mathsf{T} \nabla^2 f(\zeta)(y - x)$$

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2}t\|\nabla f(x_k)\|^2$$

from which

$$f(x_{k+1}) - f(x^\star) \leq f(x_k) - f(x^\star) - \frac{1}{2}t\|\nabla f(x_k)\|^2. \tag{3}$$

Also, by Taylor expansion: $\forall x, y \in \mathcal{X}$,

$$f(y) = f(x)+\nabla f(x)^\mathsf{T}(y-x)+\frac{1}{2}(y-x)^\mathsf{T}\nabla^2 f(\zeta)(y-x) \geq f(x)+\nabla f(x)^\mathsf{T}(y-x)+\frac{l}{2}\|y-x\|^2$$

Convex optimization algorithms: convergence speed of gradient descent

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2} t \|\nabla f(x_k)\|^2$$

from which

$$f(x_{k+1}) - f(x^\star) \leq f(x_k) - f(x^\star) - \frac{1}{2} t \|\nabla f(x_k)\|^2. \tag{3}$$

Also, by Taylor expansion: $\forall x, y \in \mathcal{X}$,

$$f(y) = f(x) + \nabla f(x)^\mathsf{T}(y-x) + \frac{1}{2}(y-x)^\mathsf{T} \nabla^2 f(\zeta)(y-x) \geq f(x) + \nabla f(x)^\mathsf{T}(y-x) + \frac{l}{2}\|y-x\|^2$$

Right-hand side minimized for $y = x - \frac{1}{l}\nabla f(x)$ (differentiate along y): $\forall x, y \in \mathcal{X}$,

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

Proof.

We already know, since $t \leq \frac{1}{L}$,

$$f(x_{k+1}) \leq f(x_k) - \frac{1}{2}t\|\nabla f(x_k)\|^2$$

from which

$$f(x_{k+1}) - f(x^\star) \leq f(x_k) - f(x^\star) - \frac{1}{2}t\|\nabla f(x_k)\|^2. \tag{3}$$

Also, by Taylor expansion: $\forall x, y \in \mathcal{X}$,

$$f(y) = f(x) + \nabla f(x)^\mathsf{T}(y-x) + \frac{1}{2}(y-x)^\mathsf{T}\nabla^2 f(\zeta)(y-x) \geq f(x) + \nabla f(x)^\mathsf{T}(y-x) + \frac{l}{2}\|y-x\|^2$$

Right-hand side minimized for $y = x - \frac{1}{l}\nabla f(x)$ (differentiate along y): $\forall x, y \in \mathcal{X}$,

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

Applied to $y = x^\star$ and $x = x_k$,

$$-\frac{t}{2}\|\nabla f(x_k)\|^2 \leq tl(f(x^\star) - f(x_k)).$$

Convex optimization algorithms: convergence speed of gradient descent

Proof.

Back to (3), this implies

$$f(x_{k+1}) - f(x^\star) \leq (1 - tl)\,(f(x_k) - f(x^\star)), \quad 1 - tl = C < 1 \text{ (by assumption)}.$$

Convex optimization algorithms: convergence speed of gradient descent

Proof.

Back to (3), this implies

$$f(x_{k+1}) - f(x^\star) \leq (1 - tl)\,(f(x_k) - f(x^\star)), \quad 1 - tl = C < 1 \text{ (by assumption)}.$$

Applied to $k = 1, \ldots, K$, this is

$$f(x_{K+1}) - f(x^\star) \leq C^K (f(x_1) - f(x^\star)).$$

$\square$

Convex optimization algorithms: Newton's method

Intuition of Newton's method: second-order Taylor expansion of f

$$f(x+h) = \underbrace{f(x) + \nabla f(x)^{\mathsf{T}} h + \frac{1}{2} h^{\mathsf{T}} \nabla^2 f(x) h}_{\equiv \hat{f}(x+h)} + o(\|h\|^2).$$

Convex optimization algorithms: Newton's method

Intuition of Newton's method: second-order Taylor expansion of f

$$f(x+h) = \underbrace{f(x) + \nabla f(x)^{\mathsf{T}} h + \frac{1}{2} h^{\mathsf{T}} \nabla^2 f(x) h}_{\equiv \hat{f}(x+h)} + o(\|h\|^2).$$

Idea:

- approximate $f(x+h)$ by $\hat{f}(x+h)$ for every $x \in \mathcal{X}$
- solve local minimization of $f(x+h)$ via minimization of $\hat{f}(x+h)$ for h, i.e., for

$$h = -[\nabla^2 f(x)]^{-1} \nabla f(x).$$

Convex optimization algorithms: Newton's method

Intuition of Newton's method: second-order Taylor expansion of f

$$f(x+h) = \underbrace{f(x) + \nabla f(x)^{\mathsf{T}} h + \frac{1}{2} h^{\mathsf{T}} \nabla^2 f(x) h}_{\equiv \hat{f}(x+h)} + o(\|h\|^2).$$

Idea:

- approximate $f(x+h)$ by $\hat{f}(x+h)$ for every $x \in \mathcal{X}$
- solve local minimization of $f(x+h)$ via minimization of $\hat{f}(x+h)$ for h, i.e., for

$$h = -[\nabla^2 f(x)]^{-1} \nabla f(x).$$

Definition (Newton's Method)

For f twice-differentiable and $\nabla^2 f(x) \succ 0$ for all $x \in \mathcal{X}$. Then Newton's method:

$$\begin{cases} \Delta x_k & = - \left[\nabla^2 f(x_k)\right] \nabla f(x_k). \\ t_k & = 1 \end{cases}$$

Convex optimization algorithms: Newton's method

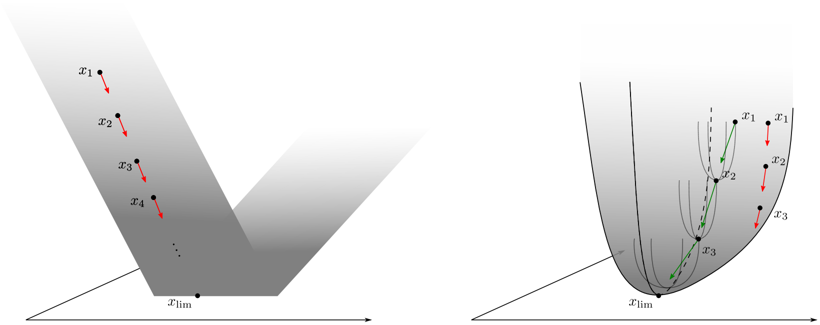


Figure: **(left)** Gradient descent fast on hyperplane-shaped f; **(right)** Newton improves convergence speed, while not following the *steepest descent*.

Convex optimization algorithms: Newton's method

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^{\mathsf{T}} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \leq 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^{\mathsf{T}} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \leq 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Remark

- *linear invariance: if* $x = Ay$ *and* $g(y) = f(x) = f(Ay)$*, and* $\{x_k\}$ *is a Newton descent on* f,

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^{\mathsf{T}} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \leq 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Remark

- *linear invariance: if* $x = Ay$ *and* $g(y) = f(x) = f(Ay)$*, and* $\{x_k\}$ *is a Newton descent on* f*, then* $y_{k+1} = Ax_{k+1}$ *is a Newton descent on* g*.*

Convex optimization algorithms: Newton's method

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^{\mathsf{T}} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \leq 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Remark

- *linear invariance: if* $x = Ay$ *and* $g(y) = f(x) = f(Ay)$*, and* $\{x_k\}$ *is a Newton descent on* f*, then* $y_{k+1} = Ax_{k+1}$ *is a Newton descent on* g*.*
  Not true for gradient descent!

Convex optimization algorithms: Newton's method

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^{\mathsf{T}} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \leq 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Remark

- *linear invariance: if* $x = Ay$ *and* $g(y) = f(x) = f(Ay)$*, and* $\{x_k\}$ *is a Newton descent on* f*, then* $y_{k+1} = Ax_{k+1}$ *is a Newton descent on* g*.*
  Not true for gradient descent!
- *If* $\nabla^2 f(x)$ *almost singular, Newton's method can be very slow and even diverge.*

Convex optimization algorithms: Newton's method

Property (Newton's Method is a Descent Method)

Since $\nabla^2 f(x) \succ 0$,

$$-\nabla f(x)^\mathsf{T} \left[\nabla^2 f(x_k)\right] \nabla f(x_k) \le 0$$

with equality for $\nabla f(x_k) = 0$*: Newton's method is a valid descent method.*

Remark

- *linear invariance: if* $x = Ay$ *and* $g(y) = f(x) = f(Ay)$*, and* $\{x_k\}$ *is a Newton descent on* f*, then* $y_{k+1} = Ax_{k+1}$ *is a Newton descent on* g*.*
  Not true for gradient descent!
- *If* $\nabla^2 f(x)$ *almost singular, Newton's method can be very slow and even diverge.*
- *For* $n \gg 1$*, can be extremely costly (inversion of* $\nabla^2 f(x_k)$ *for every* k*!).*

Convex optimization algorithms: Newton's method

Solution: to avoid singular $\nabla^2 f$, Newton with a step-size adaption,

Solution: to avoid singular $\nabla^2 f$, Newton with a step-size adaption,

Definition (Damped Newton's Method)

Damped Newton's method:

$$x_{k+1} = x_k - t_k \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k)$$

with t_k obtained by *backtracking line search*.

Convex optimization algorithms: Newton's method

Solution: to avoid singular $\nabla^2 f$, Newton with a step-size adaption,

Definition (Damped Newton's Method)

Damped Newton's method:

$$x_{k+1} = x_k - t_k \left[\nabla^2 f(x_k)\right]^{-1} \nabla f(x_k)$$

with t_k obtained by *backtracking line search*.

Theorem (Convergence of damped Newton's method)

Assume $lI_n \preceq \nabla^2 f(x) \preceq LI_n$ and $\nabla^2 f$ is M-Lipschitz, i.e.,

$$\forall x, y, \left\|\nabla^2 f(y) - \nabla^2 f(x)\right\| \leq M\|y - x\|.$$

Then damped Newton's method converges sublinearly then quadratically as soon as $\|\nabla f(x_k)\| < \eta$ for some small $\eta > 0$; besides, from this point on, $t_k = 1$.

Convex optimization algorithms: Newton's method

We only show the second part of the proof and take $t_k = 1$.

Convex optimization algorithms: Newton's method

We only show the second part of the proof and take $t_k = 1$.

Proof.

First write

$$\begin{aligned}
\|\nabla f(x_{k+1})\| &= \|\nabla f(x_k + \Delta x_k) \underbrace{-\nabla f(x_k) - \nabla^2 f(x_k)\Delta x_k}_{=0}\| \\
&= \left\| \int_0^1 (\nabla^2 f(x_k + u\Delta x_k) - \nabla^2 f(x_k))\Delta x_k du \right\| \\
&\le \frac{M}{2}\|\Delta x_k\|^2 = \frac{M}{2}\|[\nabla^2 f(x_k)]^{-1}\nabla f(x_k)\|^2 \le \frac{M}{2l^2}\|\nabla f(x_k)\|^2.
\end{aligned}$$

Convex optimization algorithms: Newton's method

We only show the second part of the proof and take $t_k = 1$.

Proof.

First write

$$\begin{aligned}
\|\nabla f(x_{k+1})\| &= \|\nabla f(x_k + \Delta x_k) \underbrace{-\nabla f(x_k) - \nabla^2 f(x_k)\Delta x_k}_{=0} \| \\
&= \left\| \int_0^1 (\nabla^2 f(x_k + u\Delta x_k) - \nabla^2 f(x_k))\Delta x_k du \right\| \\
&\le \frac{M}{2}\|\Delta x_k\|^2 = \frac{M}{2}\|[\nabla^2 f(x_k)]^{-1}\nabla f(x_k)\|^2 \le \frac{M}{2l^2}\|\nabla f(x_k)\|^2.
\end{aligned}$$

Multiplying both sides by $M/(2l^2)$,

$$\frac{M}{2l^2}\|\nabla f(x_K)\| \le \left(\frac{M}{2l^2}\|\nabla f(x_{k_0})\|\right)^2.$$

Convex optimization algorithms: Newton's method

We only show the second part of the proof and take $t_k = 1$.

Proof.

First write

$$\|\nabla f(x_{k+1})\| = \|\nabla f(x_k + \Delta x_k) \underbrace{-\nabla f(x_k) - \nabla^2 f(x_k)\Delta x_k}_{=0} \|$$

$$= \left\| \int_0^1 (\nabla^2 f(x_k + u\Delta x_k) - \nabla^2 f(x_k))\Delta x_k du \right\|$$

$$\leq \frac{M}{2}\|\Delta x_k\|^2 = \frac{M}{2}\|[\nabla^2 f(x_k)]^{-1}\nabla f(x_k)\|^2 \leq \frac{M}{2l^2}\|\nabla f(x_k)\|^2.$$

Multiplying both sides by $M/(2l^2)$,

$$\frac{M}{2l^2}\|\nabla f(x_K)\| \leq \left(\frac{M}{2l^2}\|\nabla f(x_{k_0})\|\right)^2.$$

Iterated over $k = k_0, \ldots, K$,

$$\|\nabla f(x_K)\| \leq \alpha C^{2^{K-k_0}}$$

with $C = \frac{M}{2l^2}\|\nabla f(x_{k_0})\| < 1$ if $\|\nabla f(x_{k_0})\| < \eta = \frac{2l^2}{M}$. $\square$

Outline

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Example (Linear Programming)

$$\min_{x \in \mathbb{R}^n} \{c^\mathsf{T} x\} \text{ such that } Ax \leq b \quad (Ax \leq b \text{ understood entry-wise})$$

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Example (Linear Programming)

$$\min_{x \in \mathbb{R}^n} \{c^\top x\} \text{ such that } Ax \leq b \quad (Ax \leq b \text{ understood entry-wise})$$

This is equivalent to

$$\min_{x \in \mathbb{R}^n,\ Ax \leq b} c^\top x$$

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Example (Linear Programming)

$$\min_{x \in \mathbb{R}^n} \{c^\top x\} \text{ such that } Ax \leq b \quad (Ax \leq b \text{ understood entry-wise})$$

This is equivalent to

$$\min_{x \in \mathbb{R}^n,\ Ax \leq b} c^\top x \quad \Leftrightarrow \quad \min_{x \in \mathbb{R}^n} c^\top x + \imath_{\{Ax \leq b\}}(x).$$

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Example (Linear Programming)

$$\min_{x \in \mathbb{R}^n} \{c^\top x\} \text{ such that } Ax \leq b \quad (Ax \leq b \text{ understood entry-wise})$$

This is equivalent to

$$\min_{x \in \mathbb{R}^n,\ Ax \leq b} c^\top x \quad \Leftrightarrow \quad \min_{x \in \mathbb{R}^n} c^\top x + \imath_{\{Ax \leq b\}}(x).$$

Solution: a corner point of Ω!

Inequality constrained optimization

Setup: So far, $\Omega \subset \mathcal{X}$ is unbounded. What if Ω has strict boundaries?

Example: if we impose $\forall i,\ [x]_i > 0$, what if gradient descent points to $[x]_i < 0$?

Example (Linear Programming)

$$\min_{x \in \mathbb{R}^n} \{c^\top x\} \text{ such that } Ax \leq b \quad (Ax \leq b \text{ understood entry-wise})$$

This is equivalent to

$$\min_{x \in \mathbb{R}^n,\ Ax \leq b} c^\top x \quad \Leftrightarrow \quad \min_{x \in \mathbb{R}^n} c^\top x + \imath_{\{Ax \leq b\}}(x).$$

Solution: a corner point of Ω!

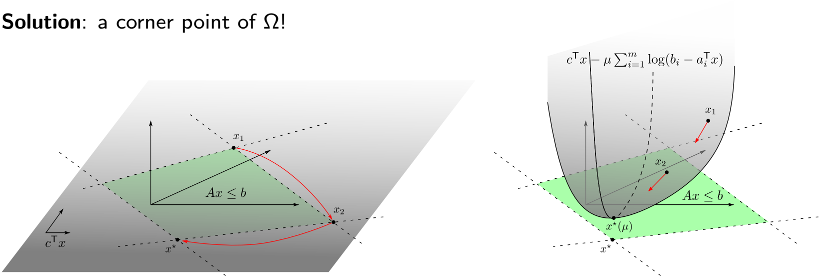


Figure: Linear Programming. **(left)** Simplex method; **(right)** barrier method.

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \; i = 1, \ldots, m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \; i = 1, \ldots, m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \;\; i = 1, \ldots, m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

- relax $f(x)$ via additional cost on *barriers* of constraint set.

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \;\; i = 1, \ldots, m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

- relax $f(x)$ via additional cost on *barriers* of constraint set.

Definition (Barrier Method)

For f continuously differentiable, for $\mu > 0$, let

$$\phi(x; \mu) \equiv f(x) - \mu \sum_{i=1}^{m} \log(c_i(x)).$$

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x\in\mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \ \ i = 1, \dots, m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

- relax $f(x)$ via additional cost on *barriers* of constraint set.

Definition (Barrier Method)

For f continuously differentiable, for $\mu > 0$, let

$$\phi(x;\mu) \equiv f(x) - \mu \sum_{i=1}^{m} \log(c_i(x)).$$

- Start with $x_0(\mu) \in \mathcal{X}$ such that $\forall i, \ \ c_i(x_0(\mu)) > 0$,

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x \in \mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0, \;\; i = 1, \ldots, m$$

where $c_i(x) = a_i^{\mathsf{T}} x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

- relax $f(x)$ via additional cost on *barriers* of constraint set.

Definition (Barrier Method)

For f continuously differentiable, for $\mu > 0$, let

$$\phi(x; \mu) \equiv f(x) - \mu \sum_{i=1}^{m} \log(c_i(x)).$$

- Start with $x_0(\mu) \in \mathcal{X}$ such that $\forall i, \;\; c_i(x_0(\mu)) > 0$,
- descent algorithm on

$$\min_{x \in \mathbb{R}^n} \phi(x; \mu)$$

  with solution $x^\star(\mu)$.

Inequality constrained optimization: the barrier method

Considered problem:

$$\min_{x\in\mathbb{R}^n} f(x) \text{ such that } c_i(x) \geq 0,\ \ i = 1,\ldots,m$$

where $c_i(x) = a_i^\top x - b_i$ for some $a_i, b_i \in \mathbb{R}^n$.

Generic solution: *Interior point* (or *barrier*) method:

- relax $f(x)$ via additional cost on *barriers* of constraint set.

Definition (Barrier Method)

For f continuously differentiable, for $\mu > 0$, let

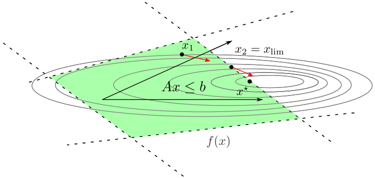
$$\phi(x;\mu) \equiv f(x) - \mu \sum_{i=1}^{m} \log(c_i(x)).$$

- Start with $x_0(\mu) \in \mathcal{X}$ such that $\forall i,\ \ c_i(x_0(\mu)) > 0$,
- descent algorithm on

$$\min_{x\in\mathbb{R}^n} \phi(x;\mu)$$

  with solution $x^\star(\mu)$.
- decrease μ and, starting from the previous $x^\star(\mu)$, repeat.

Inequality constrained optimization: the barrier method



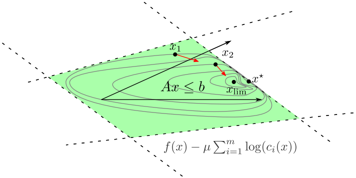


Figure: Barrier Method. **(left)** Level sets of f and constraint set: algorithm "stuck"; **(right)** Level sets of $f - \mu \sum_{i=1}^{m} \log(c_i(x))$ and constraint set: algorithm finds approximation for $x^\star$.

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*
  - *exacerbated for solutions near or at a constraint (solution hard to reach!).*

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*
  - *exacerbated for solutions near or at a constraint (solution hard to reach!).*
  - *on stark barriers, step sizes need very thin adapting: avoid "jumps" over solution.*

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*
  - *exacerbated for solutions near or at a constraint (solution hard to reach!).*
  - *on stark barriers, step sizes need very thin adapting: avoid "jumps" over solution.*
- *barrier method only valid for inequality constraints.*

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*
  - *exacerbated for solutions near or at a constraint (solution hard to reach!).*
  - *on stark barriers, step sizes need very thin adapting: avoid "jumps" over solution.*
- *barrier method only valid for inequality constraints.*

Consequence: barrier method not often used in practice, but flexible to all f.

Inequality constrained optimization: the barrier method

Remark (Difficulties of Barrier Method)

Far from ideal...:

- *descent directions may be invalid: line-search or backtrack necessary to stay in Ω;*
- *costly double-iteration with refined μ; often difficult to handle:*
  - *initialization point in next μ-step must be close to μ-step solution to avoid slow descents (but too small μ-updates slows convergence).*
  - *exacerbated for solutions near or at a constraint (solution hard to reach!).*
  - *on stark barriers, step sizes need very thin adapting: avoid "jumps" over solution.*
- *barrier method only valid for inequality constraints.*

Consequence: barrier method not often used in practice, but flexible to all f.

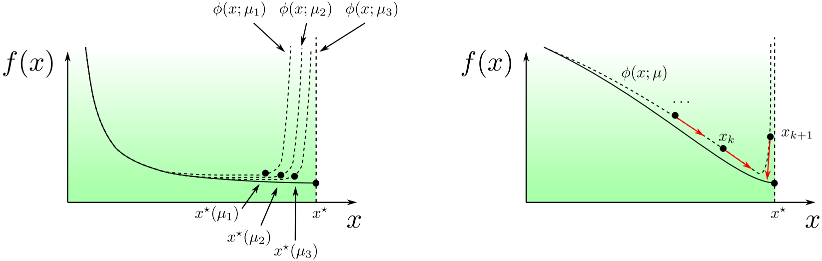


Figure: Barrier Method. **(left)** Sequence of $\phi(x;\mu)$ approx; **(right)** Difficulty raised by sharp minima and "ping-ponging" effect.

Outline

Outline

Linear constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } h_i(x) = 0, \;\; i = 1, \ldots, p. \tag{4}$$

Linear constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } h_i(x) = 0, \; i = 1, \ldots, p. \tag{4}$$

Theorem

If $x^\star$ solution to (4), then $\exists \lambda_1, \ldots, \lambda_p \in \mathbb{R}$ such that

$$\nabla f(x^\star) = \sum_{i=1}^{p} (-\lambda_i) \nabla h_i(x^\star).$$

Linear constraints

Geometric Proof for $p = 1$.

1. **Gradient orthogonal to level sets**: level set $\ell_c(g) \equiv \{x \mid g(x) = c\}$.

Linear constraints

Geometric Proof for $p = 1$.

1. Gradient orthogonal to level sets: level set $\ell_c(g) \equiv \{x \mid g(x) = c\}$.
For $h \in \mathcal{X}$ such that $g(x) = g(x+h) = c$ and $\|h\| \to 0$,

$$0 = (g(x+h) - g(x))/\|h\| = \nabla g(x)^{\mathsf{T}}(h/\|h\|) + o(1)$$

Geometric Proof for $p = 1$.

1. Gradient orthogonal to level sets: level set $\ell_c(g) \equiv \{x \mid g(x) = c\}$.
For $h \in \mathcal{X}$ such that $g(x) = g(x + h) = c$ and $\|h\| \to 0$,

$$0 = (g(x + h) - g(x))/\|h\| = \nabla g(x)^{\mathsf{T}}(h/\|h\|) + o(1)$$

Thus $\nabla g(x)$ orthogonal to $\ell_c(g)$.

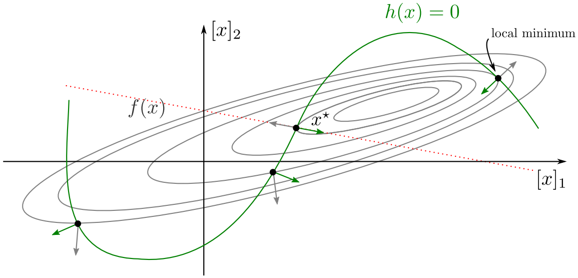
Geometric Proof for $p = 1$.

1. Gradient orthogonal to level sets: level set $\ell_c(g) \equiv \{x \mid g(x) = c\}$.
For $h \in \mathcal{X}$ such that $g(x) = g(x + h) = c$ and $\|h\| \to 0$,

$$0 = (g(x + h) - g(x))/\|h\| = \nabla g(x)^{\mathsf{T}}(h/\|h\|) + o(1)$$

Thus $\nabla g(x)$ orthogonal to $\ell_c(g)$.

2. Gradient of f and h aligned at local minimum: see Figure. In particular true for $x^\star$, so $\exists\lambda$ such that $\nabla f(x^\star) = \lambda \nabla h(x^\star)$.



Linear constraints

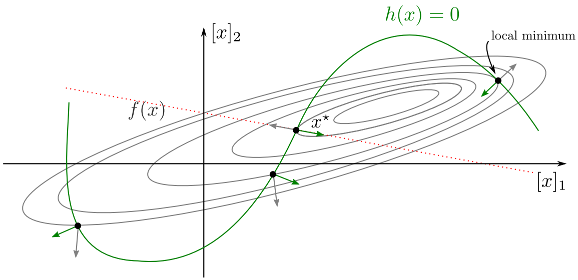
Geometric Proof for $p = 1$.

1. Gradient orthogonal to level sets: level set $\ell_c(g) \equiv \{x \mid g(x) = c\}$.
For $h \in \mathcal{X}$ such that $g(x) = g(x+h) = c$ and $\|h\| \to 0$,

$$0 = (g(x+h) - g(x))/\|h\| = \nabla g(x)^{\mathsf{T}}(h/\|h\|) + o(1)$$

Thus $\nabla g(x)$ orthogonal to $\ell_c(g)$.

2. Gradient of f and h aligned at local minimum: see Figure. In particular true for $x^\star$, so $\exists \lambda$ such that $\nabla f(x^\star) = \lambda \nabla h(x^\star)$.



3. When minimum of f coincides with $h(x) = 0$: formula still holds with $\lambda = 0$. □

Linear constraints: Lagrange dual

Consequence: *Necessary condition* for extremum for f under the constraints h_i: find x such that $f(x) + \sum_i \lambda_i h_i(x)$ has zero gradient for some $\lambda_1, \ldots, \lambda_p$.

Linear constraints: Lagrange dual

Consequence: *Necessary condition* for extremum for f under the constraints h_i: find x such that $f(x) + \sum_i \lambda_i h_i(x)$ has zero gradient for some $\lambda_1, \ldots, \lambda_p$.

Definition (Lagrange dual function)

For $\lambda \in \mathbb{R}^p$, Lagrange dual g of f is

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda)$$

$$L(x; \lambda) \equiv f(x) + \sum_{i=1}^{p} \lambda_i h_i(x).$$

Linear constraints: Lagrange dual

Consequence: *Necessary condition* for extremum for f under the constraints h_i: find x such that $f(x) + \sum_i \lambda_i h_i(x)$ has zero gradient for some $\lambda_1, \ldots, \lambda_p$.

Definition (Lagrange dual function)

For $\lambda \in \mathbb{R}^p$, Lagrange dual g of f is

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda)$$

$$L(x; \lambda) \equiv f(x) + \sum_{i=1}^{p} \lambda_i h_i(x).$$

The coefficients $\lambda_1, \ldots, \lambda_p$ are called the *Lagrange multipliers*.

Linear constraints: Lagrange dual

Consequence: *Necessary condition* for extremum for f under the constraints h_i: find x such that $f(x) + \sum_i \lambda_i h_i(x)$ has zero gradient for some $\lambda_1, \ldots, \lambda_p$.

Definition (Lagrange dual function)

For $\lambda \in \mathbb{R}^p$, Lagrange dual g of f is

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda)$$

$$L(x; \lambda) \equiv f(x) + \sum_{i=1}^{p} \lambda_i h_i(x).$$

The coefficients $\lambda_1, \ldots, \lambda_p$ are called the *Lagrange multipliers*.

Property (Lagrange dual as lower bound)

For $x^\star$ solution, since $h_i(x^\star) = 0$, we have for all $\lambda \in \mathbb{R}^p$,

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda) \leq L(x^\star; \lambda) = f(x^\star).$$

Linear constraints: Lagrange dual

Consequence: *Necessary condition* for extremum for f under the constraints h_i: find x such that $f(x) + \sum_i \lambda_i h_i(x)$ has zero gradient for some $\lambda_1, \ldots, \lambda_p$.

Definition (Lagrange dual function)

For $\lambda \in \mathbb{R}^p$, Lagrange dual g of f is

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda)$$

$$L(x; \lambda) \equiv f(x) + \sum_{i=1}^{p} \lambda_i h_i(x).$$

The coefficients $\lambda_1, \ldots, \lambda_p$ are called the *Lagrange multipliers*.

Property (Lagrange dual as lower bound)

For $x^\star$ solution, since $h_i(x^\star) = 0$, we have for all $\lambda \in \mathbb{R}^p$,

$$g(\lambda) = \inf_{x \in \mathcal{X}} L(x; \lambda) \leq L(x^\star; \lambda) = f(x^\star).$$

In particular

$$\sup_{\lambda \in \mathbb{R}^p} g(\lambda) \leq f(x^\star).$$

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x; \lambda) \right\}.$$

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x; \lambda) \right\}.$$

We denote $\lambda^\star \in \mathbb{R}^n$ any point of $\operatorname{argmax}_\lambda g(\lambda)$ (maybe empty).

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x; \lambda) \right\}.$$

We denote $\lambda^\star \in \mathbb{R}^n$ any point of $\operatorname{argmax}_\lambda g(\lambda)$ (maybe empty).

- $g(\lambda^\star) - f(x^\star) \geq 0$ is the *duality gap*

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x; \lambda) \right\}.$$

We denote $\lambda^\star \in \mathbb{R}^n$ any point of $\text{argmax}_\lambda\, g(\lambda)$ (maybe empty).

- $g(\lambda^\star) - f(x^\star) \geq 0$ is the *duality gap*
- if duality gap is zero, the original (*primal*) problem is solved by Lagrange dual.

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x;\lambda) \right\}.$$

We denote $\lambda^\star \in \mathbb{R}^n$ any point of $\operatorname{argmax}_\lambda g(\lambda)$ (maybe empty).

- $g(\lambda^\star) - f(x^\star) \geq 0$ is the *duality gap*
- if duality gap is zero, the original (*primal*) problem is solved by Lagrange dual.

Property

Lagrange dual $\lambda \mapsto g(\lambda)$ is concave, irrespective of f (convex or not!).

Linear constraints: Lagrange dual

Definition (Lagrange dual problem)

$$\sup_{\lambda \in \mathbb{R}^n} g(\lambda) = \sup_{\lambda \in \mathbb{R}^p} \left\{ \inf_{x \in \mathcal{X}} L(x; \lambda) \right\}.$$

We denote $\lambda^\star \in \mathbb{R}^n$ any point of $\operatorname{argmax}_\lambda g(\lambda)$ (maybe empty).

- $g(\lambda^\star) - f(x^\star) \geq 0$ is the *duality gap*
- if duality gap is zero, the original (*primal*) problem is solved by Lagrange dual.

Property

Lagrange dual $\lambda \mapsto g(\lambda)$ is concave, irrespective of f (convex or not!).

Proof.

For $\lambda_1, \lambda_2 \in \mathbb{R}^p$, $\alpha \in [0, 1]$,

$$\begin{aligned} g(\alpha\lambda_1 + (1-\alpha)\lambda_2) &= \inf_{x \in \mathcal{X}} \left\{ \alpha \left(f(x) + \sum_{i=1}^p \lambda_{1i} h_i(x) \right) + (1-\alpha) \left(f(x) + \sum_{i=1}^p \lambda_{2i} h_i(x) \right) \right\} \\ &\geq \alpha \inf_{x \in \mathcal{X}} \left\{ f(x) + \sum_{i=1}^p \lambda_{1i} h_i(x) \right\} + (1-\alpha) \inf_{x \in \mathcal{X}} \left\{ f(x) + \sum_{i=1}^p \lambda_{2i} h_i(x) \right\} \\ &= \alpha g(\lambda_1) + (1-\alpha) g(\lambda_2) \end{aligned}$$

(inequality follows from $\inf_x\{f_1(x) + f_2(x)\} \geq \inf_x\{f_1(x)\} + \inf_x\{f_2(x)\}$).

Linear constraints: strong duality

Remarks:

- $\inf_\lambda -g(\lambda)$ convex: dual can be solved by standard *unconstrained* convex optimization.

Linear constraints: strong duality

Remarks:

- $\inf_\lambda -g(\lambda)$ convex: dual can be solved by standard *unconstrained* convex optimization.
- if f not convex ($\min f$ difficult to solve), at least $\max g$ can be solved: lower bounding $\min f$.

Linear constraints: strong duality

Remarks:

- $\inf_\lambda -g(\lambda)$ convex: dual can be solved by standard *unconstrained* convex optimization.
- if f not convex ($\min f$ difficult to solve), at least $\max g$ can be solved: lower bounding $\min f$.

Theorem (Slater's condition for strong duality)

If $\exists x \in \mathcal{X}$ such that $\forall i,\ h_i(x) = 0$ (feasibility), f is convex and h_i affine ($h_i(x) = a_i^\top x + b_i$), then strong duality holds.

Proof.

Let $\bar\lambda \in \mathbb{R}^p$ be such that $\nabla f(x^\star) = \sum_{i=1}^p (-\bar\lambda_i)\nabla h_i(x^\star)$. Then

$$g(\bar\lambda) = \inf_{x\in\mathcal{X}} f(x) + \sum_{i=1}^p \bar\lambda_i h_i(x) = f(x^\star).$$

Linear constraints: strong duality

Remarks:

- $\inf_\lambda -g(\lambda)$ convex: dual can be solved by standard *unconstrained* convex optimization.
- if f not convex ($\min f$ difficult to solve), at least $\max g$ can be solved: lower bounding $\min f$.

Theorem (Slater's condition for strong duality)

If $\exists x \in \mathcal{X}$ such that $\forall i,\ h_i(x) = 0$ (feasibility), f is convex and h_i affine ($h_i(x) = a_i^\top x + b_i$), then strong duality holds.

Proof.

Let $\bar{\lambda} \in \mathbb{R}^p$ be such that $\nabla f(x^\star) = \sum_{i=1}^p (-\bar{\lambda}_i) \nabla h_i(x^\star)$. Then

$$g(\bar{\lambda}) = \inf_{x \in \mathcal{X}} f(x) + \sum_{i=1}^p \bar{\lambda}_i h_i(x) = f(x^\star).$$

Indeed, $x \mapsto f(x) + \sum_{i=1}^p \bar{\lambda}_i h_i(x)$ convex (h_i affine), so minimal at zero gradient: true for x having same cost as $x^\star$, i.e., $f(x^\star) + \sum_{i=1}^p \bar{\lambda}_i h_i(x^\star) = f(x^\star)$.

Linear constraints: strong duality

Remarks:

- $\inf_\lambda -g(\lambda)$ convex: dual can be solved by standard *unconstrained* convex optimization.
- if f not convex ($\min f$ difficult to solve), at least $\max g$ can be solved: lower bounding $\min f$.

Theorem (Slater's condition for strong duality)

If $\exists x \in \mathcal{X}$ such that $\forall i,\ h_i(x) = 0$ (feasibility), f is convex and h_i affine ($h_i(x) = a_i^\top x + b_i$), then strong duality holds.

Proof.

Let $\bar{\lambda} \in \mathbb{R}^p$ be such that $\nabla f(x^\star) = \sum_{i=1}^p (-\bar{\lambda}_i) \nabla h_i(x^\star)$. Then

$$g(\bar{\lambda}) = \inf_{x \in \mathcal{X}} f(x) + \sum_{i=1}^p \bar{\lambda}_i h_i(x) = f(x^\star).$$

Indeed, $x \mapsto f(x) + \sum_{i=1}^p \bar{\lambda}_i h_i(x)$ convex (h_i affine), so minimal at zero gradient: true for x having same cost as $x^\star$, i.e., $f(x^\star) + \sum_{i=1}^p \bar{\lambda}_i h_i(x^\star) = f(x^\star)$.
As a consequence,

$$g(\lambda^\star) = \max_{\lambda \in \mathbb{R}^p} g(\lambda) \geq g(\bar{\lambda}) = f(x^\star)$$
$$g(\lambda^\star) \leq f(x^\star)$$

so $g(\lambda^\star) = f(x^\star)$. $\square$

Outline

Equality and inequality constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0,\ \ i = 1, \ldots, m \text{ and } h_j(x) = 0,\ j = 1, \ldots, p. \tag{5}$$

Equality and inequality constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0, \; i = 1, \ldots, m \text{ and } h_j(x) = 0, \; j = 1, \ldots, p. \qquad (5)$$

Method: For inequalities, additional multipliers.

Equality and inequality constraints

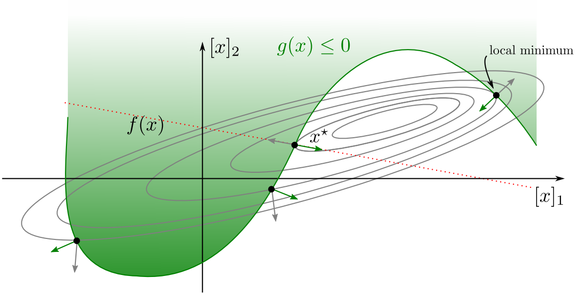
$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0, \; i = 1, \ldots, m \text{ and } h_j(x) = 0, \; j = 1, \ldots, p. \tag{5}$$

Method: For inequalities, additional multipliers. Main difference: multipliers imposed to be *positive*.

Equality and inequality constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0, \; i = 1, \ldots, m \text{ and } h_j(x) = 0, \; j = 1, \ldots, p. \tag{5}$$

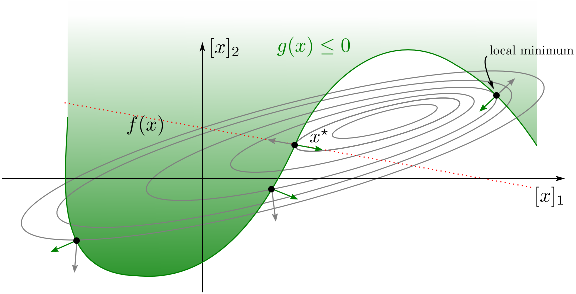
Method: For inequalities, additional multipliers. Main difference: multipliers imposed to be *positive*.



Equality and inequality constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0,\ i = 1, \ldots, m \text{ and } h_j(x) = 0,\ j = 1, \ldots, p. \quad (5)$$

Method: For inequalities, additional multipliers. Main difference: multipliers imposed to be *positive*.

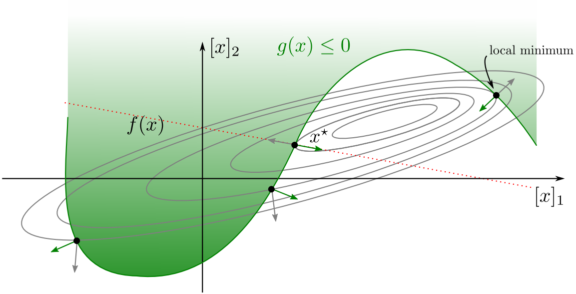


- if, at minimum, **constraint enforced** (minimum at edge), inequality becomes equality: Lagrangian multiplier non zero and positive (see figure).

Equality and inequality constraints

$$\min_{x \in \mathcal{X}} f(x) \text{ such that } g_i(x) \leq 0,\ i = 1, \ldots, m \text{ and } h_j(x) = 0,\ j = 1, \ldots, p. \tag{5}$$

Method: For inequalities, additional multipliers. Main difference: multipliers imposed to be *positive*.



- if, at minimum, **constraint enforced** (minimum at edge), inequality becomes equality: Lagrangian multiplier non zero and positive (see figure).
- if **constraint not enforced** (minimum within constraint set), then Lagrange multiplier is zero.

Equality and inequality constraints

Definition (Lagrange Dual Problem)

Lagrange dual of (5) is

$$\max_{\lambda \in \mathbb{R}^p,\ \nu \in \mathbb{R}^m_+} g(\lambda, \nu), \quad g(\lambda, \nu) \equiv \inf_{x \in \mathcal{X}} L(x; \lambda, \nu)$$

$$L(x; \lambda, \nu) \equiv f(x) + \sum_{i=1}^{m} \nu_i g_i(x) + \sum_{j=1}^{p} \lambda_j h_j(x).$$

Equality and inequality constraints

Definition (Lagrange Dual Problem)

Lagrange dual of (5) is

$$\max_{\lambda \in \mathbb{R}^p,\ \nu \in \mathbb{R}_+^m} g(\lambda, \nu), \quad g(\lambda, \nu) \equiv \inf_{x \in \mathcal{X}} L(x; \lambda, \nu)$$

$$L(x; \lambda, \nu) \equiv f(x) + \sum_{i=1}^{m} \nu_i g_i(x) + \sum_{j=1}^{p} \lambda_j h_j(x).$$

Theorem (Slater's Condition)

For f be convex, g_i convex, h_j affine, and $\exists x \in \mathcal{X}$ such that $h_i(x) = 0$ and $g_j(x) \leq 0$ for all i, j (feasibility). Then strong duality holds.

Equality and inequality constraints

Definition (Lagrange Dual Problem)

Lagrange dual of (5) is

$$\max_{\lambda \in \mathbb{R}^p,\ \nu \in \mathbb{R}^m_+} g(\lambda, \nu), \quad g(\lambda, \nu) \equiv \inf_{x \in \mathcal{X}} L(x; \lambda, \nu)$$

$$L(x; \lambda, \nu) \equiv f(x) + \sum_{i=1}^{m} \nu_i g_i(x) + \sum_{j=1}^{p} \lambda_j h_j(x).$$

Theorem (Slater's Condition)

For f be convex, g_i convex, h_j affine, and $\exists x \in \mathcal{X}$ such that $h_i(x) = 0$ and $g_j(x) \leq 0$ for all i, j (feasibility). Then strong duality holds.

Remark:

- for g_j convex, $\mathcal{G}_j = \left\{x \middle| g_j(x) \leq 0\right\}$ is convex.
- for h_i affine, $\mathcal{H}_i = \left\{x \middle| h_i(x) = 0\right\}$ also convex (but not if h_i convex!).

Equality and inequality constraints

Definition (Lagrange Dual Problem)

Lagrange dual of (5) is

$$\max_{\lambda \in \mathbb{R}^p,\ \nu \in \mathbb{R}_+^m} g(\lambda, \nu), \quad g(\lambda, \nu) \equiv \inf_{x \in \mathcal{X}} L(x; \lambda, \nu)$$

$$L(x; \lambda, \nu) \equiv f(x) + \sum_{i=1}^{m} \nu_i g_i(x) + \sum_{j=1}^{p} \lambda_j h_j(x).$$

Theorem (Slater's Condition)

For f be convex, g_i convex, h_j affine, and $\exists x \in \mathcal{X}$ such that $h_i(x) = 0$ and $g_j(x) \leq 0$ for all i, j (feasibility). Then strong duality holds.

Remark:

- for g_j convex, $\mathcal{G}_j = \left\{x \middle| g_j(x) \leq 0\right\}$ is convex.
- for h_i affine, $\mathcal{H}_i = \left\{x \middle| h_i(x) = 0\right\}$ also convex (but not if h_i convex!).
- Hence,

$$x^\star = \arg \min_{\mathcal{X} \cap \left(\bigcap_j \mathcal{G}_j\right) \cap \left(\bigcap_i \mathcal{H}_i\right)} f(x)$$

  i.e., minimising convex f over convex set.

Outline

Outline

Non-differentiable optimization

Setup: f convex but not everywhere differentiable.

Non-differentiable optimization

Setup: f convex but not everywhere differentiable.

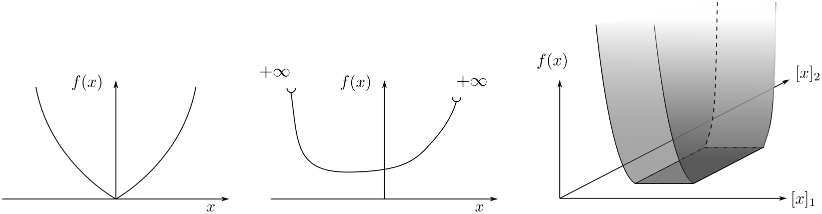


Figure: Examples of not-everywhere differentiable convex functions

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^\mathsf{T}(z - x).$$

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^{\mathsf{T}}(z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^{\mathsf{T}}(z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Generalization: *subdifferential* of convex f:

Definition (Subdifferential)

Let $f : \mathcal{X} \to \mathbb{R}$. The *subdifferential* ∂f of f is

$$\begin{aligned} \partial f : \mathcal{X} &\to 2^{\mathcal{X}} \\ x &\mapsto \left\{ u \in \mathcal{X} \mid \forall z \in \mathcal{X},\ f(x) \leq f(z) + u^{\mathsf{T}}(x - z) \right\}. \end{aligned}$$

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^\mathsf{T}(z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Generalization: *subdifferential* of convex f:

Definition (Subdifferential)

Let $f : \mathcal{X} \to \mathbb{R}$. The *subdifferential* ∂f of f is

$$\begin{aligned} \partial f &: \mathcal{X} \to 2^{\mathcal{X}} \\ &x \mapsto \left\{ u \in \mathcal{X} \mid \forall z \in \mathcal{X},\ f(x) \leq f(z) + u^\mathsf{T}(x - z) \right\}. \end{aligned}$$

Careful: $\partial f(x)$ is a *set-valued* function: members of the set are the *subderivatives*.

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^{\mathsf{T}}(z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Generalization: *subdifferential* of convex f:

Definition (Subdifferential)

Let $f : \mathcal{X} \to \mathbb{R}$. The *subdifferential* ∂f of f is

$$\begin{aligned} \partial f &: \mathcal{X} \to 2^{\mathcal{X}} \\ & x \mapsto \left\{ u \in \mathcal{X} \mid \forall z \in \mathcal{X},\ f(x) \leq f(z) + u^{\mathsf{T}}(x - z) \right\}. \end{aligned}$$

Careful: $\partial f(x)$ is a *set-valued* function: members of the set are the *subderivatives*.

Property

For convex f, $\partial f(x)$ at those x where f is differentiable is a singleton:

$$\partial f(x) = \{\nabla f(x)\}.$$

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^\top (z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Generalization: *subdifferential* of convex f:

Definition (Subdifferential)

Let $f : \mathcal{X} \to \mathbb{R}$. The *subdifferential* ∂f of f is

$$\begin{aligned} \partial f : \mathcal{X} &\to 2^{\mathcal{X}} \\ x &\mapsto \left\{ u \in \mathcal{X} \mid \forall z \in \mathcal{X},\ f(x) \leq f(z) + u^\top (x - z) \right\}. \end{aligned}$$

Careful: $\partial f(x)$ is a *set-valued* function: members of the set are the *subderivatives*.

Property

For convex f, $\partial f(x)$ at those x where f is differentiable is a singleton:

$$\partial f(x) = \{\nabla f(x)\}.$$

Proof.

Let $u \in \partial f(x)$, hence $x \in \arg\min_{z \in \mathcal{X}} f(z) - u^\top z$.i

Non-differentiable optimization: subgradient

Reminder: first order conditions for convex differentiable $f : \mathcal{X} \to \mathbb{R} \cup \{+\infty\}$,

$$\forall x, z \in \mathrm{dom}(f), \quad f(z) \geq f(x) + \nabla f(x)^{\mathsf{T}}(z - x).$$

$\longrightarrow$ can be used to define ∇f for convex f: only linear function satisfying inequality.

Generalization: *subdifferential* of convex f:

Definition (Subdifferential)

Let $f : \mathcal{X} \to \mathbb{R}$. The *subdifferential* ∂f of f is

$$\begin{aligned} \partial f : \mathcal{X} &\to 2^{\mathcal{X}} \\ x &\mapsto \left\{ u \in \mathcal{X} \mid \forall z \in \mathcal{X},\ f(x) \leq f(z) + u^{\mathsf{T}}(x - z) \right\}. \end{aligned}$$

Careful: $\partial f(x)$ is a *set-valued* function: members of the set are the *subderivatives*.

Property

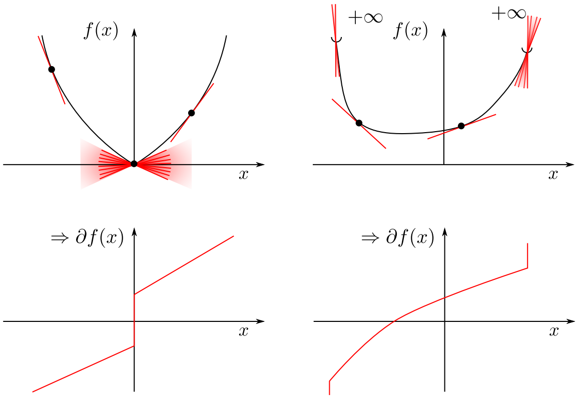
For convex f, $\partial f(x)$ at those x where f is differentiable is a singleton:

$$\partial f(x) = \{\nabla f(x)\}.$$

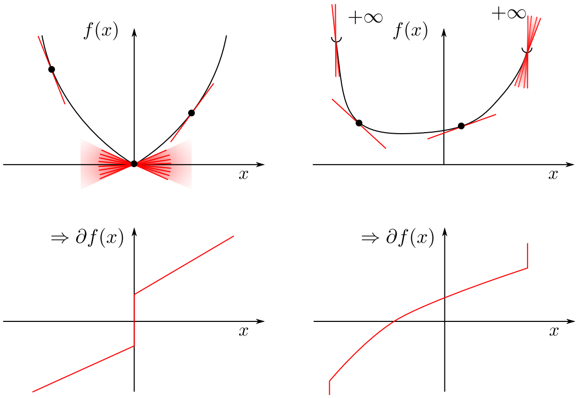
Proof.

Let $u \in \partial f(x)$, hence $x \in \arg\min_{z \in \mathcal{X}} f(z) - u^{\mathsf{T}} z$.i
Since f differentiable at x, first order condition gives $\nabla f(x) = u$. □

Non-differentiable optimization: subgradient



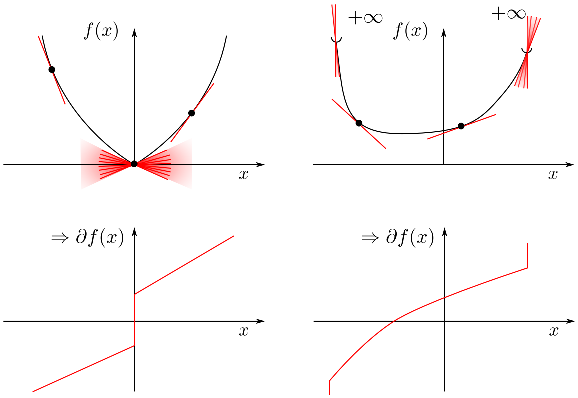
Non-differentiable optimization: subgradient



Property (Subdifferential as a convex set)

$\partial f(x)$ *is a nonempty convex compact set.*

Non-differentiable optimization: subgradient



Property (Subdifferential as a convex set)

$\partial f(x)$ *is a nonempty convex compact set.*

Property (Subdifferential as union of supporting hyperplanes)

$\partial f(x)$ *consists of the hyperplanes that support* $\mathrm{epi}(f)$ *at* $(x, f(x))$.

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ *convex,*

$$x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ convex,

$$x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Proof.

$\Rightarrow$. $\partial f(x^\star)$ must (at least) contain 0, since $x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} f(x) + 0^\mathsf{T} x$.

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ convex,

$$x^\star \in \text{argmin}_{x\in\mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Proof.

$\Rightarrow$. $\partial f(x^\star)$ must (at least) contain 0, since $x^\star \in \text{argmin}_{x\in\mathcal{X}} f(x) + 0^\mathsf{T} x$.

$\Leftarrow$. $0 \in \partial f(x) \implies x \in \arg\min_{z\in\mathcal{X}} f(z)$, but then x must be a solution. $\square$

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ convex,

$$x^\star \in \operatorname{argmin}_{x\in\mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Proof.

$\Rightarrow$. $\partial f(x^\star)$ must (at least) contain 0, since $x^\star \in \operatorname{argmin}_{x\in\mathcal{X}} f(x) + 0^\mathsf{T} x$.

$\Leftarrow$. $0 \in \partial f(x) \implies x \in \arg\min_{z\in\mathcal{X}} f(z)$, but then x must be a solution. □

Careful: looking for 0 in one of the sets $\partial f(x)$, $x \in \mathcal{X}$, different from looking for singleton $\{0\}$ among the sets $\partial f(x)$, $x \in \mathcal{X}$.

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ convex,

$$x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Proof.

$\Rightarrow$. $\partial f(x^\star)$ must (at least) contain 0, since $x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} f(x) + 0^\mathsf{T} x$.

$\Leftarrow$. $0 \in \partial f(x) \implies x \in \arg\min_{z \in \mathcal{X}} f(z)$, but then x must be a solution. $\square$

Careful: looking for 0 in one of the sets $\partial f(x)$, $x \in \mathcal{X}$, different from looking for singleton $\{0\}$ among the sets $\partial f(x)$, $x \in \mathcal{X}$.

Definition (Subgradient algorithm)

Under conditions of gradient descent theorem, with *all Lipschitz subgradients*, subgradient algorithm:

1. $x_{k+1} = x_k - t_k u_k$, for any $u_k \in \partial f(x_k)$
2. $f_{\mathrm{best}}^{k+1} = \min\{f_{\mathrm{best}}^k, f(x_{k+1})\}$.

Non-differentiable optimization: subgradient

Theorem (Fermat's rule extension)

For $f : \mathcal{X} \to \mathbb{R}$ convex,

$$x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star).$$

Proof.

$\Rightarrow$. $\partial f(x^\star)$ must (at least) contain 0, since $x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} f(x) + 0^\mathsf{T} x$.

$\Leftarrow$. $0 \in \partial f(x) \implies x \in \arg\min_{z \in \mathcal{X}} f(z)$, but then x must be a solution. □

Careful: looking for 0 in one of the sets $\partial f(x)$, $x \in \mathcal{X}$, different from looking for singleton $\{0\}$ among the sets $\partial f(x)$, $x \in \mathcal{X}$.

Definition (Subgradient algorithm)

Under conditions of gradient descent theorem, with *all Lipschitz subgradients*, subgradient algorithm:

1. $x_{k+1} = x_k - t_k u_k$, for any $u_k \in \partial f(x_k)$
2. $f_{\mathrm{best}}^{k+1} = \min\{f_{\mathrm{best}}^k, f(x_{k+1})\}$.

Remark: 2nd step underlies major weakness of the method (rarely used in practice): algorithm is *not* a descent method.

Outline

From subgradient to proximal

Exercise (The Projection Operator)

For Ω a convex set and $\imath_\Omega$ the set indicator ($\imath_\Omega(x) = 0$ if $x \in \Omega$ and $= +\infty$ if not), define

$$\min_{x \in \mathcal{X}} \frac{1}{2}\|x - y\|^2 + \imath_\Omega(x).$$

From subgradient to proximal

Exercise (The Projection Operator)

For Ω a convex set and $\imath_\Omega$ the set indicator ($\imath_\Omega(x) = 0$ if $x \in \Omega$ and $= +\infty$ if not), define

$$\min_{x \in \mathcal{X}} \frac{1}{2}\|x - y\|^2 + \imath_\Omega(x).$$

Show that $x^\star$ is the (Euclidean) projection of y onto $\Omega \cap \mathcal{X}$.

From subgradient to proximal

Exercise (The Projection Operator)

For Ω a convex set and $\imath_\Omega$ the set indicator ($\imath_\Omega(x) = 0$ if $x \in \Omega$ and $= +\infty$ if not), define

$$\min_{x \in \mathcal{X}} \frac{1}{2}\|x - y\|^2 + \imath_\Omega(x).$$

Show that $x^\star$ is the (Euclidean) projection of y onto $\Omega \cap \mathcal{X}$.

Projection and proximity: $x^\star$ is the "proximal" point of y:

From subgradient to proximal

Exercise (The Projection Operator)

For Ω a convex set and $\imath_\Omega$ the set indicator ($\imath_\Omega(x) = 0$ if $x \in \Omega$ and $= +\infty$ if not), define

$$\min_{x \in \mathcal{X}} \frac{1}{2}\|x - y\|^2 + \imath_\Omega(x).$$

Show that $x^\star$ is the (Euclidean) projection of y onto $\Omega \cap \mathcal{X}$.

Projection and proximity: $x^\star$ is the "proximal" point of y:

- stays close to x (through $\| \cdot - y\|^2$ term)

From subgradient to proximal

Exercise (The Projection Operator)

For Ω a convex set and $\imath_\Omega$ the set indicator ($\imath_\Omega(x) = 0$ if $x \in \Omega$ and $= +\infty$ if not), define

$$\min_{x \in \mathcal{X}} \frac{1}{2}\|x - y\|^2 + \imath_\Omega(x).$$

Show that $x^\star$ is the (Euclidean) projection of y onto $\Omega \cap \mathcal{X}$.

Projection and proximity: $x^\star$ is the "proximal" point of y:

- stays close to x (through $\|\cdot - y\|^2$ term)
- simultaneously (approximately) minimizes objective function, here $\imath_\Omega$.

Non-differentiable optimization: proximal methods

Definition (The Proximal (Point) Operator)

For $f : \mathcal{X} \to \mathbb{R}$ convex, proximal operator prox_f of f is

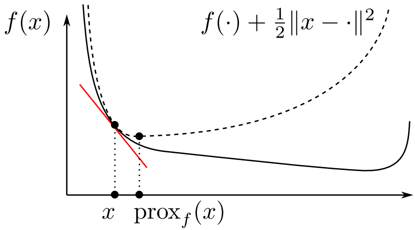
$$\begin{aligned}\mathrm{prox}_f &: \mathcal{X} \to \mathcal{X} \\ x &\mapsto \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2}\|y - x\|^2 \right\}.\end{aligned}$$

Non-differentiable optimization: proximal methods

Definition (The Proximal (Point) Operator)

For $f : \mathcal{X} \to \mathbb{R}$ convex, proximal operator prox_f of f is

$$\begin{aligned} \mathrm{prox}_f &: \mathcal{X} \to \mathcal{X} \\ & x \mapsto \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2} \|y - x\|^2 \right\}. \end{aligned}$$

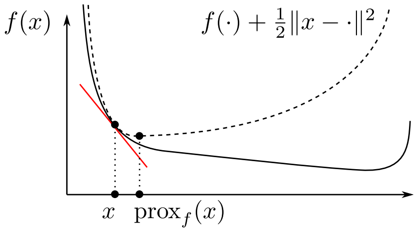


Non-differentiable optimization: proximal methods

Definition (The Proximal (Point) Operator)

For $f : \mathcal{X} \to \mathbb{R}$ convex, proximal operator prox_f of f is

$$\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$$
$$x \mapsto \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2}\|y - x\|^2 \right\}.$$



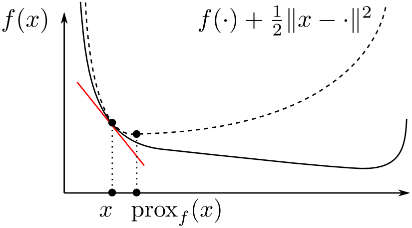
Remark: proximal point operator is *single-valued*.

Non-differentiable optimization: proximal methods

Definition (The Proximal (Point) Operator)

For $f : \mathcal{X} \to \mathbb{R}$ convex, proximal operator prox_f of f is

$$\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$$
$$x \mapsto \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2}\|y - x\|^2 \right\}.$$



Remark: proximal point operator is *single-valued*. **Not obvious! See next!**

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^\mathsf{T} (y - x) \geq 0.$$

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^{\mathsf{T}} (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^T (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

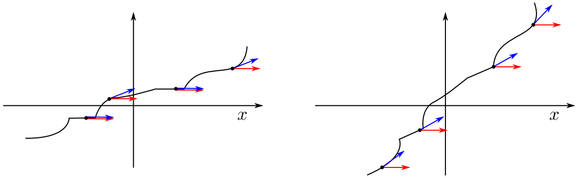


Figure: Monotone (left) and strictly monotone (right) operators.

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^{\mathsf{T}} (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

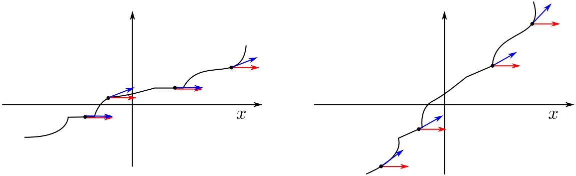


Figure: Monotone (left) and strictly monotone (right) operators.

Property (Inverse of a strictly monotone operator)

Inverse of the strictly monotone operator is single-valued.

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^{\mathsf{T}} (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

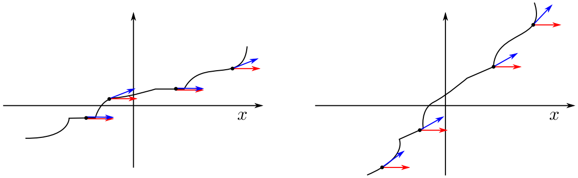


Figure: Monotone (left) and strictly monotone (right) operators.

Property (Inverse of a strictly monotone operator)

Inverse of the strictly monotone operator is single-valued.

Proof.

Proof by contradiction. Let $x \in \mathcal{X}$ with $\delta_x \in D(x)$. Suppose $\exists x'$ with $\delta_x \in D(x')$.

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^\mathsf{T} (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

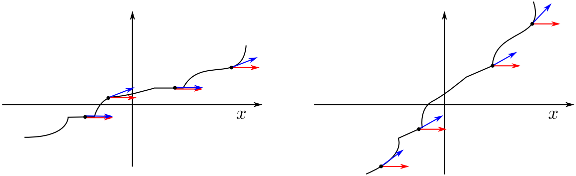


Figure: Monotone (left) and strictly monotone (right) operators.

Property (Inverse of a strictly monotone operator)

Inverse of the strictly monotone operator is single-valued.

Proof.

Proof by contradiction. Let $x \in \mathcal{X}$ with $\delta_x \in D(x)$. Suppose $\exists x'$ with $\delta_x \in D(x')$.
But, by strict monotonicity, $0 < (\delta_x - \delta_x)^\mathsf{T}(x - x') = 0$:

Non-differentiable optimization: proximal methods

Definition ((Strictly) Monotone operator)

Operator $D : \mathcal{X} \to 2^{\mathcal{X}}$ is monotone if

$$\forall x, y \in \mathcal{X}, D : d_x \in D(x), d_y \in D(y) \implies (d_y - d_x)^\mathsf{T} (y - x) \geq 0.$$

Strictly monotone: equality only for $x = y$.

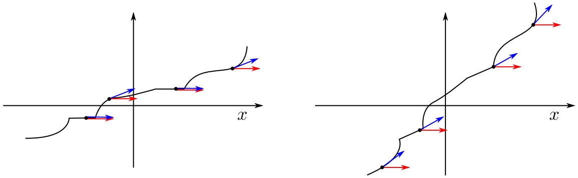


Figure: Monotone (left) and strictly monotone (right) operators.

Property (Inverse of a strictly monotone operator)

Inverse of the strictly monotone operator is single-valued.

Proof.

Proof by contradiction. Let $x \in \mathcal{X}$ with $\delta_x \in D(x)$. Suppose $\exists x'$ with $\delta_x \in D(x')$.
But, by strict monotonicity, $0 < (\delta_x - \delta_x)^\mathsf{T}(x - x') = 0$:
by contradiction, inverse of D is single-valued.

Non-differentiable optimization: proximal methods

Property

The operator prox_f *is single-valued (and thus well-defined).*

Non-differentiable optimization: proximal methods

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\ d_y \in \partial f(y)$,

$$(d_y - d_x)^\mathsf{T}(y - x) \geq 0 \ .$$

Non-differentiable optimization: proximal methods

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\ d_y \in \partial f(y)$,

$$(d_y - d_x)^\mathsf{T}(y - x) \geq 0 \ .$$

Follows from summing $f(x) \geq f(y) + d_y^\mathsf{T}(x - y)$ and $f(y) \geq f(x) + d_x^\mathsf{T}(y - x)$ (1st order relations).

Non-differentiable optimization: proximal methods

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\; d_y \in \partial f(y)$,

$$(d_y - d_x)^\mathsf{T}(y - x) \geq 0 \;\; .$$

Follows from summing $f(x) \geq f(y) + d_y^\mathsf{T}(x - y)$ and $f(y) \geq f(x) + d_x^\mathsf{T}(y - x)$ (1st order relations).
Implies $I + \partial f$ *strictly monotone* operator:

$$((y + d_y) - (x + d_x))^\mathsf{T}(y - x) = (d_y - d_x)^\mathsf{T}(y - x) + \|y - x\|^2 > 0 \;\; .$$

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\ d_y \in \partial f(y)$,

$$(d_y - d_x)^{\mathsf{T}}(y - x) \geq 0 \ .$$

Follows from summing $f(x) \geq f(y) + d_y^{\mathsf{T}}(x - y)$ and $f(y) \geq f(x) + d_x^{\mathsf{T}}(y - x)$ (1st order relations).

Implies $I + \partial f$ *strictly monotone* operator:

$$((y + d_y) - (x + d_x))^{\mathsf{T}}(y - x) = (d_y - d_x)^{\mathsf{T}}(y - x) + \|y - x\|^2 > 0 \ .$$

For $y \in \mathrm{prox}_f(x)$ ($= \mathrm{argmin}_z\, f(z) + \frac{1}{2}\|z - x\|^2$), 1st order optimality says

$$0 \in \partial f(y) + y - x = (I + \partial f)(y) - x \quad \Leftrightarrow \quad y \in (I + \partial f)^{-1}(x).$$

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\ d_y \in \partial f(y)$,

$$(d_y - d_x)^\mathsf{T}(y - x) \geq 0 \ .$$

Follows from summing $f(x) \geq f(y) + d_y^\mathsf{T}(x - y)$ and $f(y) \geq f(x) + d_x^\mathsf{T}(y - x)$ (1st order relations).

Implies $I + \partial f$ *strictly monotone* operator:

$$((y + d_y) - (x + d_x))^\mathsf{T}(y - x) = (d_y - d_x)^\mathsf{T}(y - x) + \|y - x\|^2 > 0 \ .$$

For $y \in \mathrm{prox}_f(x)$ ($= \mathrm{argmin}_z\, f(z) + \frac{1}{2}\|z - x\|^2$), 1st order optimality says

$$0 \in \partial f(y) + y - x = (I + \partial f)(y) - x \quad \Leftrightarrow \quad y \in (I + \partial f)^{-1}(x).$$

But inverse of strictly monotone $I + \partial f$ single-valued! $\square$

Non-differentiable optimization: proximal methods

Property

The operator prox_f *is single-valued (and thus well-defined).*

Proof.

Idea. ∂f is a *monotone operator*: $\forall d_x \in \partial f(x),\ d_y \in \partial f(y)$,

$$(d_y - d_x)^{\mathsf{T}}(y - x) \geq 0 \ .$$

Follows from summing $f(x) \geq f(y) + d_y^{\mathsf{T}}(x - y)$ and $f(y) \geq f(x) + d_x^{\mathsf{T}}(y - x)$ (1st order relations).
Implies $I + \partial f$ *strictly monotone* operator:

$$((y + d_y) - (x + d_x))^{\mathsf{T}}(y - x) = (d_y - d_x)^{\mathsf{T}}(y - x) + \|y - x\|^2 > 0 \ .$$

For $y \in \mathrm{prox}_f(x)$ $(= \mathrm{argmin}_z\, f(z) + \frac{1}{2}\|z - x\|^2)$, 1st order optimality says

$$0 \in \partial f(y) + y - x = (I + \partial f)(y) - x \quad \Leftrightarrow \quad y \in (I + \partial f)^{-1}(x).$$

But inverse of strictly monotone $I + \partial f$ single-valued! $\square$

Consequence. Uniqueness of prox_f makes optimization simpler: f may have multiple minima, $\mathrm{prox}_f(x)$ always unique.

Remark (Properties of prox_f)

For $\lambda > 0$,

$$\mathrm{prox}_{\lambda f}(x) = \underset{y \in \mathcal{X}}{\mathrm{argmin}} \left\{ f(y) + \frac{1}{2\lambda} \|x - y\|^2 \right\}.$$

Non-differentiable optimization: proximal methods

Remark (Properties of prox_f)

For $\lambda > 0$,

$$\mathrm{prox}_{\lambda f}(x) = \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2\lambda} \|x - y\|^2 \right\}.$$

For differentiable f,

$$y = \mathrm{prox}_{\lambda f}(x) = x - \lambda \nabla f(y) \iff y + \nabla f(y) = x.$$

Remark (Properties of prox_f)

For $\lambda > 0$,

$$\mathrm{prox}_{\lambda f}(x) = \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2\lambda} \|x - y\|^2 \right\}.$$

For differentiable f,

$$y = \mathrm{prox}_{\lambda f}(x) = x - \lambda \nabla f(y) \iff y + \nabla f(y) = x.$$

Consequence. *Iterating prox_f (from x to y) resembles "backward gradient ascent": if started from y, step along gradient at destination point points to starting point (with λ the step size).*

Remark (Properties of prox_f)

For $\lambda > 0$,

$$\mathrm{prox}_{\lambda f}(x) = \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2\lambda} \|x - y\|^2 \right\}.$$

For differentiable f,

$$y = \mathrm{prox}_{\lambda f}(x) = x - \lambda \nabla f(y) \iff y + \nabla f(y) = x.$$

Consequence. *Iterating prox_f (from x to y) resembles "backward gradient ascent": if started from y, step along gradient at destination point points to starting point (with λ the step size).*

Still for differentiable f,

$$\nabla \left(f(y) + \frac{1}{2\lambda} \|y - x\|^2 \right) = \nabla f(y) + \frac{1}{\lambda}(x - y).$$

Non-differentiable optimization: proximal methods

Remark (Properties of prox_f)

For $\lambda > 0$,

$$\mathrm{prox}_{\lambda f}(x) = \operatorname*{argmin}_{y \in \mathcal{X}} \left\{ f(y) + \frac{1}{2\lambda} \|x - y\|^2 \right\}.$$

For differentiable f,

$$y = \mathrm{prox}_{\lambda f}(x) = x - \lambda \nabla f(y) \iff y + \nabla f(y) = x.$$

Consequence. *Iterating prox_f (from x to y) resembles "backward gradient ascent": if started from y, step along gradient at destination point points to starting point (with λ the step size).*

Still for differentiable f,

$$\nabla \left(f(y) + \frac{1}{2\lambda} \|y - x\|^2 \right) = \nabla f(y) + \frac{1}{\lambda}(x - y).$$

Thus, at $y = x$, f and $f + \frac{1}{2\lambda}\|x - \cdot\|^2$ have same value and gradient: prox_f minimizes "local approximation" of f.

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the fixed points of prox_f*:*

$$x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the fixed points of prox_f*:*

$$x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Proof.

Follows from:

$$\begin{aligned} x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) &\Leftrightarrow 0 \in \partial f(x^\star) \\ &\Leftrightarrow 0 \in \partial f(x^\star) + (x^\star - x^\star) \\ &\Leftrightarrow x^\star = \mathrm{prox}_f(x^\star) \end{aligned}$$

(last line from $x^\star = \mathrm{prox}_f(x^\star) \implies x^\star \in \arg\min_x f(x)$). $\square$

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the fixed points of prox_f:

$$x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Proof.

Follows from:

$$\begin{aligned} x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x) &\Leftrightarrow 0 \in \partial f(x^\star) \\ &\Leftrightarrow 0 \in \partial f(x^\star) + (x^\star - x^\star) \\ &\Leftrightarrow x^\star = \mathrm{prox}_f(x^\star) \end{aligned}$$

(last line from $x^\star = \mathrm{prox}_f(x^\star) \implies x^\star \in \arg\min_x f(x)$). $\square$

Consequence: Suggests that *fixed-point algorithm* $x_{k+1} = \mathrm{prox}_f(x_k)$ converges to minimum of f.

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the fixed points of prox_f*:*

$$x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Proof.

Follows from:

$$\begin{aligned} x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) &\Leftrightarrow 0 \in \partial f(x^\star) \\ &\Leftrightarrow 0 \in \partial f(x^\star) + (x^\star - x^\star) \\ &\Leftrightarrow x^\star = \mathrm{prox}_f(x^\star) \end{aligned}$$

(last line from $x^\star = \mathrm{prox}_f(x^\star) \implies x^\star \in \arg\min_x f(x)$). □

Consequence: Suggests that *fixed-point algorithm* $x_{k+1} = \mathrm{prox}_f(x_k)$ converges to minimum of f. **But... does it converge?**

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the fixed points of prox_f:

$$x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Proof.

Follows from:

$$\begin{aligned} x^\star \in \operatorname*{argmin}_{x \in \mathcal{X}} f(x) &\Leftrightarrow 0 \in \partial f(x^\star) \\ &\Leftrightarrow 0 \in \partial f(x^\star) + (x^\star - x^\star) \\ &\Leftrightarrow x^\star = \mathrm{prox}_f(x^\star) \end{aligned}$$

(last line from $x^\star = \mathrm{prox}_f(x^\star) \implies x^\star \in \arg\min_x f(x)$). □

Consequence: Suggests that *fixed-point algorithm* $x_{k+1} = \mathrm{prox}_f(x_k)$ converges to minimum of f. **But... does it converge?**

- prox_f unfortunately **not** contractive (i.e., α-Lipschitz with $\alpha \in (0, 1)$ so that $\|x_{k+1} - x^\star\| \leq \alpha \|x_k - x^\star\|$)

Non-differentiable optimization: proximal methods

Key property:

Property (Proximal fixed-points and minimizers)

Minimizers of f are the *fixed points of* prox_f*:*

$$x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) \Leftrightarrow 0 \in \partial f(x^\star) \Leftrightarrow x^\star = \mathrm{prox}_f(x^\star).$$

Proof.

Follows from:

$$\begin{aligned} x^\star \in \operatorname*{argmin}_{x\in\mathcal{X}} f(x) &\Leftrightarrow 0 \in \partial f(x^\star) \\ &\Leftrightarrow 0 \in \partial f(x^\star) + (x^\star - x^\star) \\ &\Leftrightarrow x^\star = \mathrm{prox}_f(x^\star) \end{aligned}$$

(last line from $x^\star = \mathrm{prox}_f(x^\star) \implies x^\star \in \arg\min_x f(x)$). $\square$

Consequence: Suggests that *fixed-point algorithm* $x_{k+1} = \mathrm{prox}_f(x_k)$ converges to minimum of f. **But... does it converge?**

- prox_f unfortunately **not** contractive (i.e., α-Lipschitz with $\alpha \in (0,1)$ so that $\|x_{k+1} - x^\star\| \le \alpha\|x_k - x^\star\|$)
- but prox_f *firmly non-expansive*!

Non-differentiable optimization: proximal methods

Definition (Non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *non-expansive* if $\forall x, y \in \mathcal{X}$,

$$\|g(x) - g(y)\| \leq \|x - y\|.$$

Non-differentiable optimization: proximal methods

Definition (Non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *non-expansive* if $\forall x, y \in \mathcal{X}$,

$$\|g(x) - g(y)\| \leq \|x - y\|.$$

i.e., g **is** 1-**Lipschitz**.

Non-differentiable optimization: proximal methods

Definition (Non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *non-expansive* if $\forall x, y \in \mathcal{X}$,

$$\|g(x) - g(y)\| \le \|x - y\|.$$

i.e., g **is** 1-**Lipschitz**.

Definition (Firm non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *firmly non-expansive* if $\exists G : \mathcal{X} \to \mathcal{X}$ non-expansive with $g = \frac{1}{2}(I + G)$.

Non-differentiable optimization: proximal methods

Definition (Non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *non-expansive* if $\forall x, y \in \mathcal{X}$,

$$\|g(x) - g(y)\| \leq \|x - y\|.$$

i.e., g **is** 1-**Lipschitz**.

Definition (Firm non-expansiveness)

$g : \mathcal{X} \to \mathcal{X}$ *firmly non-expansive* if $\exists G : \mathcal{X} \to \mathcal{X}$ non-expansive with $g = \frac{1}{2}(I + G)$.

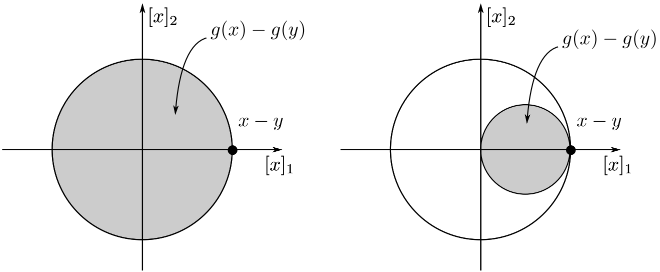


Figure: Non-expansive g (left) and firmly non-expansive g (right).

Non-differentiable optimization: proximal methods

Theorem

For convex f, $\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$, $x \mapsto \mathsf{argmin}_y\, f(y) + \frac{1}{2}\|x - y\|^2$ firmly non-expansive.

Non-differentiable optimization: proximal methods

Theorem

For convex f, $\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$, $x \mapsto \operatorname{argmin}_y f(y) + \frac{1}{2}\|x - y\|^2$ firmly non-expansive.

Proof.

Idea: Prove that $2\mathrm{prox}_f - I$ non-expansive, i.e., $\forall x, y \in \mathcal{X}$,

$$
\begin{aligned}
& \|(2\mathrm{prox}_f(x) - x) - (2\mathrm{prox}_f(y) - y)\|^2 \leq \|x - y\|^2 \\
\Leftrightarrow \quad & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^{\mathsf{T}}(x - y) \leq 0.
\end{aligned}
$$

Non-differentiable optimization: proximal methods

Theorem

For convex f, $\operatorname{prox}_f : \mathcal{X} \to \mathcal{X}$, $x \mapsto \operatorname{argmin}_y f(y) + \frac{1}{2}\|x - y\|^2$ firmly non-expansive.

Proof.

Idea: Prove that $2\operatorname{prox}_f - I$ non-expansive, i.e., $\forall x, y \in \mathcal{X}$,

$$\begin{aligned} & \|(2\operatorname{prox}_f(x) - x) - (2\operatorname{prox}_f(y) - y)\|^2 \leq \|x - y\|^2 \\ \Leftrightarrow \quad & \|\operatorname{prox}_f(x) - \operatorname{prox}_f(y)\|^2 - (\operatorname{prox}_f(x) - \operatorname{prox}_f(y))^{\mathsf{T}}(x - y) \leq 0. \end{aligned}$$

For this, recall ∂f is monotone: for $a = \operatorname{prox}_f(x)$ and $b = \operatorname{prox}_f(y)$, then

$$x - a \in \partial f(a) \quad \text{and} \quad y - b \in \partial f(b).$$

Non-differentiable optimization: proximal methods

Theorem

For convex f*,* $\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$*,* $x \mapsto \operatorname{argmin}_y f(y) + \frac{1}{2}\|x-y\|^2$ *firmly non-expansive.*

Proof.

Idea: Prove that $2\mathrm{prox}_f - I$ non-expansive, i.e., $\forall x, y \in \mathcal{X}$,

$$\begin{aligned} & \|(2\mathrm{prox}_f(x) - x) - (2\mathrm{prox}_f(y) - y)\|^2 \leq \|x-y\|^2 \\ \Leftrightarrow \quad & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^{\mathsf{T}}(x-y) \leq 0. \end{aligned}$$

For this, recall ∂f is monotone: for $a = \mathrm{prox}_f(x)$ and $b = \mathrm{prox}_f(y)$, then

$$x - a \in \partial f(a) \quad \text{and} \quad y - b \in \partial f(b).$$

Thus

$$((x - \mathrm{prox}_f(x)) - (y - \mathrm{prox}_f(y)))^{\mathsf{T}}(\mathrm{prox}_f(x) - \mathrm{prox}_f(y)) \geq 0.$$

Non-differentiable optimization: proximal methods

Theorem

For convex f, $\mathrm{prox}_f : \mathcal{X} \to \mathcal{X}$, $x \mapsto \mathrm{argmin}_y\, f(y) + \frac{1}{2}\|x-y\|^2$ firmly non-expansive.

Proof.

Idea: Prove that $2\mathrm{prox}_f - I$ non-expansive, i.e., $\forall x, y \in \mathcal{X}$,

$$\begin{aligned} & \|(2\mathrm{prox}_f(x) - x) - (2\mathrm{prox}_f(y) - y)\|^2 \leq \|x-y\|^2 \\ \Leftrightarrow \quad & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^{\mathsf{T}}(x-y) \leq 0. \end{aligned}$$

For this, recall ∂f is monotone: for $a = \mathrm{prox}_f(x)$ and $b = \mathrm{prox}_f(y)$, then

$$x - a \in \partial f(a) \quad \text{and} \quad y - b \in \partial f(b).$$

Thus

$$((x - \mathrm{prox}_f(x)) - (y - \mathrm{prox}_f(y)))^{\mathsf{T}}(\mathrm{prox}_f(x) - \mathrm{prox}_f(y)) \geq 0.$$

Implies

$$(\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^{\mathsf{T}}(x-y) \geq \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 \geq 0 \ .$$

$\square$

Non-differentiable optimization: proximal methods

Main property:

Theorem (The Proximal Point Algorithm)

For $f : \mathcal{X} \to \mathbb{R}$ *convex,* $x_1 \in \mathcal{X}$*, let*

$$x_{k+1} = \mathrm{prox}_f(x_k), \quad \forall k \geq 1.$$

Then $x_k \to x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} \{f(x)\}$.

Non-differentiable optimization: proximal methods

Main property:

Theorem (The Proximal Point Algorithm)

For $f : \mathcal{X} \to \mathbb{R}$ convex, $x_1 \in \mathcal{X}$, let

$$x_{k+1} = \mathrm{prox}_f(x_k), \quad \forall k \geq 1.$$

Then $x_k \to x^\star \in \mathrm{argmin}_{x \in \mathcal{X}}\{f(x)\}$.

Proof.

$$\begin{aligned}
&\|x_{k+1} - x_k\|^2 \\
&= \|\mathrm{prox}_f(x_k) - x_k\|^2 \\
&= \|(\mathrm{prox}_f(x_k) - x_k) - (\mathrm{prox}_f(x^\star) - x^\star)\|^2 \\
&= \|\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\|^2 + \|x_k - x^\star\|^2 - 2\left(\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\right)^{\mathsf{T}}(x_k - x^\star) \\
&\leq \|x_k - x^\star\|^2 - \|\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\|^2.
\end{aligned}$$

Non-differentiable optimization: proximal methods

Main property:

Theorem (The Proximal Point Algorithm)

For $f : \mathcal{X} \to \mathbb{R}$ convex, $x_1 \in \mathcal{X}$, let

$$x_{k+1} = \mathrm{prox}_f(x_k), \quad \forall k \geq 1.$$

Then $x_k \to x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} \{f(x)\}$.

Proof.

$$\begin{aligned}
&\|x_{k+1} - x_k\|^2 \\
&= \|\mathrm{prox}_f(x_k) - x_k\|^2 \\
&= \|(\mathrm{prox}_f(x_k) - x_k) - (\mathrm{prox}_f(x^\star) - x^\star)\|^2 \\
&= \|\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\|^2 + \|x_k - x^\star\|^2 - 2\left(\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\right)^\mathsf{T}(x_k - x^\star) \\
&\leq \|x_k - x^\star\|^2 - \|\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\|^2.
\end{aligned}$$

Last inequality uses *firm non-expansiveness* of prox_f:

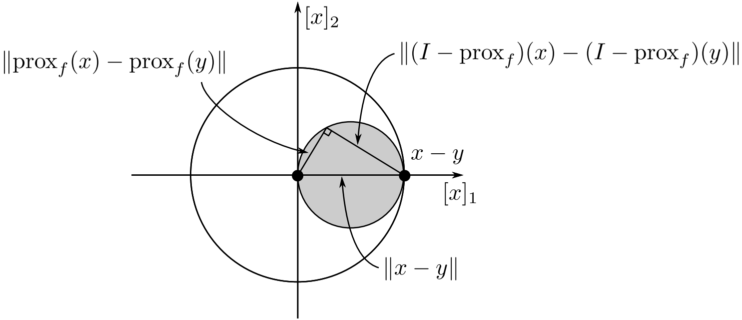
$$(\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star))^\mathsf{T}(x_k - x^\star) \geq \|\mathrm{prox}_f(x_k) - \mathrm{prox}_f(x^\star)\|^2 \geq 0 \ .$$

$\square$

Non-differentiable optimization: proximal methods

Proof.

Geometric interpretation:



$\square$

Non-differentiable optimization: proximal methods

Proof.

Recall now (non-expansiveness equivalence):

$$
\begin{aligned}
&\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) \le 0 \\
\iff & 2\,\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - 2(\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) + \|x-y\|^2 \le \|x-y\|^2 \\
\iff & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 + \|(I - \mathrm{prox}_f)(y) - (I - \mathrm{prox}_f)(x)\|^2 \le \|x-y\|^2
\end{aligned}
$$

Non-differentiable optimization: proximal methods

Proof.

Recall now (non-expansiveness equivalence):

$$
\begin{aligned}
& \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) \le 0 \\
\iff & 2\,\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - 2(\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) + \|x-y\|^2 \le \|x-y\|^2 \\
\iff & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 + \|(I - \mathrm{prox}_f)(y) - (I - \mathrm{prox}_f)(x)\|^2 \le \|x-y\|^2
\end{aligned}
$$

In particular

$$
\begin{gathered}
\|x_{k+1} - x^\star\|^2 + \|x_{k+1} - x_k\|^2 \le \|x_k - x^\star\|^2 \\
\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 \le \|x-y\|^2 .
\end{gathered}
$$

Non-differentiable optimization: proximal methods

Proof.

Recall now (non-expansiveness equivalence):

$$\begin{aligned}
&\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) \le 0 \\
\iff &2\,\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - 2(\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) + \|x-y\|^2 \le \|x-y\|^2 \\
\iff &\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 + \|(I - \mathrm{prox}_f)(y) - (I - \mathrm{prox}_f)(x)\|^2 \le \|x-y\|^2
\end{aligned}$$

In particular

$$\|x_{k+1} - x^\star\|^2 + \|x_{k+1} - x_k\|^2 \le \|x_k - x^\star\|^2$$
$$\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 \le \|x-y\|^2 .$$

Summing over $k = 1, \ldots, K$:

$$K\|x_{K+1} - x_K\|^2 \le \|x_1 - x^\star\|^2 - \|x_{K+1} - x^\star\|^2 \le \|x_1 - x^\star\|^2$$

Non-differentiable optimization: proximal methods

Proof.

Recall now (non-expansiveness equivalence):

$$
\begin{aligned}
& \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - (\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) \le 0 \\
\iff & 2\,\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 - 2(\mathrm{prox}_f(x) - \mathrm{prox}_f(y))^\mathsf{T}(x-y) + \|x-y\|^2 \le \|x-y\|^2 \\
\iff & \|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 + \|(I - \mathrm{prox}_f)(y) - (I - \mathrm{prox}_f)(x)\|^2 \le \|x-y\|^2
\end{aligned}
$$

In particular

$$
\begin{aligned}
\|x_{k+1} - x^\star\|^2 + \|x_{k+1} - x_k\|^2 &\le \|x_k - x^\star\|^2 \\
\|\mathrm{prox}_f(x) - \mathrm{prox}_f(y)\|^2 &\le \|x-y\|^2 .
\end{aligned}
$$

Summing over $k = 1, \dots, K$:

$$
K\|x_{K+1} - x_K\|^2 \le \|x_1 - x^\star\|^2 - \|x_{K+1} - x^\star\|^2 \le \|x_1 - x^\star\|^2
$$

and thus

$$
\|x_{K+1} - x_K\| \le \frac{1}{\sqrt{K}}\|x_1 - x^\star\| \to 0
$$

as $K \to \infty$, i.e., $\|\mathrm{prox}_f(x_k) - x_k\| \to 0$. $\square$

Non-differentiable optimization: proximal methods

Remark (On proximal point algorithm)

- *does not need differentiable f, does not have step size constraint;*

Non-differentiable optimization: proximal methods

Remark (On proximal point algorithm)

- *does not need differentiable f, does not have step size constraint;*
- *one can change f in λf $(\lambda > 0)$: not affecting algorithm, but possibly performance;*

Non-differentiable optimization: proximal methods

Remark (On proximal point algorithm)

- *does not need differentiable f, does not have step size constraint;*
- *one can change f in λf ($\lambda > 0$): not affecting algorithm, but possibly performance;*
- **but** *2 main difficulties:*
  - prox_f *can be difficult to evaluate*

Non-differentiable optimization: proximal methods

Remark (On proximal point algorithm)

- *does not need differentiable f, does not have step size constraint;*
- *one can change f in λf $(\lambda > 0)$: not affecting algorithm, but possibly performance;*
- **but** *2 main difficulties:*
  - prox_f *can be difficult to evaluate*
  - *in worst case, sublinear convergence rate.*

Non-differentiable optimization: proximal methods

Remark (On proximal point algorithm)

- *does not need differentiable f, does not have step size constraint;*
- *one can change f in λf ($\lambda > 0$): not affecting algorithm, but possibly performance;*
- **but** *2 main difficulties:*
  - prox_f *can be difficult to evaluate*
  - *in worst case, sublinear convergence rate.*

Table of classical prox operators:

f	$\mathrm{prox}_f(x)$	$\nabla f(x)$ -
0	x	0
$\imath_\Omega(x)$	$P_\Omega(x)$	-
$\imath_{\mathbb{R}^n_+}(x)$	$\{\max([x]_i, 0)\}_{i=1}^N$	-
$\lambda\|x\|_1$	$\{\mathrm{sgn}([x]_i)\max(\|[x]_i\| - \lambda, 0)\}_{i=1}^n$	-
$\imath_{\{\bar{x}, A\bar{x}=y\}}(x)$	$x + A^\mathsf{T}(AA^\mathsf{T})^{-1}(y - Ax)$	-
$\frac{1}{2}\|Ax - y\|^2$	$(I_n + A^\mathsf{T}A)^{-1}(x + A^\mathsf{T}y)$	$A^\mathsf{T}(Ax - y)$
$x^\mathsf{T}A^\mathsf{T}y$	$x - A^\mathsf{T}y$	$A^\mathsf{T}y$
$\frac{1}{2}x^\mathsf{T}Ax$	$(I_n + A)^{-1}x$	Ax

Outline

Non-differentiable optimization: sum of two functions

Two-function optimization problem: for any convex f_1 and f_2,

$$\min_{x \in \mathcal{X}} f_1(x) + f_2(x)$$

Non-differentiable optimization: sum of two functions

Two-function optimization problem: for any convex f_1 and f_2,

$$\min_{x \in \mathcal{X}} f_1(x) + f_2(x)$$

Crucial example:

- $f_1(x) = \imath_\Omega(x)$ for convex $\Omega \subset \mathcal{X}$
- f_2 any convex function (our previous f).

Non-differentiable optimization: sum of two functions

Two-function optimization problem: for any convex f_1 and f_2,

$$\min_{x \in \mathcal{X}} f_1(x) + f_2(x)$$

Crucial example:

- $f_1(x) = \imath_\Omega(x)$ for convex $\Omega \subset \mathcal{X}$
- f_2 any convex function (our previous f).

Case of differentiable convex f_2: with L-Lipschitz gradient ∇f_2 (f_1 *only convex*).

Non-differentiable optimization: sum of two functions

Two-function optimization problem: for any convex f_1 and f_2,

$$\min_{x \in \mathcal{X}} f_1(x) + f_2(x)$$

Crucial example:

- $f_1(x) = \imath_\Omega(x)$ for convex $\Omega \subset \mathcal{X}$
- f_2 any convex function (our previous f).

Case of differentiable convex f_2: with L-Lipschitz gradient ∇f_2 (f_1 *only convex*).
Then:

$$\begin{aligned}
x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} \{f_1(x) + f_2(x)\} &\Leftrightarrow 0 \in \partial f_1(x^\star) + \nabla f_2(x^\star) \\
&\Leftrightarrow 0 \in \gamma \partial f_1(x^\star) + \gamma \nabla f_2(x^\star) \\
&\Leftrightarrow x^\star \in x^\star + \gamma \partial f_1(x^\star) + \gamma \nabla f_2(x^\star) \\
&\Leftrightarrow x^\star - \gamma \nabla f_2(x^\star) \in x^\star + \gamma \partial f_1(x^\star) \\
&\Leftrightarrow x^\star = \mathrm{prox}_{\gamma f_1}((I - \gamma \nabla f_2)(x^\star)) \\
&\Leftrightarrow x^\star = \left(\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)\right)(x^\star).
\end{aligned}$$

Non-differentiable optimization: sum of two functions

Two-function optimization problem: for any convex f_1 and f_2,

$$\min_{x\in\mathcal{X}} f_1(x) + f_2(x)$$

Crucial example:

- $f_1(x) = \imath_\Omega(x)$ for convex $\Omega \subset \mathcal{X}$
- f_2 any convex function (our previous f).

Case of differentiable convex f_2: with L-Lipschitz gradient ∇f_2 (f_1 *only convex*).
Then:

$$\begin{aligned}
x^\star \in \operatorname{argmin}_{x\in\mathcal{X}} \{f_1(x) + f_2(x)\} &\Leftrightarrow 0 \in \partial f_1(x^\star) + \nabla f_2(x^\star)\\
&\Leftrightarrow 0 \in \gamma\partial f_1(x^\star) + \gamma\nabla f_2(x^\star)\\
&\Leftrightarrow x^\star \in x^\star + \gamma\partial f_1(x^\star) + \gamma\nabla f_2(x^\star)\\
&\Leftrightarrow x^\star - \gamma\nabla f_2(x^\star) \in x^\star + \gamma\partial f_1(x^\star)\\
&\Leftrightarrow x^\star = \mathrm{prox}_{\gamma f_1}((I - \gamma\nabla f_2)(x^\star))\\
&\Leftrightarrow x^\star = \left(\mathrm{prox}_{\gamma f_1} \circ (I - \gamma\nabla f_2)\right)(x^\star).
\end{aligned}$$

Consequence: equivalent to finding fixed-point for:

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma\nabla f_2).$$

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ *seems artificial.* **But**, *to ensure convergence of fixed-point algorithm,*

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive.

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ *seems artificial.* **But**, *to ensure convergence of fixed-point algorithm,*

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if** $\gamma < \frac{1}{L}$**!**

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ *seems artificial.* **But**, *to ensure convergence of fixed-point algorithm,*

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if** $\gamma < \frac{1}{L}$**!**

Theorem (Forward-Backward Splitting algorithm)

For $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ *convex with* f_2 *differentiable and with L-Lipschitz gradient, let*

$$\begin{aligned} x_1 &\in \mathcal{X} \\ x_{k+1} &= \mathrm{prox}_{\gamma f_1}\left(x_k - \gamma \nabla f_2(x_k)\right), \quad k \geq 1. \end{aligned}$$

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ seems artificial. **But**, *to ensure convergence of fixed-point algorithm,*

$$\operatorname{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if $\gamma < \frac{1}{L}$!**

Theorem (Forward-Backward Splitting algorithm)

For $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ convex with f_2 differentiable and with L-Lipschitz gradient, let

$$\begin{aligned} x_1 &\in \mathcal{X} \\ x_{k+1} &= \operatorname{prox}_{\gamma f_1}\left(x_k - \gamma \nabla f_2(x_k)\right), \quad k \geq 1. \end{aligned}$$

Then, as $k \to \infty$,

$$x_k \to x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} \left\{ f_1(x) + f_2(x) \right\}.$$

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ seems artificial. **But**, *to ensure convergence of fixed-point algorithm,*

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if $\gamma < \frac{1}{L}$!**

Theorem (Forward-Backward Splitting algorithm)

For $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ convex with f_2 differentiable and with L-Lipschitz gradient, let

$$\begin{aligned} x_1 &\in \mathcal{X} \\ x_{k+1} &= \mathrm{prox}_{\gamma f_1}\left(x_k - \gamma \nabla f_2(x_k)\right), \quad k \geq 1. \end{aligned}$$

Then, as $k \to \infty$,

$$x_k \to x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} \left\{ f_1(x) + f_2(x) \right\}.$$

Why forward-backward splitting?

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ seems artificial. **But**, *to ensure convergence of fixed-point algorithm,*

$$\operatorname{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if $\gamma < \frac{1}{L}$!**

Theorem (Forward-Backward Splitting algorithm)

For $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ convex with f_2 differentiable and with L-Lipschitz gradient, let

$$\begin{aligned} x_1 &\in \mathcal{X} \\ x_{k+1} &= \operatorname{prox}_{\gamma f_1} (x_k - \gamma \nabla f_2(x_k)), \quad k \geq 1. \end{aligned}$$

Then, as $k \to \infty$,

$$x_k \to x^\star \in \operatorname{argmin}_{x \in \mathcal{X}} \{f_1(x) + f_2(x)\}.$$

Why forward-backward splitting? Two-step approach:

1. move from x_k to $\tilde{x}_k \equiv x_k - \gamma \nabla f_2(x_k)$, i.e., gradient descent step on f_2 (*forward* progression to minimizing f_2);

Non-differentiable optimization: sum of two functions

Remark (On parameter γ)

γ *seems artificial.* **But**, *to ensure convergence of fixed-point algorithm,*

$$\mathrm{prox}_{\gamma f_1} \circ (I - \gamma \nabla f_2)$$

must be firmly non-expansive. **Only true if $\gamma < \frac{1}{L}$!**

Theorem (Forward-Backward Splitting algorithm)

For $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ convex with f_2 differentiable and with L-Lipschitz gradient, let

$$\begin{aligned} x_1 &\in \mathcal{X} \\ x_{k+1} &= \mathrm{prox}_{\gamma f_1}\left(x_k - \gamma \nabla f_2(x_k)\right), \quad k \geq 1. \end{aligned}$$

Then, as $k \to \infty$,

$$x_k \to x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} \left\{ f_1(x) + f_2(x) \right\}.$$

Why forward-backward splitting? Two-step approach:

1. move from x_k to $\tilde{x}_k \equiv x_k - \gamma \nabla f_2(x_k)$, i.e., gradient descent step on f_2 (*forward* progression to minimizing f_2);
2. move from $\tilde{x}_k$ to $x_{k+1} = \mathrm{prox}_{\gamma f_1}(\tilde{x}_k)$, i.e., "*backward*" move from $\tilde{x}_k$ to $x_{k+1} = (I + \partial f_1)^{-1}(\tilde{x}_k)$.

Non-differentiable optimization: sum of two functions

Remark (Forward-Backward Splitting in Practice)

Very convenient in practice to minimize convex differentiable $f = f_2$ under convex constraints given by f_1,

Non-differentiable optimization: sum of two functions

Remark (Forward-Backward Splitting in Practice)

Very convenient in practice to minimize convex differentiable $f = f_2$ under convex constraints given by f_1, e.g.,

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

Non-differentiable optimization: sum of two functions

Remark (Forward-Backward Splitting in Practice)

Very convenient in practice to minimize convex differentiable $f = f_2$ under convex constraints given by f_1, e.g.,

$$\min_{x\in\Omega} f(x) \quad \Leftrightarrow \quad \min_{x\in\mathcal{X}} \imath_\Omega(x) + f(x)$$

Non-differentiable optimization: sum of two functions

Remark (Forward-Backward Splitting in Practice)

Very convenient in practice to minimize convex differentiable $f = f_2$ under convex constraints given by f_1, e.g.,

$$\min_{x \in \Omega} f(x) \quad \Leftrightarrow \quad \min_{x \in \mathcal{X}} \imath_\Omega(x) + f(x)$$

Main advantage: *constrained minimization turned into a much simpler unconstrained minimization of two functions.*

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2:

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x)$$

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x) \quad \Leftrightarrow \quad x = 2\mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - (2\tilde{x} - x)$$

where $\tilde{x} \equiv \mathrm{prox}_{\gamma f_1}(x)$ (i.e., $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x) \quad \Leftrightarrow \quad x = 2\mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - (2\tilde{x} - x)$$

where $\tilde{x} \equiv \mathrm{prox}_{\gamma f_1}(x)$ (i.e., $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).
Further equivalent to

$$\begin{aligned}
&\Leftrightarrow 0 = \mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - \tilde{x} \\
&\Leftrightarrow 2\tilde{x} - x \in (\gamma \partial f_2 + I)(\tilde{x}) \\
&\Leftrightarrow \tilde{x} - x \in \gamma \partial f_2(\tilde{x}) \\
&\Leftrightarrow 0 \in \gamma \partial f_1(x) + \gamma \partial f_2(x)
\end{aligned}$$

(last line uses $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x) \quad \Leftrightarrow \quad x = 2\mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - (2\tilde{x} - x)$$

where $\tilde{x} \equiv \mathrm{prox}_{\gamma f_1}(x)$ (i.e., $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).
Further equivalent to

$$\begin{aligned}
&\Leftrightarrow 0 = \mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - \tilde{x} \\
&\Leftrightarrow 2\tilde{x} - x \in (\gamma \partial f_2 + I)(\tilde{x}) \\
&\Leftrightarrow \tilde{x} - x \in \gamma \partial f_2(\tilde{x}) \\
&\Leftrightarrow 0 \in \gamma \partial f_1(x) + \gamma \partial f_2(x)
\end{aligned}$$

(last line uses $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).

Major issue: only *non-expansive* iterations;

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x) \quad \Leftrightarrow \quad x = 2\mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - (2\tilde{x} - x)$$

where $\tilde{x} \equiv \mathrm{prox}_{\gamma f_1}(x)$ (i.e., $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).
Further equivalent to

$$\begin{aligned}
&\Leftrightarrow 0 = \mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - \tilde{x} \\
&\Leftrightarrow 2\tilde{x} - x \in (\gamma \partial f_2 + I)(\tilde{x}) \\
&\Leftrightarrow \tilde{x} - x \in \gamma \partial f_2(\tilde{x}) \\
&\Leftrightarrow 0 \in \gamma \partial f_1(x) + \gamma \partial f_2(x)
\end{aligned}$$

(last line uses $x - \tilde{x} \in \gamma \partial f_1(\tilde{x})$).

Major issue: only *non-expansive* iterations; does not guarantee convergence.

Non-differentiable optimization: sum of two functions

Relaxing differentiable f_2: Proceeding as before, algorithm now iterates

$$(2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I).$$

Why? Follows from:

$$x = (2\mathrm{prox}_{\gamma f_2} - I) \circ (2\mathrm{prox}_{\gamma f_1} - I)(x) \quad \Leftrightarrow \quad x = 2\mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - (2\tilde{x} - x)$$

where $\tilde{x} \equiv \mathrm{prox}_{\gamma f_1}(x)$ (i.e., $x - \tilde{x} \in \gamma\partial f_1(\tilde{x})$).
Further equivalent to

$$\begin{aligned}
&\Leftrightarrow 0 = \mathrm{prox}_{\gamma f_2}(2\tilde{x} - x) - \tilde{x} \\
&\Leftrightarrow 2\tilde{x} - x \in (\gamma\partial f_2 + I)(\tilde{x}) \\
&\Leftrightarrow \tilde{x} - x \in \gamma\partial f_2(\tilde{x}) \\
&\Leftrightarrow 0 \in \gamma\partial f_1(x) + \gamma\partial f_2(x)
\end{aligned}$$

(last line uses $x - \tilde{x} \in \gamma\partial f_1(\tilde{x})$).

Major issue: only *non-expansive* iterations; does not guarantee convergence.

Solution: add extra $\rho \in (0, 1)$ in algorithm steps.

Non-differentiable optimization: sum of two functions

Theorem (Douglas-Rachford Splitting)

Let $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ *convex. For* $x_0 \in \mathcal{X}$*,* $\lambda > 0$*,* $\rho \in (0,1)$*, and* $k \geq 1$*, let*

$$\begin{aligned} \tilde{x}_k &= \mathrm{prox}_{\gamma f_1}(x_k) \\ x_{k+1} &= x_k + 2\rho \left(\mathrm{prox}_{\gamma f_2}(2\tilde{x}_k - x_k) - \tilde{x}_k \right). \end{aligned}$$

Non-differentiable optimization: sum of two functions

Theorem (Douglas-Rachford Splitting)

Let $f_1, f_2 : \mathcal{X} \to \mathbb{R}$ convex. For $x_0 \in \mathcal{X}$, $\lambda > 0$, $\rho \in (0, 1)$, and $k \geq 1$, let

$$\tilde{x}_k = \mathrm{prox}_{\gamma f_1}(x_k)$$
$$x_{k+1} = x_k + 2\rho \left(\mathrm{prox}_{\gamma f_2}(2\tilde{x}_k - x_k) - \tilde{x}_k\right).$$

Then, as $k \to \infty$,

$$x_k \to x^\star \in \mathrm{argmin}_{x \in \mathcal{X}} f_1(x) + f_2(x).$$

The End.