Optimization for Machine Learning

Clément Royer

CIMPA School "Control, Optimization and Model Reduction in Machine Learning"

February 24, 2025

About the instructor

Who I am: Clément Royer

- *Maître de conférences* at Dauphine since 2019.
- Research topics: Optimization and applications.
- Email: `clement.royer@lamsade.dauphine.fr`
- Webpage: https://www.lamsade.dauphine.fr/~croyer

About these lectures

A tour d'horizon

- Review main concepts in optimization;
- In perspective with machine learning applications.

Learning goals

- Identify characteristics of an ML problem;
- Have an optimization toolbox for ML;
- Know the theoretical underpinnings;
- Practical experience.

Outline

A warning

Optimization $\not\subset$ Machine Learning

- Optimization is a mathematical tool;
- Used in many areas: Economics, Chemistry, Physics, Social sciences,...
- Appears in other branches of (applied) mathematics: Linear Algebra, PDEs, Statistics, etc.

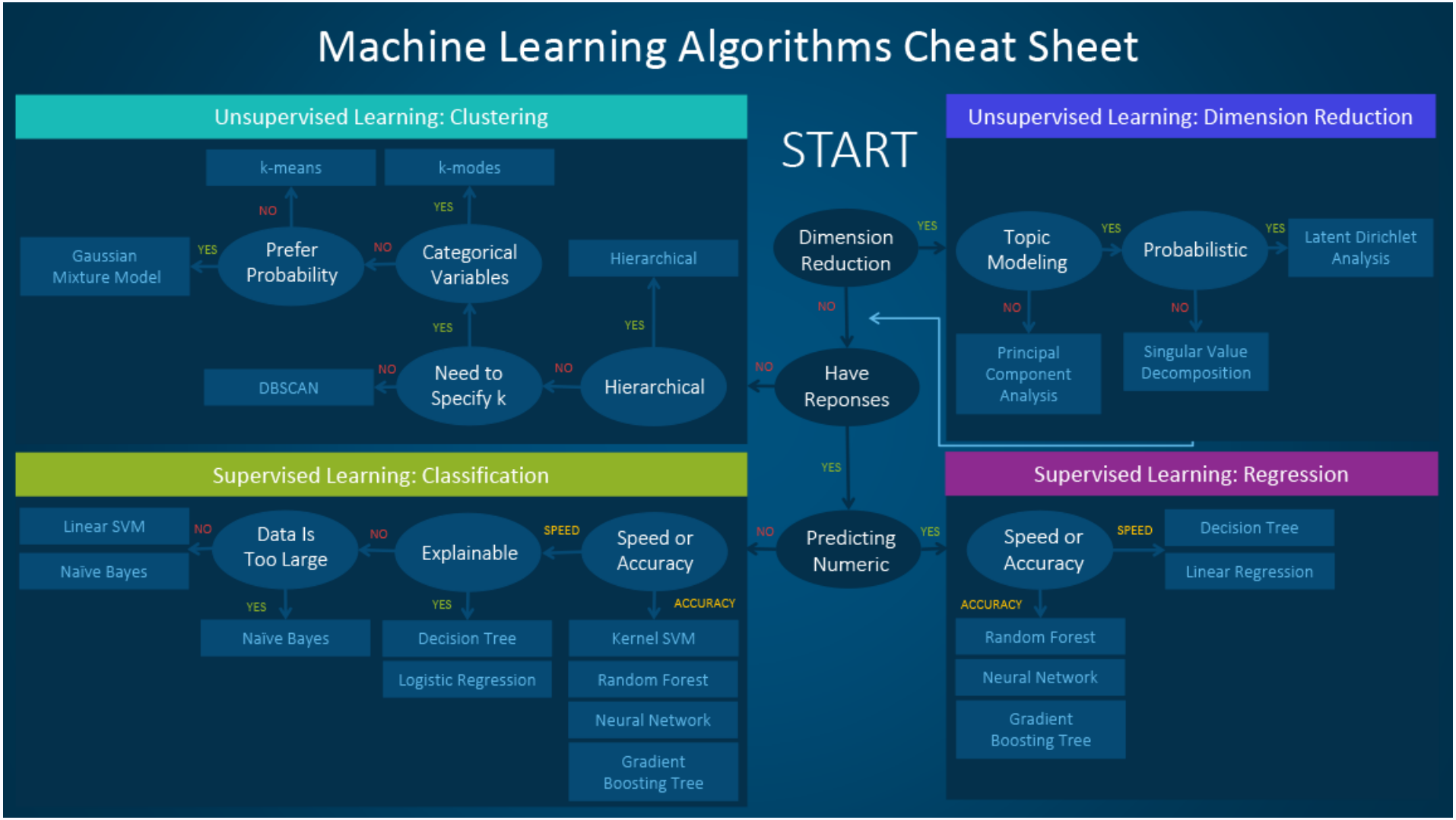
A warning

Optimization $\not\subset$ Machine Learning

- Optimization is a mathematical tool;
- Used in many areas: Economics, Chemistry, Physics, Social sciences,...
- Appears in other branches of (applied) mathematics: Linear Algebra, PDEs, Statistics, etc.

Machine Learning $\not\subset$ Optimization

- Optimization targets a certain problem;
- ML is not just about this problem;
- Other features of ML (data cleaning, hardware,...) will not appear in the optimization.



Source: https://blogs.sas.com/content/subconsciousmusings/2017/04/12/ machine-learning-algorithm-use/

Example: Document classification

Given: A dataset $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$.

- $\boldsymbol{x}_i$ is a feature vector in $\mathbb{R}^d$;
- y_i is a label.

Example: Document classification

Given: A dataset $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)\}$.

- $\boldsymbol{x}_i$ is a feature vector in $\mathbb{R}^d$;
- y_i is a label.

Example: text classification

Using d words for classification:

- $\boldsymbol{x}_i$ represents the words contained in a text document:

$$[\boldsymbol{x}_i]_j = \begin{cases} 1 & \text{if word } j \text{ is in document } i, \\ 0 & \text{otherwise.} \end{cases}$$

- y_i is equal to $+1$ if the document addresses a certain topic of interest, to -1 otherwise.

Prediction and classification

Learning process

- Given $\{(\boldsymbol{x}_i, y_i)\}_i$, discover a function $h : \mathbb{R}^d \to \mathbb{R}$ such that $h(\boldsymbol{x}_i) \approx y_i \ \forall i = 1, \ldots, n$.
- Choose the predictor function h among a set $\mathcal{H}$ parameterized by a vector $\boldsymbol{w} \in \mathbb{R}^d$: $\mathcal{H} = \left\{ h \mid h = h(\cdot; \boldsymbol{w}), \ \boldsymbol{w} \in \mathbb{R}^{\hat{d}} \right\}$;

Prediction and classification

Learning process

- Given $\{(\boldsymbol{x}_i, y_i)\}_i$, discover a function $h : \mathbb{R}^d \to \mathbb{R}$ such that $h(\boldsymbol{x}_i) \approx y_i \ \forall i = 1, \ldots, n$.
- Choose the predictor function h among a set $\mathcal{H}$ parameterized by a vector $\boldsymbol{w} \in \mathbb{R}^d$: $\mathcal{H} = \left\{ h \mid h = h(\cdot; \boldsymbol{w}), \ \boldsymbol{w} \in \mathbb{R}^{\hat{d}} \right\}$;

Linear model for text classification

- We seek a hyperplane in $\mathbb{R}^d$ separating the feature vectors associated with $y_i = +1$ and those associated with $y_i = -1$;
- This corresponds to a linear model $h(x) = \boldsymbol{x}^{\mathrm{T}} \boldsymbol{w}$, and we want to choose $\boldsymbol{w}$ such that:

$$\forall i = 1, \ldots, n, \qquad \begin{cases} \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} \geq 1 & \text{if } y_i = +1 \\ \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} \leq -1 & \text{if } y_i = -1. \end{cases}$$

Objective of the problem

An objective to optimize over

- Our goal is to penalize values of $\boldsymbol{w}$ for which $h(\boldsymbol{x}_i)$ does not predict y_i well enough.
- One possibility: the **hinge loss function**

$$\forall (h, y) \in \mathbb{R}^2, \quad \ell(h, y) = \max\{1 - yh, 0\}.$$

About the hinge loss

- $hy > 1 \Rightarrow \ell(h, y) = 0$: no penalty ($h$ and y are of the same sign, $|h| > 1$ so this is a good prediction);
- $hy < -1 \Rightarrow \ell(h, y) > 2$: large penalty ($h$ and y are of opposite sign and $|h| > 1$, this is a bad prediction);
- $|hy| \leq 1 \Rightarrow \ell(h, y) \in [0, 2]$: small penalty ($h$ and y can be of the same sign, but the value of $|h|$ makes the prediction less certain).

Optimization formulation

An optimization problem

$$\min_{\boldsymbol{u},v} \frac{1}{n} \sum_{i=1}^{n} \max\left\{1 - y_i(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}), 0\right\} \qquad .$$

Optimization formulation

An optimization problem

$$\min_{\boldsymbol{u},v} \frac{1}{n} \sum_{i=1}^{n} \max\left\{1 - y_i(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}), 0\right\} \qquad .$$

- Minimize the sum of the losses for all examples;

Optimization formulation

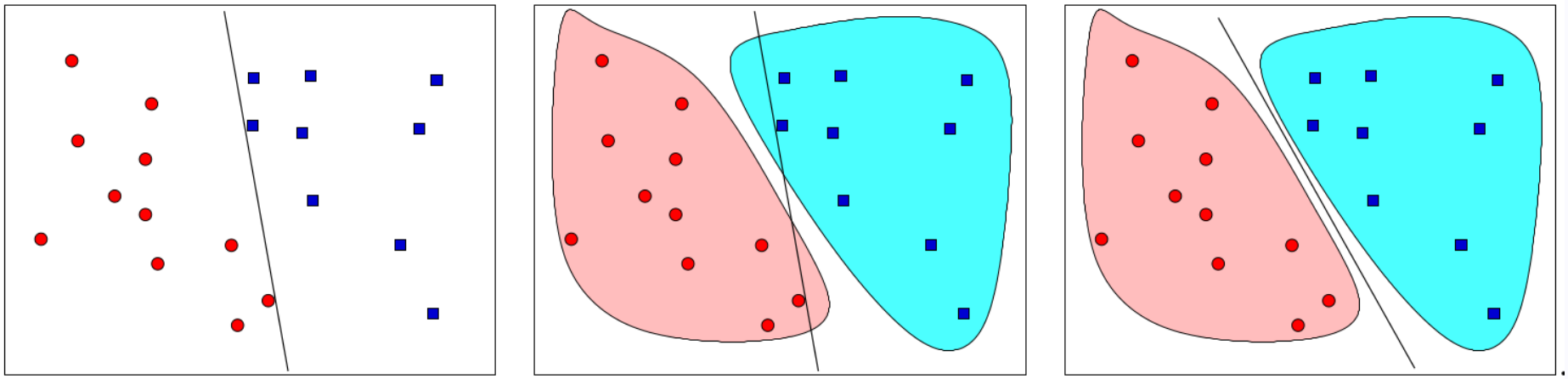
An optimization problem

$$\min_{\boldsymbol{u},v} \frac{1}{n} \sum_{i=1}^{n} \max\left\{1 - y_i(\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}), 0\right\} + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2.$$

for $\lambda \geq 0$.

- Minimize the sum of the losses for all examples;
- Regularizing term to promote small-norm solutions (more on that later).

Different solutions



Source: S. J. Wright, Optimization Algorithms for Data Analysis, 2018.

- Red/Blue dots: data points labeled +1/-1;
- Red/Blue clouds: distribution of the text documents;
- Two linear classifiers;
- Rightmost plot: maximal-margin solution.

Example: Logistic regression

Context

- Data set $\{(\boldsymbol{x}_i, y_i)\}_i$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, +1\}$;
- Goal: Classification through a linear classifier $\boldsymbol{x} \mapsto \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$;
- **Difference with SVM:** we want probabilities of belonging to a class!

Example: Logistic regression

Context

- Data set $\{(\boldsymbol{x}_i, y_i)\}_i$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, +1\}$;
- Goal: Classification through a linear classifier $\boldsymbol{x} \mapsto \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$;
- **Difference with SVM:** we want probabilities of belonging to a class!

A probabilistic measure

- We define an odds-like function

$$p(\boldsymbol{x}; \boldsymbol{w}) = (1 + e^{\boldsymbol{x}^{\mathrm{T}}\boldsymbol{w}})^{-1} \in (0, 1).$$

- The parameters $\boldsymbol{w}$ should be chosen such that

$$\begin{cases} p(\boldsymbol{x}_i; \boldsymbol{w}) \approx 1 & \text{if } y_i = +1; \\ p(\boldsymbol{x}_i; \boldsymbol{w}) \approx 0 & \text{if } y_i = -1. \end{cases}$$

Example: Logistic regression (2)

Towards an objective function

$$p(\boldsymbol{x}; \boldsymbol{w}) = (1 + e^{\boldsymbol{x}^{\mathrm{T}} \boldsymbol{w}})^{-1},$$

- Penalize cases where
  - $y_i = +1$ and $p(\boldsymbol{x}_i; \boldsymbol{w})$ is close to 0;
  - $y_i = -1$ and $p(\boldsymbol{x}_i; \boldsymbol{w})$ is close to 1;
- Use logarithm of the $p(\boldsymbol{x}_i; \boldsymbol{w})$ in the cost function:
  - Motivation: Statistical interpretation (joint distribution);
  - Mathematical interest for gradient calculations.

Example: Logistic regression (2)

Towards an objective function

$$p(\boldsymbol{x}; \boldsymbol{w}) = (1 + e^{\boldsymbol{x}^{\mathrm{T}} \boldsymbol{w}})^{-1},$$

- Penalize cases where
  - $y_i = +1$ and $p(\boldsymbol{x}_i; \boldsymbol{w})$ is close to 0;
  - $y_i = -1$ and $p(\boldsymbol{x}_i; \boldsymbol{w})$ is close to 1;
- Use logarithm of the $p(\boldsymbol{x}_i; \boldsymbol{w})$ in the cost function:
  - Motivation: Statistical interpretation (joint distribution);
  - Mathematical interest for gradient calculations.

Resulting function: logistic loss

$$f(\boldsymbol{w}) = \frac{1}{n} \left\{ \sum_{y_i=-1} \ln\left(1 + e^{-\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}}\right) + \sum_{y_i=+1} \ln\left(1 + e^{\boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}}\right) \right\}.$$

Example: Logistic regression (3)

Logistic regression problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{n}\left\{\sum_{y_i=-1} \ln\left(1+e^{-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}\right)+\sum_{y_i=+1} \ln\left(1+e^{\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}\right)\right\}$$

Example: Logistic regression (3)

Logistic regression problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{n}\left\{\sum_{y_i=-1} \ln\left(1+e^{-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}\right)+\sum_{y_i=+1} \ln\left(1+e^{\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}\right)\right\}$$

- The logistic loss is convex (but not strongly);
- To make it convex, possible to add a regularizing term $\frac{\mu}{2}\|\boldsymbol{w}\|^2$
  $\Rightarrow$ The problem becomes μ-strongly convex!

Example: Nonlinear regression

Context

- Data set $\{(\boldsymbol{x}_i, y_i)\}_i$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{0, 1\}$;
- Goal: Classification through a linear classifier $\boldsymbol{x} \mapsto \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$.

Example: Nonlinear regression

Context

- Data set $\{(\boldsymbol{x}_i, y_i)\}_i$, $\boldsymbol{x}_i \in \mathbb{R}^d$, $y_i \in \{0, 1\}$;
- Goal: Classification through a linear classifier $\boldsymbol{x} \mapsto \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}$.

A loss function

- We use a sigmoid function: $\phi(\boldsymbol{x}_i; \boldsymbol{w}) = \left(1 + e^{-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}\right)$;
- Our goal is now to penalize the squared error $(y_i - \phi(\boldsymbol{x}_i; \boldsymbol{w}))^2$.

Example: Nonconvex loss function (2)

The optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \frac{1}{1+e^{-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}}\right)^2.$$

Example: Nonconvex loss function (2)

The optimization problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) = \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \frac{1}{1+e^{-\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}}}\right)^2.$$

- Nonconvex problem;
- Nonlinear least-squares structure;
- Smooth: can apply gradient descent.

Outline

What's optimization?

- Operations research;
- Decision-making;
- Decision sciences;
- Mathematical programming;
- Mathematical optimization.

$\Rightarrow$ All of these can be considered as optimization.

What's optimization?

- Operations research;
- Decision-making;
- Decision sciences;
- Mathematical programming;
- Mathematical optimization.

$\Rightarrow$ All of these can be considered as optimization.

My definition

The purpose of optimization is to make the best decision out of a set of alternatives.

Formulation of an optimization problem

A **minimization problem** of d real parameters is written as follows :

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{F}$$

Formulation of an optimization problem

A **minimization problem** of d real parameters is written as follows :

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{F}$$

- $\boldsymbol{w}$ represents the optimization variable(s);
- d is the dimension of the problem (we will assume $d \geq 1$);
- $f(\cdot)$ is the objective/cost/loss function;
- $\mathcal{F}$ is the constraint/feasible set.

Formulation of an optimization problem

A **minimization problem** of d real parameters is written as follows :

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{F}$$

- $\boldsymbol{w}$ represents the optimization variable(s);
- d is the dimension of the problem (we will assume $d \geq 1$);
- $f(\cdot)$ is the objective/cost/loss function;
- $\mathcal{F}$ is the constraint/feasible set.

Maximizing f is equivalent to minimizing $-f$.

Local and global solutions

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) \quad \text{subject to} \quad \boldsymbol{w} \in \mathcal{F}$$

Local minimum (also called minimizer)

- A point $\boldsymbol{w}^*$ is a **local minimum** of the problem if there exists a neighborhood $\mathcal{N}$ of $\boldsymbol{w}^*$ such that $f(\boldsymbol{w}^*) \leq f(\boldsymbol{w})\ \forall \boldsymbol{w} \in \mathcal{N} \cap \mathcal{F}$;
- A local minimum such that $f(\boldsymbol{w}^*) < f(\boldsymbol{w})\ \forall \boldsymbol{w} \in \mathcal{N} \cap \mathcal{F},\ \boldsymbol{w} \neq \boldsymbol{w}^*$ is called a strict local minimum.

Global minimum

A point $\boldsymbol{w}^*$ is a **global minimum** of the problem if $f(\boldsymbol{w}^*) \leq f(\boldsymbol{w})\ \forall \boldsymbol{w} \in \mathcal{F}$.

Local and global solutions (2)

- In general, finding global solutions is hard;
- Local solutions can also be hard to find.

Local and global solutions (2)

- In general, finding global solutions is hard;
- Local solutions can also be hard to find.

Tractable cases

- When the objective function behaves nicely;
- Suitable properties of the constraint set (if any).

Optimality conditions for unconstrained optimization

Unconstrained problem: $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$,
f continuously differentiable.

Optimality conditions for unconstrained optimization

Unconstrained problem: $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$,
f continuously differentiable.

First-order necessary condition

If $\boldsymbol{w}^*$ is a local minimum of the problem, then

$$\|\nabla f(\boldsymbol{w}^*)\| = 0.$$

Optimality conditions for unconstrained optimization

Unconstrained problem: $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$,
f continuously differentiable.

First-order necessary condition

If $\boldsymbol{w}^*$ is a local minimum of the problem, then

$$\|\nabla f(\boldsymbol{w}^*)\| = 0.$$

- This condition is only necessary;
- A point such that $\|\nabla f(\boldsymbol{w}^*)\| = 0$ can also be a local maximum or a saddle point.

Convex set

A set $\mathcal{C} \in \mathbb{R}^d$ is called **convex** if

$$\forall (\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}^2, \ \forall t \in [0,1], \qquad t\boldsymbol{u} + (1-t)\boldsymbol{v} \in \mathcal{C}.$$

Convex set

A set $\mathcal{C} \in \mathbb{R}^d$ is called **convex** if

$$\forall (\boldsymbol{u}, \boldsymbol{v}) \in \mathcal{C}^2, \ \forall t \in [0,1], \qquad t\boldsymbol{u} + (1-t)\boldsymbol{v} \in \mathcal{C}.$$

Examples:

- $\mathbb{R}^d$;
- Line segment: $\{t\boldsymbol{w} | t \in \mathbb{R}\}$ for some $\boldsymbol{w} \in \mathbb{R}^d$;
- Sphere: $\left\{ \boldsymbol{w} \in \mathbb{R}^d | \|\boldsymbol{w}\|_2^2 = \sum_{i=1}^d [\boldsymbol{w}]_i^2 \leq 1 \right\}$.

Convex function

Generic definition

A function $f : \mathbb{R}^d \to \mathbb{R}$ is **convex** if

$$\forall (\boldsymbol{u}, \boldsymbol{v}) \in (\mathbb{R}^d)^2,\ \forall t \in [0,1], \qquad f(t\boldsymbol{u} + (1-t)\boldsymbol{v}) \leq t\, f(\boldsymbol{u}) + (1-t)\, f(\boldsymbol{v}).$$

Convex function

Generic definition

A function $f : \mathbb{R}^d \to \mathbb{R}$ is **convex** if

$$\forall (\boldsymbol{u}, \boldsymbol{v}) \in (\mathbb{R}^d)^2,\ \forall t \in [0,1], \qquad f(t\boldsymbol{u} + (1-t)\boldsymbol{v}) \leq t\, f(\boldsymbol{u}) + (1-t)\, f(\boldsymbol{v}).$$

Examples:

- Linear function: $f(\boldsymbol{w}) = \boldsymbol{a}^{\mathrm{T}}\boldsymbol{w} + b$;
- Squared Euclidean norm: $f(\boldsymbol{w}) = \|\boldsymbol{w}\|_2^2 = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{w}$.

Convexity and gradient

A continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if and only if

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d, \quad f(v) \geq f(u) + \nabla f(u)^{\mathrm{T}}(v - u).$$

Smooth convex functions

Convexity and gradient

A continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if and only if

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d, \quad f(v) \geq f(u) + \nabla f(u)^{\mathrm{T}}(v - u).$$

The other key inequality in optimization.

Convex optimization problem

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}), f \text{ convex.}$$

Convex optimization problem

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}), f \text{ convex.}$$

Theorem

Every local minimum of f is a global minimum.

Convex optimization problem

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}), f \text{ convex.}$$

Theorem

Every local minimum of f is a global minimum.

Corollary

If f is continuously differentiable, every point $\boldsymbol{w}^*$ such that $\|\nabla f(\boldsymbol{w}^*)\| = 0$ is a global minimum of f.

Strong convexity

Definition

A function $f : \mathbb{R}^d \to \mathbb{R}$ in $\mathcal{C}^1$ is μ-strongly convex (or *strongly convex of modulus* $\mu > 0$) if for all $(\boldsymbol{u}, \boldsymbol{v}) \in (\mathbb{R}^d)^2$ and $t \in [0, 1]$,

$$f(t\boldsymbol{u} + (1-t)\boldsymbol{v}) \leq t f(\boldsymbol{u}) + (1-t) f(\boldsymbol{v}) - \frac{\mu}{2} t(1-t)\|\boldsymbol{v} - \boldsymbol{u}\|^2.$$

Gradient and strong convexity

Let $f : \mathbb{R}^d \to \mathbb{R}$, $f \in \mathcal{C}^1$. Then,

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d, \quad f(\boldsymbol{v}) \geq f(\boldsymbol{u}) + \nabla f(\boldsymbol{u})^{\mathrm{T}}(\boldsymbol{v} - \boldsymbol{u}) + \frac{\mu}{2}\|\boldsymbol{v} - \boldsymbol{u}\|^2.$$

Strong convexity

Definition

A function $f : \mathbb{R}^d \to \mathbb{R}$ in $\mathcal{C}^1$ is μ-strongly convex (or *strongly convex of modulus* $\mu > 0$) if for all $(\boldsymbol{u}, \boldsymbol{v}) \in (\mathbb{R}^d)^2$ and $t \in [0, 1]$,

$$f(t\boldsymbol{u} + (1-t)\boldsymbol{v}) \leq t f(\boldsymbol{u}) + (1-t) f(\boldsymbol{v}) - \frac{\mu}{2} t(1-t)\|\boldsymbol{v} - \boldsymbol{u}\|^2.$$

Theorem

Any strongly convex function in $\mathcal{C}^1$ has a unique global minimizer.

Gradient and strong convexity

Let $f : \mathbb{R}^d \to \mathbb{R}$, $f \in \mathcal{C}^1$. Then,

$$\forall \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^d, \quad f(\boldsymbol{v}) \geq f(\boldsymbol{u}) + \nabla f(\boldsymbol{u})^{\mathrm{T}}(\boldsymbol{v} - \boldsymbol{u}) + \frac{\mu}{2}\|\boldsymbol{v} - \boldsymbol{u}\|^2.$$

Outline

Three ways to study optimization problems

- **Mathematical :** Prove existence of solutions, well-posedness of a problem. Study complex optimization formulations.
- **Computational :** Write a piece of software to solve specific or generic optimization problems in practice.
- **Algorithmic :** Design algorithms, establish theoretical guarantees and validate their practical implementation.

Three ways to study optimization problems

- **Mathematical :** Prove existence of solutions, well-posedness of a problem. Study complex optimization formulations.
- **Computational :** Write a piece of software to solve specific or generic optimization problems in practice.
- **Algorithmic :** Design algorithms, establish theoretical guarantees and validate their practical implementation.

This course is about the third category.

How to solve an optimization problem?

The ideal approach

- Find the solutions of $\|\nabla f(\boldsymbol{w})\| = 0$;
- Choose the one with the lowest function value.

How to solve an optimization problem?

The ideal approach

- Find the solutions of $\|\nabla f(\boldsymbol{w})\| = 0$;
- Choose the one with the lowest function value.

What's wrong with that?

- Solving a nonlinear equation directly is hard;
- There can be infinitely many solutions;
- The procedure has to be implemented eventually.

How we shall proceed

Iterative procedures

- Driving principle : given the current solution, move towards a (potentially) better point;
- Requires a certain amount of calculation at every iteration.

Our goal in the rest of the course

- Propose several algorithms;
- Analyze their theoretical behavior and guarantees;
- Check their practical appeal (lab sessions).

What do we expect?

In order to solve $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$, we hope to achieve one of the following:

1. The iterates should get close to a solution;
2. The function values should get close to the optimum;
3. The optimality conditions should get close to be satisfied.

What do we expect?

In order to solve $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$, we hope to achieve one of the following:

1. The iterates should get close to a solution;
2. The function values should get close to the optimum;
3. The optimality conditions should get close to be satisfied.

Convergence of iterates

The method generates a sequence of points (iterates) $\{\boldsymbol{w}_k\}_k$ such that

$$\|\boldsymbol{w}_k - \boldsymbol{w}^*\| \to 0 \quad \text{when } k \to \infty,$$

where $\boldsymbol{w}^*$ is an optimal value of the problem.
(Typical of (strongly) convex functions.)

What do we expect? ('ed)

Convergence in function value

$$f(\boldsymbol{w}_k) \to f^* \quad \text{when } k \to \infty,$$

where f^* is the optimal value of the problem.
(Typical of (strongly) convex functions.)

What do we expect? ('ed)

Convergence in function value

$$f(\boldsymbol{w}_k) \to f^* \quad \text{when } k \to \infty,$$

where f^* is the optimal value of the problem.
(Typical of (strongly) convex functions.)

Convergence to a stationary point for differentiable f

$$\|\nabla f(\boldsymbol{w}_k)\| \to 0 \quad \text{when } k \to \infty.$$

More generic condition.

Why these conditions?

Unlike in theory, in practice:

- We do not know the optimal solution(s);
- We do not know the optimal value.

Why these conditions?

Unlike in theory, in practice:

- We do not know the optimal solution(s);
- We do not know the optimal value.

From an algorithmic standpoint,

- We can measure the behavior of the iterates;
- We can evaluate the objective and try to decrease it iteratively;
- We can evaluate/estimate the gradient norm and measure its decrease to zero.

Remark: Convergence and convergence rates

- In optimization, classical results are asymptotic:

$$Ex: \ \|\nabla f(\boldsymbol{w}_k)\| \to 0 \quad \text{when } k \to \infty.$$

Remark: Convergence and convergence rates

- In optimization, classical results are asymptotic:

$$Ex: \ \|\nabla f(\boldsymbol{w}_k)\| \to 0 \quad \text{when } k \to \infty.$$

- **Global convergence rates** are now very popular:

$$Ex: \|\nabla f(\boldsymbol{w}_k)\| = \mathcal{O}\left(\frac{1}{k}\right) \quad \Leftrightarrow \quad \exists C > 0, \|\nabla f(\boldsymbol{w}_k)\| \leq \frac{C}{k} \ \forall k.$$

An equivalent notion is that of a **worst-case complexity bound**:

$$Ex: \text{Algorithm } \mathcal{O}\left(\epsilon^{-1}\right) \Leftrightarrow \exists C > 0, \ \|\nabla f(\boldsymbol{w}_k)\| \leq \epsilon \text{ when } k \geq \frac{C}{\epsilon}.$$

- Common in convex optimization;
- Standard in theoretical computer science/statistics.

On the computational side

Optimizers code in...

- `C/C++/Fortran` (high-performance computing)
- `Matlab, Python` (prototyping);
- `Julia`.

On the computational side

Optimizers code in...

- `C/C++/Fortran` (high-performance computing)
- `Matlab, Python` (prototyping);
- `Julia`.

Specific optimization modeling languages

- `GAMS, AMPL, CVX` are broad-spectrum languages;
- `MATPOWER, PyTorch` are domain-oriented;
- Can be interfaced with the languages above.

Conclusions: basics of optimization

Modeling framework

- Objective, constraints;
- Characterization of the solutions.

Conclusions: basics of optimization

Modeling framework

- Objective, constraints;
- Characterization of the solutions.

Important tools

- Derivatives and Taylor expansion;
- Convexity.

Conclusions: basics of optimization

Modeling framework

- Objective, constraints;
- Characterization of the solutions.

Important tools

- Derivatives and Taylor expansion;
- Convexity.

Algorithmic principle

- Iterative process: find a sequence of points that leads to a solution;
- Quantify how fast.

Outline

General optimization problem

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}).$$

Assumptions: f smooth, bounded below.

General optimization problem

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}).$$

Assumptions: f smooth, bounded below.

Key properties

- Smoothness: We will exploit the gradient of f.
- In presence of convexity, get better guarantees.

Negative gradient direction

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}), \quad f \in \mathcal{C}_L^{1,1}.$$

Consider any $\boldsymbol{w} \in \mathbb{R}^d$. Then, one of the two assertions below holds:

1. Either $\boldsymbol{w}$ is a local minimum and $\nabla f(\boldsymbol{w}) = 0$;
2. Or the function f decreases locally from $\boldsymbol{w}$ in the direction of $-\nabla f(\boldsymbol{w})$. *Proof: Taylor expansion.*

Gradient descent method

Inputs: $\boldsymbol{w}_0 \in \mathbb{R}^d,\ \alpha_0 > 0,\ \varepsilon > 0, k_{\max} \in \mathbb{N}$.
Set $k = 0$.

1. Evaluate $\nabla f(\boldsymbol{w}_k)$; if $\|\nabla f(\boldsymbol{w}_k)\| < \varepsilon$ stop.
2. Set $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)$.
3. Increment k by 1; if $k = k_{\max}$ stop, otherwise go to Step 1.

Gradient descent method

Inputs: $\boldsymbol{w}_0 \in \mathbb{R}^d,\ \alpha_0 > 0,\ \varepsilon > 0, k_{\max} \in \mathbb{N}$.
Set $k = 0$.

1. Evaluate $\nabla f(\boldsymbol{w}_k)$; if $\|\nabla f(\boldsymbol{w}_k)\| < \varepsilon$ stop.
2. Set $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)$.
3. Increment k by 1; if $k = k_{\max}$ stop, otherwise go to Step 1.

Stopping criterion

- Convergence criterion: $\|\nabla f(\boldsymbol{w}_k)\| < \varepsilon$;
- Budget criterion (optional): $k = k_{\max}$.

Choosing the stepsize

Constant stepsize

If $f \in \mathcal{C}_L^{1,1}$, set $\alpha_k = \frac{1}{L}$:

- Guaranteed decrease at every iteration;
- But requires knowledge of L.

Decreasing stepsize

Choose α_k such that $\alpha_k \to 0$.

- Guarantees that f will decrease eventually (for small stepsizes);
- But steps get smaller and smaller.

Choosing the stepsize (2)

What's done in optimization

- Line search: At every iteration, α_k is obtained by *backtracking* on a subset of values (ex: $1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \ldots,$).
- The chosen value must satisfy certain conditions (ex: decreasing the function value).

What's done in optimization for ML

- Start with a fixed value until the method starts stalling (gradient gets small);
- Decrease the step size, then repeat.

Back to the general setup

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} f(\boldsymbol{x}), \qquad f \in \mathcal{C}_L^{1,1}.$$

Gradient descent

- Iteration: $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)$, stop if $\nabla f(\boldsymbol{w}_k) = 0$.
- Typical choice in theory : $\alpha_k = \frac{1}{L}$.

Theoretical analysis

- Convergence: Show that $\|\nabla f(\boldsymbol{w}_k)\| \to 0$;
- Convergence rate: Look at how fast $\|\nabla f(\boldsymbol{w}_k)\|$ decreases.
- Worst-case complexity: Equivalent to convergence rate, measures the cost of satisfying $\|\nabla f(\boldsymbol{w}_k)\| \leq \epsilon$ for $\epsilon > 0$.

Complexity measures

Nonconvex case

- Goal: $\|\nabla f(\boldsymbol{w}_k)\| \leq \epsilon$;
- Get close to being a stationary point.

Convex/Strongly convex case

- Goal: $f(\boldsymbol{w}_k) - f^* \leq \epsilon$, with $f^* = \min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w})$;
- Get close to the optimal function value.

Complexity results for gradient descent

Theorem (nonconvex functions)

If $f \in \mathcal{C}_L^{1,1}$ and $\alpha_k = \frac{1}{L}$, GD finds $\boldsymbol{w}_k$ such that $\|\nabla f(\boldsymbol{w}_k)\| \leq \epsilon$ in at most $\mathcal{O}(\epsilon^{-2})$ iterations.

Theorem (convex functions)

If $f \in \mathcal{C}_L^{1,1}$ is convex and $\alpha_k = \frac{1}{L}$, GD finds $\boldsymbol{w}_k$ such that $f(\boldsymbol{w}_k) - f^* \leq \epsilon$ in at most

- $\mathcal{O}(\epsilon^{-1})$ iterations;
- $\mathcal{O}(\log(\epsilon^{-1}))$ iterations if f is strongly convex.

Complexity results for gradient descent (2)

Interpretation

- GD on nonconvex functions: $\mathcal{O}(\epsilon^{-2})$;
- GD on convex functions: $\mathcal{O}(\epsilon^{-1})$;
- GD on strongly convex functions: $\mathcal{O}(\log(\epsilon^{-1}))$

At a lower cost, can get better guarantees for convex and strongly convex problems.

Complexity VS Convergence rate

- Nonconvex: $\|\nabla f(\boldsymbol{w}_k)\| \leq C/k^{1/2}$;
- Convex:

$$f(\boldsymbol{w}_k) - f^* \leq \begin{cases} C/k & \text{if } f \text{ convex;} \\ C(1-\frac{\mu}{L})^k & \text{if } f \ \mu\text{-strongly convex.} \end{cases}$$

Outline

Nonsmooth functions

Definition

A function is called **nonsmooth** if it is not differentiable everywhere.
NB: Nonsmooth $\neq$ Discontinuous.

Example of nonsmooth functions

- $w \mapsto |w|$ from $\mathbb{R}$ to $\mathbb{R}$;
- $w \mapsto \|w\|_1$ from $\mathbb{R}^d$ to $\mathbb{R}$;
- ReLU: $w \mapsto \max\{w, 0\}$ from $\mathbb{R}^d$ to $\mathbb{R}$.

Nonsmoothness and derivatives

Optimizing nonsmooth problems

- Optimization methods use derivatives to make progress;
- Nonsmooth problems may not have derivatives;
- In general, generalized derivatives are used (many definitions).

Alternatives

- When possible, use a smooth reformulation (involving constraints):
  Ex) $\min_{w \in \mathbb{R}} |w|$ is equivalent to
  $\min_{w,t^+,t^- \in \mathbb{R}} t^+ - t^- \quad \text{s. t.} \quad w = t^+ - t^-, t^+ \geq 0, t^- \geq 0.$
- When the *function* is Lipschitz continuous, it has a gradient at almost every point.
  Ex) Neural networks using activation functions like ReLU.
  Warning: The functions are often non-differentiable at their minima.

Subgradients for nonsmooth convex problems

Definition

Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function. A vector $\boldsymbol{g} \in \mathbb{R}^n$ is called a *subgradient* of f at $\boldsymbol{w} \in \mathbb{R}^n$ if

$$\forall \boldsymbol{z} \in \mathbb{R}^n, \qquad f(\boldsymbol{z}) \geq f(\boldsymbol{w}) + \boldsymbol{g}^{\mathrm{T}}(\boldsymbol{z} - \boldsymbol{w}).$$

The set of all subgradients of f at $\boldsymbol{w}$ is called the *subdifferential* of f at $\boldsymbol{w}$, and denoted by $\partial f(\boldsymbol{w})$.

- If f differentiable at $\boldsymbol{w}$, $\partial f(\boldsymbol{w}) = \{\nabla f(\boldsymbol{w})\}$;
- $0 \in \partial f(\boldsymbol{w}) \Leftrightarrow \boldsymbol{w}$ minimum of f!

Example: Let $f : \mathbb{R} \to \mathbb{R}$, $f(w) = |w|$.

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

Subgradient method

Iteration for $\min_{\boldsymbol{w}} f(\boldsymbol{w})$, f convex nonsmooth

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \boldsymbol{g}_k, \qquad \boldsymbol{g}_k \in \partial f(\boldsymbol{w}_k).$$

- Flexibility in choosing the subgradient;
- Choosing stepsize is more difficult than in gradient descent because the function does not vary smoothly: a subgradient can be a direction of increase!

Back to where we started

The linear SVM problem

$$\min_{\boldsymbol{w}\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} \max\{1 - y_i \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}, 0\} + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2.$$

- The problem is **regularized** (by a data-independent term);
- The purpose of regularization is to enforce specific properties/structure on a solution.

General form of a regularized problem

$$\min_{\boldsymbol{w} \in R^d} \underbrace{f(\boldsymbol{w})}_{\textit{loss function}} + \underbrace{\lambda \Omega(\boldsymbol{w})}_{\textit{regularization term}} .$$

where $\lambda > 0$ is called a regularization parameter.

Example: Ridge regularization

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) + \frac{\lambda}{2} \|\boldsymbol{w}\|_2^2.$$

Interpretations:

- Equivalent to enforcing a constraint on $\|\boldsymbol{w}\|_2^2 = \sum_{i=1}^d w_i^2$;
- Penalizes $\boldsymbol{w}$s with large components;
- The variance of the solution w. r. t. the data is reduced;
- The objective function is strongly convex.

Nonsmooth regularized problems

Goal: Induce sparsity

- We want a solution $\boldsymbol{w} \in \mathbb{R}^d$ that explains the data with as few nonzero components as possible;
- Ideal problem: $\min_{\boldsymbol{w}} f(\boldsymbol{w}) + \lambda \|\boldsymbol{w}\|_0$, where $\|\boldsymbol{v}\|_0 = |\{i | [\boldsymbol{v}]_i \neq 0\}|$.
  Issues: The ℓ_0 quasi-norm is nonsmooth, discontinuous and introduces combinatorics on the problem.

A better approach: LASSO regularization

LASSO=Least Absolute Shrinkage and Selection Operator

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) + \lambda \|\boldsymbol{w}\|_1, \quad \|\boldsymbol{w}\|_1 = \sum_{i=1}^{d} |w_i|.$$

- $\|\cdot\|_1$ is convex, continuous, and a norm;
- It is nonsmooth but subgradients can be computed.

Solving regularized problems

Setup: Composite optimization

$$\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) + \lambda \Omega(\boldsymbol{w}).$$

- $f \in C^{1,1}$;
- Ω convex but nonsmooth.

Proximal approach

- Classical optimization paradigm: replace a problem by a sequence of easier (sub)problems;
- Exploit smoothness of f, use the structure of Ω to solve the subproblems;
- Those should be solvable **efficiently**.

Proximal Gradient Descent (PGD)

Iteration of PGD

$$\boldsymbol{w}_{k+1} = \underset{\boldsymbol{w}\in\mathbb{R}^d}{\operatorname{argmin}} \left\{ f(\boldsymbol{w}_k) + \nabla f(\boldsymbol{w}_k)^{\mathrm{T}}(\boldsymbol{w} - \boldsymbol{w}_k) + \frac{1}{2\alpha_k}\|\boldsymbol{w} - \boldsymbol{w}_k\|_2^2 + \lambda\Omega(\boldsymbol{w}) \right\}.$$

- If $\Omega \equiv 0$, the solution is $\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)$: **This is the Gradient Descent iteration!**
- In general, the cost of an iteration is 1 gradient call + **1 proximal subproblem solve**.

Properties

- Complexity bounds exist for nonconvex and mostly for convex f;
- Stepsize choices can be designed based on those for GD;

Example of proximal method: ISTA

Context

- Solve $\min_{\boldsymbol{w} \in \mathbb{R}^d} f(\boldsymbol{w}) + \lambda \|\boldsymbol{w}\|_1$;
- Common problem in image processing;
- Explicit form of the proximal subproblem solution.

Example of proximal method: ISTA

Context

- Solve $\min_{\boldsymbol{w}\in\mathbb{R}^d} f(\boldsymbol{w}) + \lambda\|\boldsymbol{w}\|_1$;
- Common problem in image processing;
- Explicit form of the proximal subproblem solution.

Iteration of ISTA: Iterative Soft-Thresholding Algorithm

Define $\boldsymbol{w}_{k+1}$ componentwise: for any $i \in \{1, \ldots, d\}$,

$$[\boldsymbol{w}_{k+1}]_i = \begin{cases} [\boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)]_i + \alpha_k\lambda & \text{if } [\boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)]_i < -\alpha_k\lambda \\ [\boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)]_i - \alpha_k\lambda & \text{if } [\boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)]_i > \alpha_k\lambda \\ 0 & \text{if } [\boldsymbol{w}_k - \alpha_k \nabla f(\boldsymbol{w}_k)]_i \in [-\alpha_k\lambda, \alpha_k\lambda]. \end{cases}$$

Summary

Optimization problems in ML

- Common feature: Depend on data.
- Distinctive features: Convexity, smoothness, regularization.

Gradient descent

- The basic block for optimization.
- Applies to convex and nonconvex functions.
- Some freedom in the implementation (see lab session).

Beyond gradient descent

- Nonsmoothness$\Rightarrow$ Subgradient methods!
- Regularization$\Rightarrow$ Proximal methods!
- Data dependency? $\Rightarrow$ See next lecture.

References

Textbooks:

- A. Beck, *First-order methods in optimization*, MOS-SIAM Series on Optimization, 2017.
  $\Rightarrow$ *Chapter 10* is related to proximal methods, and contains many examples of explicit proximal step calculations.
- J. Wright and Y. Ma, *High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Com- putation, and Applications*, Cambridge University Press, 2022.
  $\Rightarrow$ Numerous applications, freely available online.
- S. J. Wright and B. Recht, *Optimization for Data Analysis*, Cambridge University Press, 2022.
  $\Rightarrow$ Textbook with full analysis for gradient descent.