Optimization for Machine Learning

September 20, 2023

Today

- Syllabus
- Motivation
- Introduction to optimization

Syllabus

↳ The course is in English as most resources in the field are produced in English
↳ Please play along!
↳ Questions / Discussions in French are always welcome at the end or via email

↳ Instructors: Clément ROYER (First 4 lectures)
Florentin GOYENS (Last 4 lectures)

Contact: clement.royer@lamsade.dauphine.fr

↳ Schedule: 8 x 3 hours either 8.30am-11.45am or 1.45pm-5pm

↳ Tentative outline

09/20 (am): Intro to optimization
09/27 (am): A modern view on gradient descent
10/04 (am): Accelerated methods
10/05 (pm): Proximal methods and regularization
11/08 (am): Stochastic gradient (basics)
11/09 (pm): Stochastic gradient (advanced)
11/15 (am): Large-scale optimization
11/16 (pm): Overview

Assessment

- Exam (60%): December 13, 10am-12pm
(will be given in both French and English)

- Course project (40%)
  Individual, essentially a long Python notebook with coding tasks and questions
  (loose timeline: between Oct 6 and Dec 23)

Course webpage

- lecture notes to be updated
- Notebooks
- Exercises + solutions
- Virtual board

MOTIVATION

↳ Optimization :
- Field of study that is concerned with making the best decision out of a set of alternatives
- As a subfield of theoretical computer science / computational mathematics, matured between 1980s - 2000s with efficient software (EX: Gurobi, CPLEX)
- Success stories:
  - Finance (portfolio allocation)
  - Supply chain
  - Facility location
  - Vaccine dispatch
- Shift in the past decades (2000s - 2020s)

  Data-driven optimization

  ⇒ leads to specific optimization formulations

  ⇒ Changes the preferred classes of algorithms (compared to classical applications of optimization)

  ↳ Old algorithms became trendy (ex: stochastic gradient)

  ↳ Other algorithms are inelevant (ex: Newton's method)

Typical ML optimization

↳ Starts with data

$$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{n}$$

x_i: input, y_i: output

n samples/data points

↳ Goal: Discover a mapping between the x_is and the y_is

$\Rightarrow$ h such that $h(x_i) \simeq y_i$

Process: Training / Learning

↳ Typically h is defined/parameterized with a vector $w \in \mathbb{R}^d$ and learning consists in finding the best value for w given the data

↘ Optimize!

An optimization problem associated with this task will have the form

Want w that gives the smallest objective value ↑

minimize

Variables/parameters → $w \in \mathbb{R}^d$

$$f_{\mathcal{D}}(w) + \lambda r(w)$$

$f(w)$

$f_{\mathcal{D}}(w)$: objective function (to be optimized as a function of w)

$w \in \mathbb{R}^d : w = \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix}$ $w_i \in \mathbb{R}$

Specific objective structure : $f_D(w) + \lambda r(w)$

- $f_D(w)$ training / empirical loss that depends on the data

Average over samples

Model

$$f_D(w) = \frac{1}{n} \sum_{i=1}^{n} \ell\left(h(x_i, w) ; y_i\right)$$

"data-fitting term"

loss function (measured agreement between $h(x_i; w)$ and y_i)

Data points

$\Rightarrow$ The structure of f_D and its dependency on D are the reasons why some algorithms are better suited than others to solve this problem

- $r(w)$: "regularization term"
  - $\rightarrow$ Does not depend on data
  - $\rightarrow$ Is used to promote properties on w (and therefore on the mapping h)

- $\lambda > 0$: "regularization parameter"
  - $\Rightarrow$ Expresses a trade off between fitting the data and satisfying the properties represented by r

Examples

(1) Linear least squares

$$\text{Data} = \{(x_i, y_i)\}_{i=1}^{n} \qquad x_i \in \mathbb{R}^d \qquad y_i \in \mathbb{R} \quad \forall i = 1..n$$

$\mathbb{R}^{n \times d}$: matrices with n rows and d columns

$$X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in \mathbb{R}^{n \times d} \qquad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix} \in \mathbb{R}^n$$

$$x_i = \begin{bmatrix} \vdots \end{bmatrix} \in \mathbb{R}^d \qquad x_i^T = [\,\ldots\ldots\,]$$

Goal: Find a linear relationship between $\{x_i\}$ and $\{y_i\}$

i.e. finding $w \in \mathbb{R}^d$ such that

$$Xw \simeq y \iff \forall i = 1..n \quad x_i^T w \simeq y_i$$

↳ There may not exist w such that $x_i^T w = y_i \ \forall i$ but with optimization you can find the best possible w

Problem

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \ \frac{1}{2n} \underbrace{\|Xw - y\|^2}_{\in \mathbb{R}^n} = \frac{1}{2n} \sum_{i=1}^{n} \underbrace{\left(x_i^T w - y_i\right)^2}_{\in \mathbb{R}}$$

$$\forall v \in \mathbb{R}^n, \ \|v\| = \sqrt{\sum_{i=1}^{n} v_i^2} \quad \text{(Euclidean norm)}$$

$$\|v\| = \sqrt{v^T v}$$

scalar product

$$\forall (v, u) \in (\mathbb{R}^n)^2, \ v^T u = \sum_{i=1}^{n} v_i u_i$$

⇒ Basic formulation, also called / equivalent to linear regression

⇒ Variants

- Ridge regression

$$\text{minimize}_{w \in \mathbb{R}^d} \quad \frac{1}{2n} \|Xw - y\|^2 + \lambda \|w\|_2^2 \quad \text{with } \lambda > 0$$

$\|w\|_2^2 = \sum_{i=1}^{d} w_i^2$

$\|\cdot\|_2$: regularization

(1996) - LASSO / ℓ_1 regression

$$\text{minimize}_{w \in \mathbb{R}^d} \quad \frac{1}{2n} \|Xw - y\|^2 + \lambda \|w\|_1$$

$\|\cdot\|_1$: regularization

$$\|w\|_1 := \sum_{i=1}^{d} |w_i|$$

(2) Support vector machines (SVM)

Goal: Given $\{(x_i, y_i)\}_{i=1}^{n}$, $x_i \in \mathbb{R}^d$, $y_i \in \{-1, +1\}$ ("labels/classes")

find a mapping that classifies the x_i's correctly

Approach: Consider linear mappings that define hyperplanes

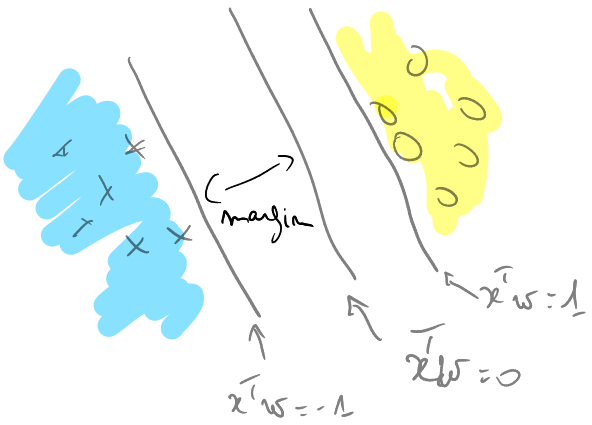
Seek $h(x; w) = x^T w$ (also possible: $x^T w + v$)

$w \in \mathbb{R}^d$ $\quad v \in \mathbb{R}$

Such that

$x_i^T w \geq 1$ if $y_i = 1$

$x_i^T w \leq -1$ if $y_i = -1$



Problem:

"Linear SVM"

↓

Connected with optimization in the late 1990s

minimize $_{w \in \mathbb{R}^d}$

$$\frac{1}{n} \sum_{i=1}^{n} \underbrace{\max(1 - y_i x_i^T w, 0)}_{\geq 0}$$

hinge loss $(x_i^T w, y_i)$

if $x_i^T w > 1$ and $y_i = 1$ $\Rightarrow 0$

if $x_i^T w < -1$ and $y_i = -1$ $\Rightarrow 0$

if $x_i^T w > 1$ and $y_i = -1$ $\Rightarrow$ loss > 0

$x_i^T w < -1$ — $y_i = 1$ $\Rightarrow$

if $|x_i^T w| \leq 1$ $\Rightarrow$ loss "small" (between 0 and 2)

↳ Can add regularization terms (as in least squares)

↳ Generalized formulations $\Rightarrow$ kernel SVM

Remark: $\forall f : \mathbb{R}^d \to \mathbb{R}$

minimize $_{w \in \mathbb{R}^d}$ $f(w)$

and

minimize $_{w \in \mathbb{R}^d}$ $\lambda \times f(w)$ with $\lambda > 0$

have the same set of solutions

(3) Deep neural networks

↳ Feed-forward architecture, multi-class classification

↳ Data: $\{(x_i, [y_{ij}]_j)\}_{i=1}^n$ $\quad x_i \in \mathbb{R}^{d_0}$

$y_i = [y_{ij}]_j \in \mathbb{R}^C$

C number of classes

1-hot encoding $\quad y_{ij} = 1$ if x_i is in class j

$y_{ij} = 0$ otherwise

↳ Neural architecture

L layers

NN: $x^0 \in \mathbb{R}^{d_0} \mapsto x^1 \in \mathbb{R}^{d_1} \mapsto \cdots \mapsto x^L \in \mathbb{R}^{d_L}$ $\quad d_L = C$

x^1: First layer of the NN computed from x^0

$\mapsto$: "maps to"

x^L: output of the NN

$$\forall l = 1..L, \quad x^l = \sigma\left(W^l x^{l-1} + b^l\right)$$

- σ: activation function applied componentwise
- W^l: weight matrix, $W^l \in \mathbb{R}^{d_l \times d_{l-1}}$
- b^l: bias vector, $b^l \in \mathbb{R}^{d_l}$

$\forall u \in \mathbb{R}^{d_l}, \quad \sigma(u) := [\sigma(u_i)]_{i=1..d_l}$

Ex): $\sigma(t) = t$ (identity)

$\sigma(t) = \max(t, 0)$ (ReLU)

$\sigma(t) = \tanh(t)$

Model: Takes $x_i \in \mathbb{R}^{d_0}$ as input $(x^0 = x_i)$

and outputs $x_i^L \in \mathbb{R}^{d_L}$ $\quad d_L = C$

Parameters: $(W^l, b^l)_{l=1..L}$ $\Rightarrow$ stacked into a vector

$$w \in \mathbb{R}^d$$

$$d = \underbrace{d_1 + d_1 d_0}_{b^1 \; W^1} + \cdots + \underbrace{d_L + d_L d_{L-1}}_{b^L \; W^L}$$

$\rightarrow L$ can be very large (hundreds at least)

$\rightarrow d$ usually is very large

Optimization problem

$$\underset{w \in \mathbb{R}^d}{\text{minimize}} \quad \frac{1}{n} \sum_{i=1}^{n} \left\{ \log\left(\sum_{j=1}^{C} \exp\left(x_{ij}^L \right) \right) - \sum_{j=1}^{C} y_{ij} \, x_{ij}^L \right\}$$

$$x_i^L = \left[x_{ij}^L \right]_{j=1}^{C}$$

$\{\;\} \Rightarrow$ negative log-likelihood

Want x_i^L to represent probabilities that x_i belongs to each of the classes

$y_i = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ ← correct class j — ideal model $\quad x_i^L = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \leftarrow j$ — what the network learns $\quad x_i^L = \begin{bmatrix} p_1 \\ \vdots \\ p_C \end{bmatrix}$ probabilities of belonging to a class

Goal: Maximize p_j
(want $p_j > p_i \ \forall i \neq j$)

(1) (2) (3)

↳ All these examples involve:
- functions of real variables that are continuous
- structure (finite sum, regularization, layer calculations, ...)

Goal of the course: Understand how to choose the best algorithms for problems of this form

BASICS OF CONTINUOUS OPTIMIZATION

An optimization problem (for us):

$$(P) \quad \boxed{\min_{w \in \mathbb{R}^d} f(w)}$$

$f : \mathbb{R}^d \to \mathbb{R}$

f: objective function

w: (vector of) variables

(More general formulation: $\min_{w \in \mathbb{R}^d} f(w)$ s.t. $w \in F$)

"Subject to"

Constraints on w

⇒ We focus on the unconstrained setting ($F = \mathbb{R}^d$)

Solution set of (P)

- $\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} f(w) = \{ \bar{w} \in \mathbb{R}^d \mid f(\bar{w}) \leq f(w) \ \forall w \in \mathbb{R}^d \}$

  $\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} f(w)$: set $\subseteq \mathbb{R}^d$

  ↳ Can be empty ($f(w) = w$ and $d = 1$)

  ↳ Can be infinite ($f(w) = 0$)

- $\min_{w \in \mathbb{R}^d} f(w)$: "minimal value of f"

  $\min_{w \in \mathbb{R}^d} f(w) \in \mathbb{R}^d \cup \{+\infty, -\infty\}$

$\forall \lambda > 0$,

$\underset{w \in \mathbb{R}^d}{\operatorname{argmin}} f(w) = \underset{w \in \mathbb{R}^d}{\operatorname{argmin}} \lambda f(w)$

$\min_{w \in \mathbb{R}^d} \lambda f(w) = \lambda \times \min_{w \in \mathbb{R}^d} f(w)$

Example of equivalent formulations

Two problems are equivalent if the solution of one can be readily obtained by solving the other

⚠ In general, it is hard (computationally hard / intractable) to compute the argmin directly

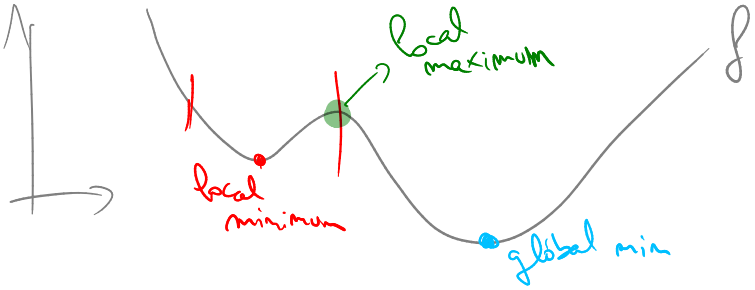
↳ Simply by function comparison

⇒ In practice, we exploit other properties of the objective to build algorithms that compute approximate solutions

Def . $\bar{w} \in \operatorname{argmin}_w f(w)$ is called a global minimum

• $\bar{w} \in \mathbb{R}^d$ is called a local minimum of f if

$\forall w \in \mathbb{R}^d$ such that [$\|\bar{w} - w\|$ is small enough] $\Rightarrow f(\bar{w}) \leq f(w)$



$\bar{w}$ local maximum $f(\bar{w}) \geq f(w)$ for $\|w - \bar{w}\|$ small enough

Local minima are (on principle) easier to converge to than global minima but they are still hard to compute and they may correspond to values of f that are far from the optimum

$\Rightarrow$ In some data science tasks, it is possible to quantify how far local minima are from global ones

↳ The notions of convexity and differentiability allow to characterize local minima (or approximations) in finite time / in an "easy" way

a) Differentiability

$f: \mathbb{R}^d \to \mathbb{R}$ is C^1 (continuously differentiable) if its derivative exists $\forall w \in \mathbb{R}^d$ and is a continuous mapping.

$\Rightarrow \forall w \in \mathbb{R}^d, \exists \nabla f(w) \in \mathbb{R}^d$ such that

↑ gradient of f at w

$\forall v \in \mathbb{R}^d, \quad f(v) \simeq f(w) + \nabla f(w)^T (v-w)$ when $\|v-w\|$ is small enough

linear function of v

$\Rightarrow$ Approximation of the change in function values around w

$\Rightarrow$ Local property that can be connected to local optimality.

Theorem (First-order necessary conditions)

Let $f: \mathbb{R}^d \to \mathbb{R}$ and $\bar{w} \in \mathbb{R}^d$

[$\bar{w}$ is a local minimum of f]

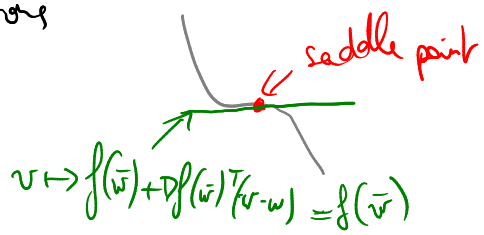
$\Rightarrow \quad \nabla f(\bar{w}) = 0_{\mathbb{R}^d} \quad (\|\nabla f(\bar{w})\| = 0)$

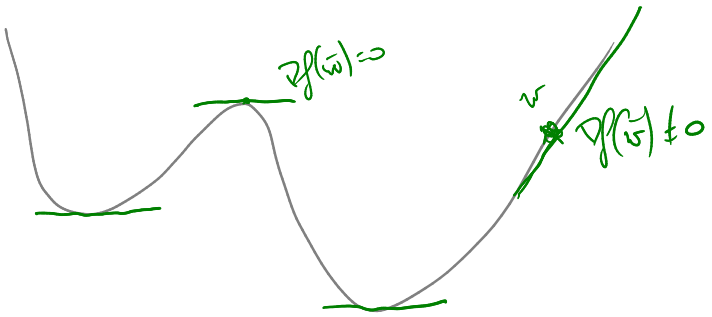
Necessary: the converse is not true

In fact, [$\bar{w}$ is a local maximum of f] $\Rightarrow \nabla f(\bar{w}) = 0_{\mathbb{R}^d}$

[$\bar{w}$ is a saddle point of f] $\Rightarrow \nabla f(\bar{w}) = 0_{\mathbb{R}^d}$

Saddle point: local min in some directions
local max in others

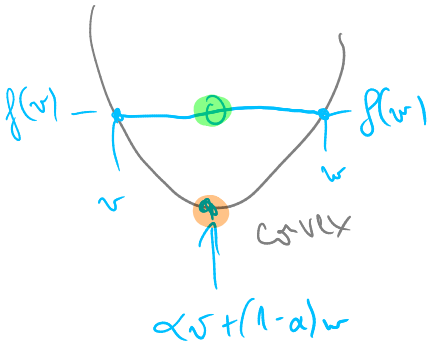


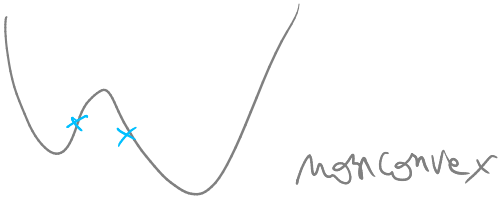


⑤ Convexity

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

$$\forall (v, w) \in (\mathbb{R}^d)^2, \quad \forall \alpha \in [0,1],$$
$$f(\alpha v + (1-\alpha) w) \leq \alpha f(v) + (1-\alpha) f(w)$$





Theorem: Let $f: \mathbb{R}^d \to \mathbb{R}$ be a convex function. Then,

i) any local minimum of f is a global minimum

ii) If $f \in C^1$, $\forall \bar{w} \in \mathbb{R}^d$

$$\left[\bar{w} \text{ is a global minimum of } f \right] \Leftrightarrow \left[\nabla f(\bar{w}) = 0 \right]$$

↑ Necessary and sufficient condition for global optimality

↳ C^1 convex functions are easy to minimize

(ex: linear least squares, SVM)

⇒ Need to find $\bar{w}$ such that $\nabla f(\bar{w}) = 0$