OPTIMIZATION FOR MACHINE LEARNING

September 16, 2024

Today (Lecture 1/16)

- Logistics
- Intro to optimization (pt 1)

LOGISTICS

Course google doc (key resource)

https://bit.ly/3XwecET

Course webpage on my website

https://www.lamsade.dauphine.fr/~croyer/teachOIM.html

INTRODUCTION TO OPTIMIZATION (IN ML)

Formal definition: Optimization is a field of study concerned with making the best decision out of a set of alternatives

↳ As a field of computational mathematics, optimization matured between 1980s and 2000s
=> Success stories: Finance, Vaccine dispatch, Scheduling, Moving plan for Dauphine's offices

→ Model-driven approach

↳ 2000s-2020s: Shift of paradigm from model-driven to data-driven

* changed the problem formulations people care about

* changed the state-of-the-art algorithms
(1990s: Newton's method
2024: Stochastic gradient and its variants)

Typical ML optimization problem

↳ Data: $\mathcal{D} = \{(a_i, y_i)\}_{i=1..n}$ — n samples

a_i: Input (vector, matrix, ...)

y_i: Output (scalar, vector, matrix, tensor, ...)

↳ Goal: * Learn a mapping from a_is to y_is
i.e. a function h such that $h(a_i) \approx y_i$

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

$x = \begin{bmatrix} x_1 \\ \vdots \\ x_d \end{bmatrix} \} d$

* In practice, consider that h is parameterized by a vector $x \in \mathbb{R}^d$

$h(\cdot; x) : a \longmapsto h(a; x)$

=> The goal becomes to find the value of x that yields the best mapping in terms of matching $h(a_i; x)$ with y_i

Problem formulation

$\lambda \times r(x)$

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

objective function $f(x)$

Alternative: $\min_{x \in \mathbb{R}^d}$ (minimize $x \in \mathbb{R}^d$)

"Minimize $f_{\mathcal{D}}(x) + \lambda r(x)$ over $x \in \mathbb{R}^d$"

x : decision variables / parameters / weights

=> Define mapping h (and thus how it fits the data)

f_D : data-fitting term, defined using h and $D = \{(a_i, y_i)\}_{i=1..m}$

Typical structure:

$$f_D(x) = \frac{1}{m} \sum_{i=1}^{m} \ell\left(h(a_i; x), y_i\right)$$

- $f_D(x) \in \mathbb{R}$
- $\frac{1}{m} \sum_{i=1}^{m}$: Average over all data points
- $h(a_i; x)$: mapping parameterized by x
- ℓ : loss function : measures agreement between $h(a_i; x)$ and y_i

$r(x)$: regularization term

* typically doesn't depend on the data
* used to penalize values of x that do not satisfy desired properties (e.g. overfitting, density)

$\lambda > 0$: regularization parameter

Expresses trade-off between regularization and data-fitting

$\lambda >> 1$: More weight on the regularization

$\lambda << 1$: ——— data-fitting term

EXAMPLES

(1) Linear least squares (aka linear regression)

Data $D = \{(a_i, y_i)\}_{i=1..m}$ $\quad a_i \in \mathbb{R}^d \quad y_i \in \mathbb{R}$

$\Rightarrow$ Representation

$$A = \begin{bmatrix} a_1^T \\ \vdots \\ a_m^T \end{bmatrix} \in \mathbb{R}^{m \times d}$$

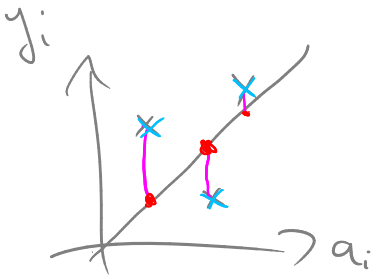
↑ Set of matrices with real coefficients, m rows, d columns

$a \in \mathbb{R}^d$ $a = [\ ]$ d (1)

$a^T = [\ \longleftrightarrow\ ]$ 1 (d)

$$y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix} \in \mathbb{R}^m$$

Goal: Compute a linear model of the data, that is a vector $x \in \mathbb{R}^d$ such that $a_i^T x \approx y_i \ \forall i = 1..m$



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

$u^T v$ = u^T (1 × d) · v (d × 1)

Problem

$$\min_{x \in \mathbb{R}^d} \frac{1}{2m} \|Ax - y\|^2 = \frac{1}{m} \sum_{i=1}^{m} \left\{ \frac{1}{2} (a_i^T x - y_i)^2 \right\}$$

mapping $a \mapsto a^T x$

loss (ℓ_2) $\ell(h, y) = \frac{1}{2}(h - y)^2$

$\forall v \in \mathbb{R}^m, \quad \|v\|^2 = \sum_{j=1}^{m} v_i^2$

where $\|Ax - y\|^2 = \sum_{i=1}^{m} (a_i^T x - y_i)^2$

Variants

* Ridge regression:

$$\min_{x \in \mathbb{R}^d} \underbrace{\frac{1}{2m} \|Ax - y\|^2}_{\text{data fitting}} + \lambda \underbrace{\|x\|^2}_{\ell_2 \text{ regularization}}$$

* LASSO: $\min_{x \in \mathbb{R}^d} \frac{1}{2n} \|Ax - y\|^2 + \lambda \|x\|_1$

$\|x\|_1 = \sum_{j=1}^{d} |x_j|$

$\Rightarrow$ Produces vector x with more zero coordinates than the original formulation

(2) Matrix completion

↳ Data: $\{(A_i, y_i)\}_{i=1..m}$ $\quad y_i \in \mathbb{R}$

$A_i = A_i^T \in \mathbb{R}^{d \times d}$ symmetric matrix

$\begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix}$ ✓ $\quad \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$ ×

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1p} \\ \vdots & & \\ A_{q1} & \cdots & A_{qp} \end{bmatrix} \in \mathbb{R}^{q \times p}$$

$$A^T = \begin{bmatrix} A_{11} & \cdots & A_{q1} \\ \vdots & & \\ A_{1p} & \cdots & A_{qp} \end{bmatrix} \in \mathbb{R}^{p \times q}$$

↳ Underlying ground truth (symmetric) matrix $M^* \in \mathbb{R}^{d \times d}$

→ A_i measurement operators

→ $y_i = \langle A_i, M^* \rangle = \text{trace}(A_i^T M^*)$

Goal: Find a good approximation for M^* that corresponds to $\{(A_i, y_i)\}_{i=1..m}$

Problem $\min_{X \in \mathbb{R}^{d \times d}} \frac{1}{2n} \sum_{i=1}^{m} (\langle A_i, X \rangle - y_i)^2$

(similar to linear least squares problem)

↳ If you know that M^* is (symmetric) positive semidefinite

consider an alternate formulation

$$\min_{U \in \mathbb{R}^{d \times d}} \frac{1}{2m} \sum_{i=1}^{m} \left(\langle A_i, UU^T \rangle - y_i \right)^2$$

$A \in \mathbb{R}^{d \times d}$ is positive semi-definite if
- symmetric
- $\forall v \in \mathbb{R}^d$, $v^T A v \geq 0$

$(UU^T)^T = UU^T$

$v^T UU^T v = \|U^T v\|^2 \geq 0$

$\Rightarrow$ Nonlinear least squares

$v^T A v$: $(1 \times d)(d \times d)(d \times 1) = 1 \times 1$

↳ If you further know that M^* has low rank r (can be represented by $< d \times d$ coefficients), can consider

$$\min_{U \in \mathbb{R}^{d \times r}} \frac{1}{2m} \sum_{i=1}^{m} \left(\langle A_i, \underbrace{UU^T}_{d \times d} \rangle - y_i \right)^2$$

UU^T: $(d \times r)(r \times d) = d \times d$

(3) Support vector machine (SVM)

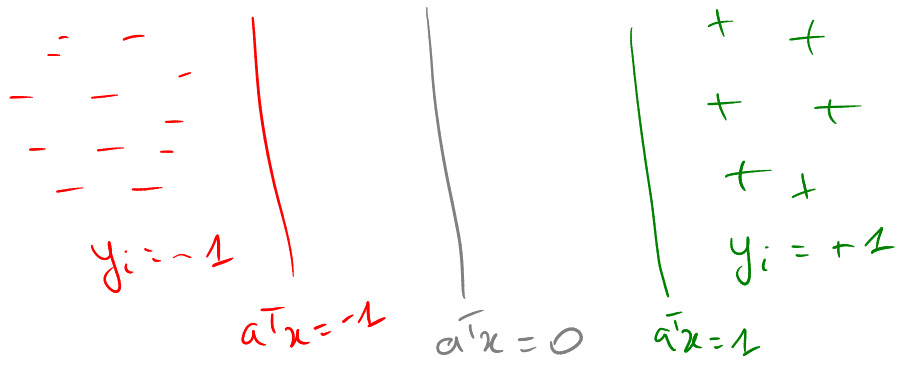
Data: $\{(a_i, y_i)\}_{i=1..m}$ $\quad a_i \in \mathbb{R}^d$ $\quad y_i \in \underbrace{\{-1, 1\}}_{\text{binary output}}$

Goal: Perform binary classification, i.e. learn how to map a_i to y_i

Approach: Consider a linear model $a \mapsto a^T x$ ($\mathbb{R}^d \to \mathbb{R}$) parameterized by $x \in \mathbb{R}^d$

- When $y_i = +1$, want $a^T x \geq 1$
- When $y_i = -1$, want $a^T x \leq -1$

← "Margin of 1"



Problem

$$\min_{x \in \mathbb{R}^d} \underbrace{\frac{1}{m}\sum_{i=1}^{m} \max(1 - y_i a_i^T x, 0)}_{\text{data fitting}} + \underbrace{\lambda \|x\|^2}_{\text{Regularization}}$$

$\max(1 - y_i a_i^T x, 0)$: $\ell(h, y) = \max(1 - yh, 0)$ "Hinge loss"

↳ Variants:
* Add other regularization (ℓ_1)
* Go beyond linear models (kernel SVM)

$a \mapsto a^T x$ $\quad$ $a \mapsto \psi(a)^T x$ with ψ a nonlinear vector function

(4) Multiclass classification using neural networks

↳ Basic feedforward NN

$$a^0 \in \mathbb{R}^{d_0} \mapsto a^1 \in \mathbb{R}^{d_1} \mapsto a^2 \in \mathbb{R}^{d_2} \mapsto \dots \mapsto a^L \in \mathbb{R}^{d_L}$$

L : number of layers

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

σ: (Typically Nonlinear) activation function, from $\mathbb{R}$ to $\mathbb{R}$; ReLU $\sigma(t) = \max(t, 0)$ applied componentwise

W^ℓ: weight matrix $\in \mathbb{R}^{d_\ell \times d_{\ell-1}}$

b^ℓ: bias vector $\in \mathbb{R}^{d_\ell}$

$\forall v \in \mathbb{R}^m$, $\sigma(v) = \begin{bmatrix} \sigma(v_1) \\ \sigma(v_m) \end{bmatrix} \in \mathbb{R}^m$ (ex) $\sigma(t) = t$, $\sigma(t) = \tanh(t)$

Mapping: $a \longmapsto NN(a; x)$, where $NN(a; x)$ is the output of the last layer (a^L) when given a as the input (a^0) and using x as parameters

x: concatenation of $W^1, b^1, \dots, W^L, b^L$ into one (long!) vector

$x \in \mathbb{R}^d$ $\quad d = d_1 d_0 + d_1 + \dots + d_L d_{L-1} + d_L$

Data: $\{(a_i, y_i)\}_{i=1..m}$ $\quad a_i \in \mathbb{R}^{d_0}$

$y_i \in \mathbb{R}^K$ $\quad K$ = number of classes

y_i 1-hot encoding of the class $\quad y_i = \begin{bmatrix} 0 \\ \vdots \\ 1 \leftarrow k : \text{class of } a_i \\ \vdots \\ 0 \end{bmatrix}$

Goal: Use NN to predict y_i from a_i

$NN(a_i; x)$ returns a vector that will be transformed into a vector of probabilities in $\mathbb{R}^K$

$\left(v = \begin{bmatrix} v_1 \\ \vdots \\ v_K \end{bmatrix} \quad \sum_{k=1}^{K} v_k = 1 \right.$
$\left. \forall k \quad v_k \geq 0 \right)$

For simplicity, assume $NN(a_i; x) \in \mathbb{R}^K$

Optimization problem: Minimizing the negative log-likelihood of $NN(a_i; x)$ given y_i + Average over the samples

$\Leftrightarrow$ Maximize the probability of correct classification

$$\min_{x \in \mathbb{R}^d} \frac{1}{n} \sum_{i=1}^{n} \left\{ \log\left(\sum_{k=1}^{K} \exp\left([NN(a_i; x)]_k \right) \right) - \sum_{k=1}^{K} [y_i]_k [NN(a_i; x)]_k \right\}$$

Takeaways

- All examples involve an average of data points (finite-sum structure)
- All involve a continuous objective function

- Additional structure: loss, mapping + possibly regularization

$\Rightarrow$ We want to build algorithms that exploit this structure

More:

- S. J. Wright and B. Recht, Optimization for Data Analysis (2022, Cambridge University Press)

BASICS OF OPTIMIZATION

General setup (no visible dependency data)

$$(P) \quad \min_{x \in \mathbb{R}^d} f(x) \qquad f : \mathbb{R}^d \to \mathbb{R}$$

Def: A solution of (P) aka a global minimum of (P) is a vector $x^* \in \mathbb{R}^d$ such that $f(x^*) \leq f(x) \ \forall x \in \mathbb{R}^d$

Notation: The set of global minima is denoted by

$$\operatorname*{argmin}_{x \in \mathbb{R}^d} f(x) \subseteq \mathbb{R}^d$$

$\operatorname*{argmin}_{x \in \mathbb{R}^d} f(x)$ can be empty ($x \mapsto x$ in $\mathbb{R}$)

$\operatorname*{argmin}_{x \in \mathbb{R}^d} f(x)$ can be finite ($x \mapsto x^2$ in $\mathbb{R}$) for

$\operatorname*{argmin}_{x \in \mathbb{R}^d} f(x)$ can be infinite ($x \mapsto 0$ in $\mathbb{R}$)

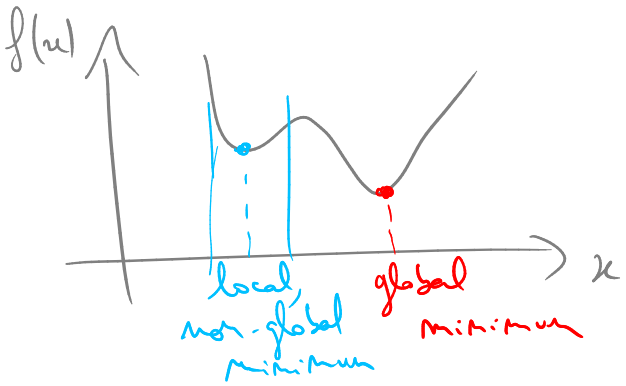
↳ In general, global minima are not tractable (hard or even impossible to compute in finite time)

Def: A vector $\bar{x} \in \mathbb{R}^d$ is a local minimum of (P) if

$\forall x \in \mathbb{R}^d$, $\|x - \bar{x}\|$ sufficiently small, $f(\bar{x}) \leq f(x)$

$\exists \varepsilon > 0, \ \forall x \in \mathbb{R}^d, \ (\|x - \bar{x}\| \leq \varepsilon) \Rightarrow f(\bar{x}) \leq f(x)$

NB: Any global minimum is a local minimum, but the converse is false in general



↳ Local minima still hard to compute in general

⇒ In ML, there are many problem instances for which we can find local minima or approximations of local minima in finite time

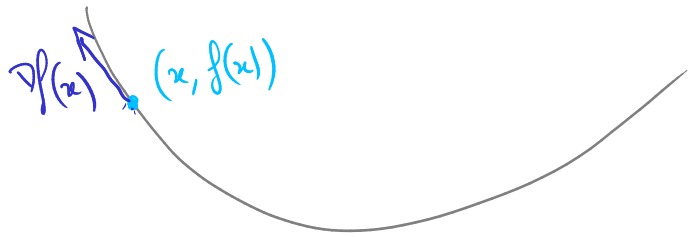
⇒ Local minima are easier to identify than global ones

↳ We will assume properties on the objective function f that guarantee that local minima can be identified, such as differentiability and convexity

Two important properties

- C^1 (aka smooth for some authors)

Def: f is C^1 ("C^1 / continuously differentiable") if $\forall x \in \mathbb{R}^d$, there exists a vector $\nabla f(x) \in \mathbb{R}^d$ (called the gradient of f at x) that represents the derivative of f at x and $\nabla f : \mathbb{R}^d \to \mathbb{R}^d$, $x \mapsto \nabla f(x)$ is continuous



Theorem: (First-order necessary condition)

Let $f: \mathbb{R}^d \to \mathbb{R}$ be C^1.

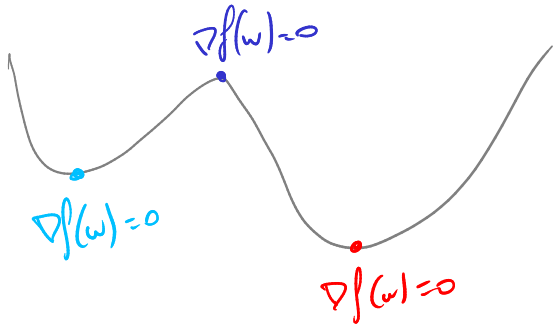
If $\bar{w} \in \mathbb{R}^d$ is a local minimum of f

Then $\nabla f(\bar{w}) = 0$ $\quad (= \|\nabla f(\bar{w})\| = 0)$

→ the gradient characterizes local changes in the function

→ Necessary condition : (IF) ... (THEN) ...

- Does not help you identify local minima
- Helps you identifying points that are not local minima



• = local maximum (yet its gradient is 0)

Any point other than the red/light blue/dark blue ones has a nonzero gradient

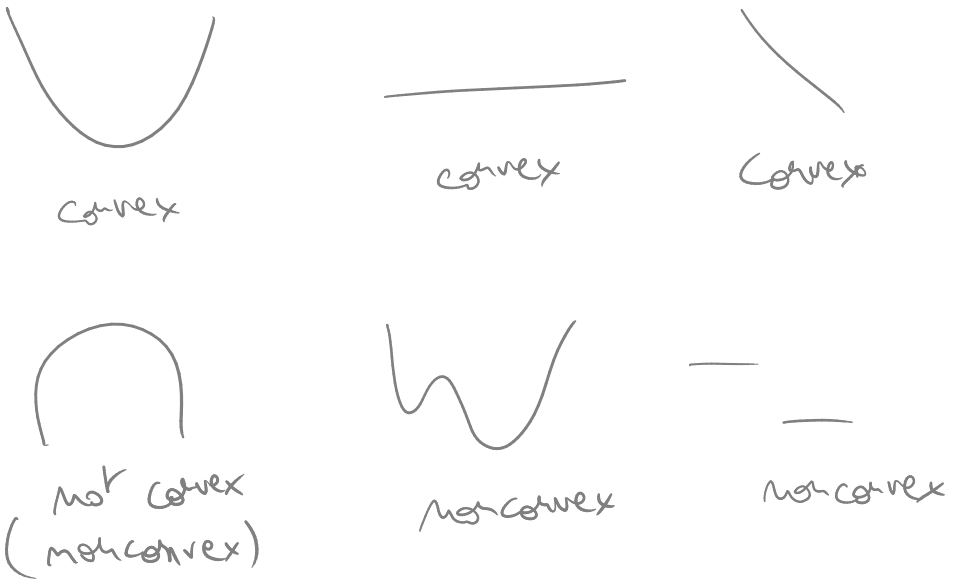
* Convex functions

Def: • $f: \mathbb{R}^d \to \mathbb{R}$ is convex if

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

- $f : \mathbb{R}^d \to \mathbb{R}$ C^1 is convex if

$\forall (x, y) \in (\mathbb{R}^d)^2$, $f(y) \geq f(x) + \nabla f(x)^\top (y - x)$



- l_2 loss, hinge loss are convex

Theorem: Let $f : \mathbb{R}^d \to \mathbb{R}$ C^1 convex and consider (P)

(i) Any local minimum of (P) is a global minimum of (P) (and thus local = global)

(ii) $[\bar{x} \in \mathbb{R}^d$ local minimum of (P)$] \iff [\nabla f(\bar{x}) = 0]$

↑ Necessary and sufficient condition

Takeaway:

- With C^1 + convexity, can find global minima by checking ∇f!
- With C^1, the gradient still helps to identify points that are not local minima.

Exercise: Find a function $f: \mathbb{R}^d \to \mathbb{R}$ C^1 nonconvex such that $\operatorname{argmin}_{x \in \mathbb{R}^d} f(x) = \{ x \in \mathbb{R}^d \mid \nabla f(x) = 0 \}$