Mathematics of Data: From Theory to Computation

Prof. Volkan Cevher
volkan.cevher@epfl.ch

Lecture 1: Data, Models, and Optimization

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

EE-556 (Fall 2020)
License Information for Mathematics of Data Slides

- This work is released under a Creative Commons License with the following terms:

  - **Attribution**
    - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees must give the original authors credit.

  - **Non-Commercial**
    - The licensor permits others to copy, distribute, display, and perform the work. In return, licensees may not use the work for commercial purposes – unless they get the licensor’s permission.

  - **Share Alike**
    - The licensor permits others to distribute derivative works only under a license identical to the one that governs the licensor’s work.

- [Full Text of the License](#)
Logistics

- **Credits:** 5
- **Prerequisites:** Previous coursework in calculus, linear algebra, and probability is required. Familiarity with optimization is useful.
- **Grading:** Homework exercises & exam (cf., syllabus)
- **Moodle:** My courses > Genie electrique et electronique (EL) > Master > EE-556 syllabus & course outline & HW exercises
- **TA’s:** Ahmet Alacaoglu (head TA), Maria Vladarean, Chaehwan Song, Ali Kavis, Mehmet Fatih Sahin, Fabian Latorre, Thomas Sanchez, Thomas Pethick and Igor Krawczuk.
Logistics for online teaching

- **Zoom link for video lectures:**
  
  https://epfl.zoom.us/j/99732416147  
  Passcode: 994779

- **Zoom link for exercise hours:**
  
  https://epfl.zoom.us/j/94022813146  
  Passcode: 076746

- **Switchtube channel for recorded videos:**
  
  https://tube.switch.ch/channels/90d486a0
Outline

- Overview of Mathematics of Data
- Empirical Risk Minimization
- Statistical Learning with Maximum Likelihood Estimators
- Decomposition of error
Recommended preliminary material

- Supplementary slides on
  1. Linear Algebra
  2. Basic Probability
  3. Complexity
The course presents data models, optimization formulations, numerical algorithms, and the associated analysis techniques with the goal of extracting information & knowledge from data while understanding the trade-offs.
A basic statistical learning framework [8]

A statistical learning problem usually consists of three elements.

1. A **generator** that produces samples \( a_i \in \mathbb{R}^p \) of a random variable \( a \) with an unknown probability distribution \( P_a \).

2. A **supervisor** that for each \( a_i \in \mathbb{R}^p \), generates a sample \( b_i \) of a random variable \( B \) with an unknown conditional probability distribution \( P_{B|a} \).

3. A **learning machine** that can respond as any function \( h(a_i) \in \mathcal{H}^o \) of \( a_i \) in some fixed function space \( \mathcal{H}^o \).

Via this framework, we will study classification, regression, and density estimation problems.
A classification example: Cancer prediction

- **Goal:** Assist doctors in diagnosis

- **Generator** $P_a$
  - Genome data $a_t$: [http://genome.ucsc.edu](http://genome.ucsc.edu)

- **Supervisor** $P_{B|a}$
  - Health $b_t = 1$ or $-1$: Cancer or not

- **Learning Machine** $h(a_i)$
  - Data scientist: Mathematics of Data
A regression example: House pricing

\[ a_i = [\text{location, size, orientation, view, distance to public transport, ...}] \]

\[ b_i = [\text{price}] \]

- **Goal:** Assist pricing decisions

- **Generator** \( P_a \)
  - Owners, architects, municipality, constructors

- **Supervisor** \( P_B | a \)
  - House data (homegate, comparis, immobilier...)

- **Learning Machine** \( h(a_i) \)
  - Data scientist: Mathematics of Data

(source: https://www.homegate.ch)
A density estimation example: Image generation

\[ \mathbf{a}_i = [ \text{...images...} ] \]
\[ \mathbf{b}_i = [ \text{...probability...} ] \]

- Goal: Games, denoising, image recovery...

- Generator \( \mathcal{P}_a \)
  - Nature

- Supervisor \( \mathcal{P}_B|_a \)
  - Frequency data

- Learning Machine \( h(\mathbf{a}_i) \)
  - Data scientist: Mathematics of Data

(source: http://mmlab.ie.cuhk.edu.hk/projects/CelebA.html)
Loss function

**Definition (Loss function)**

A **loss function** $L : B \times B \to \mathbb{R}$ on a set is a function that satisfies some or all properties of a metric. We use loss functions in statistical learning to measure the data fidelity $L(h(a), b)$.

**Definition (Metric)**

Let $B$ be a set. A function $d(\cdot, \cdot) : B \times B \to \mathbb{R}$ is a metric if $\forall b_1, 2, 3 \in B$:

(a) $d(b_1, b_2) \geq 0$ for all $b_1$ and $b_2$ \hspace{1cm} (nonnegativity)
(b) $d(b_1, b_2) = 0$ if and only if $b_1 = b_2$ \hspace{1cm} (definiteness)
(c) $d(b_1, b_2) = d(b_2, b_1)$ \hspace{1cm} (symmetry)
(d) $d(b_1, b_2) \leq d(b_1, b_3) + d(b_3, b_2)$ \hspace{1cm} (triangle inequality)

**Remarks:**

- A **pseudo-metric** satisfies (a), (c) and (d) but not necessarily (b).
- **Norms** induce **metrics** while **pseudo-norms** induce **pseudo-metrics**.
- A **divergence** satisfies (a) and (b) but not necessarily (c) or (d)
**Loss function examples**

**Definition (Hinge loss)**

For a binary classification problem, the hinge loss for a score value $b_1 \in \mathbb{R}$ and class label $b_2 \in \pm 1$ is given by $L(b_1, b_2) = \max(0, 1 - b_1 \times b_2)$.

**Definition ($\ell_q$-losses)**

For all $b_1, b_2 \in \mathbb{R}^n \times \mathbb{R}^n$, we can use $L_q(b_1, b_2) = \|b_1 - b_2\|_q^q$, where

$$\|b\|_q^q := \sum_{i=1}^n |b_i|^q \quad \text{for } b \in \mathbb{R}^n \text{ and } q \in [1, \infty)$$

**Definition (1-Wasserstein distance)**

Let $\mu$ and $\nu$ be two probability measures on $\mathbb{R}^d$ and define their couplings as $\Gamma(\mu, \nu) := \{\pi \text{ probability measure on } \mathbb{R}^d \times \mathbb{R}^d \text{ with marginals } \mu, \nu\}$.

$$W_1(\mu, \nu) := \inf_{\pi \in \Gamma(\mu, \nu)} \mathbb{E}(x, y) \sim \pi \|x - y\|$$
A risky, non-parametric reformulation of basic statistical learning

Statistical Learning Model [8]

A statistical learning model consists of the following three elements.

1. A sample of i.i.d. random variables \((a_i, b_i) \in A \times B, i = 1, \ldots, n\), following an unknown probability distribution \(P\).
2. A class (set) \(H^\circ\) of functions \(h : A \rightarrow B\).
3. A loss function \(L : B \times B \rightarrow \mathbb{R}\), measuring data fidelity.

Definition (Risk)

Let \((a, b)\) follow the probability distribution \(P\) and be independent of \((a_1, b_1), \ldots, (a_n, b_n)\). Then, the risk corresponding to any \(h \in H^\circ\) is its expected loss for a chosen loss function \(L\):

\[
R(h) := \mathbb{E}_{(a, b)} [L(h(a), b)].
\]

Statistical learning seeks to find a \(h^\circ \in H^\circ\) that minimizes the population risk, i.e., it solves

\[
h^\circ \in \arg \min_h \{ R(h) : h \in H^\circ \}.
\]

Observations:

- Since \(P\) is unknown, the optimization problem above is intractable.
- Since \(H^\circ\) is often unknown, we might have a mismatched function class in constraints.
Empirical risk minimization (ERM)

We approximate $h^\circ$ by minimizing the empirical average of the loss instead of the risk. That is, we consider

$$h^* \in \arg\min_h \left\{ \frac{1}{n} \sum_{i=1}^n L(h(a_i), b_i) : h \in \mathcal{H} \right\},$$

where $\mathcal{H}$ is our best estimate of the function class $\mathcal{H}^\circ$. Ideally, $\mathcal{H} \equiv \mathcal{H}^\circ$.

Rationale: By the law of large numbers, we can expect that for each $h \in \mathcal{H}$,

$$R(h) := \mathbb{E}_{(a,b)} [L(h(a), b)] \approx \frac{1}{n} \sum_{i=1}^n L(h(a_i), b_i)$$

when $n$ is large enough, with high probability.

Theorem (Strong Law of Large Numbers)

Let $X$ be a real-valued random variable with the finite first moment $\mathbb{E}[X]$, and let $X_1, X_2, ..., X_n$ be an infinite sequence of independent and identically distributed copies of $X$. Then, the empirical average of this sequence $\bar{X}_n := \frac{1}{n} (X_1 + ... + X_n)$ converges almost surely to $\mathbb{E}[X]$: i.e., $P\left( \lim_{n \to \infty} \bar{X}_n = \mathbb{E}[X] \right) = 1$. 
An ERM example

Statistical learning with empirical risk minimization (ERM) [8]

We approximate $h^*$ by minimizing the empirical average of the loss instead of the risk. That is, we consider

$$h^* \in \arg \min_{h \in \mathcal{H}} \left\{ R_n(h) := \frac{1}{n} \sum_{i=1}^{n} L(h(a_i), b_i) \right\}.$$ 

Observations:
- The search space $\mathcal{H}$ is possibly infinite dimensional. It is still not solvable!
  - $\mathcal{H}$ is a non-empty set with a corresponding reproducing kernel Hilbert space.
- We can find numerical solutions as if the problem is parameterized.

Statistical learning with empirical risk minimization (ERM) [8]

In contrast, when the function $h$ has a parametric form $h_x(\cdot)$, we can instead solve

$$x^* \in \arg \min_{x \in \mathcal{X}} \left\{ R_n(h_x) = \frac{1}{n} \sum_{i=1}^{n} L(h_x(a_i), b_i) \right\}.$$
**Basic statistics: Model**

**Parametric estimation model**

A parametric estimation model consists of the following four elements:

1. A **parameter space**, which is a subset \( \mathcal{X} \) of \( \mathbb{R}^p \)
2. A **parameter** \( \mathbf{x}^\natural \), which is an element of the parameter space
3. A class of probability distributions \( \mathcal{P}_X := \{ \mathcal{P}_x : x \in \mathcal{X} \} \)
4. A **sample** \((\mathbf{a}_i, b_i)\), which follows the distribution \( b_i \sim \mathcal{P}_{\mathbf{x}^\natural, \mathbf{a}_i} \in \mathcal{P}_X \)

**Example: Gaussian linear model**

Let \( \mathbf{x}^\natural \in \mathbb{R}^p \). Let \( b_i = \langle \mathbf{a}_i, \mathbf{x}^\natural \rangle + w_i \) for \( i = 1, \ldots, n \), where \( w_i \in \mathbb{R} \) is a Gaussian random variable with zero mean and variance \( \sigma^2 \) (i.e., \( w_i \sim \mathcal{N}(0, \sigma^2) \)).

- Linear model is super general (see Recitation 1).
- Models are often wrong! Robustness vs Performance.
- **Statistical estimation** seeks to approximate \( \mathbf{x}^\natural \), given \( \mathcal{X}, \mathcal{P}_X \), and \( \mathbf{b} \).
Basic statistics: Estimator

**Definition (Estimator)**

An estimator $x^*$ is a mapping that takes $\mathcal{X}$, $\mathcal{P}_\mathcal{X}$, $(a_i, b_i)_{i=1,\ldots,n}$ as inputs, and outputs a value in $\mathcal{X}$.

**Observations:**
- The output of an estimator depends on the sample, and hence, is random.
- The output of an estimator is not necessarily equal to $x^d$.

**Example: The least-squares estimator (LS)**

The least-squares estimator is given by

$$x^*_{LS} \in \text{arg min} \left\{ \frac{1}{n} \sum_{i=1}^{n} (b_i - \langle a_i, x \rangle)^2 : x \in \mathbb{R}^p \right\}.$$
Example: The least-squares estimator (LS)

The least-squares estimator is given by

$$x^*_{LS} \in \arg \min_{x \in \mathbb{R}^p} \left\{ \frac{1}{n} \|b - Ax\|_2^2 : x \in \mathbb{R}^p \right\} = \arg \min \left\{ \frac{1}{n} \sum_{i=1}^{n} (b_i - \langle a_i, x \rangle)^2 : x \in \mathbb{R}^p \right\},$$

where we define $b := (b_1, \ldots, b_n)$ and $a_i$ to be the $i$-th row of $A$.

A statistical learning view of least squares

The LS estimator corresponds to a statistical learning model, for which

- the **sample** is given by $(a_i, b_i) \in \mathbb{R}^p \times \mathbb{R}$, $i = 1, \ldots, n$,
- the **function class** $\mathcal{H}$ is given by $\mathcal{H} := \{h_x(\cdot) := \langle \cdot, x \rangle : x \in \mathbb{R}^p\}$, and
- the **loss function** is given by $L(h_x(a), b) := (b - h_x(a))^2$.

Observation: Given the estimator $x^*_{LS}$, the learning machine outputs $h_{x^*_{LS}}(a) := \langle a, x^*_{LS} \rangle$. 
One way to choose the loss function

Recall the general setting.

A parametric estimation model consists of the following four elements:

1. A parameter space, which is a subset $\mathcal{X}$ of $\mathbb{R}^P$
2. A parameter $x^\natural$, which is an element of the parameter space
3. A class of probability distributions $\mathcal{P}_X := \{ P_x : x \in \mathcal{X} \}$
4. A sample $(a_i, b_i)$, which follows the distribution $b_i \sim P_{x^\natural, a_i} \in \mathcal{P}_X$

**Definition (Maximum-likelihood estimator)**

The maximum-likelihood (ML) estimator is given by

$$ x_{ML}^* \in \arg \min_{x \in \mathcal{X}} \{ L(h_x(a), b) := -\log p_x(b) \}, $$

where $p_x(\cdot)$ denotes the probability density function or probability mass function of $P_x$, for $x \in \mathcal{X}$. 
The least squares estimator: An intuitive derivation

Gaussian linear model

Let \( \mathbf{x}^\natural \in \mathbb{R}^p \). Let \( \mathbf{b} := A \mathbf{x}^\natural + \mathbf{w} \in \mathbb{R}^n \) for some matrix \( A \in \mathbb{R}^{n \times p} \), where \( \mathbf{w} \) is a Gaussian vector with zero mean and covariance matrix \( \sigma^2 I \).

The derivation:

The probability density function \( p_{\mathbf{x}}(\cdot) \) is given by

\[
p_{\mathbf{x}}(\mathbf{b}) = \left( \frac{1}{\sqrt{2\pi \sigma^2}} \right)^n \exp \left( -\frac{1}{2\sigma^2} \| \mathbf{b} - Ax \|^2_2 \right).
\]

Therefore, the maximum likelihood (ML) estimator is defined as

\[
x^*_{\text{ML}} \in \arg \min_x \left\{ -\log p_{\mathbf{x}}(\mathbf{b}) = -\frac{n}{2} \log(2\pi \sigma^2) + \frac{1}{2\sigma^2} \| \mathbf{b} - Ax \|^2_2 : x \in \mathbb{R}^p \right\},
\]

which is equivalent to

\[
x^*_{\text{ML}} \in \arg \min_x \left\{ \frac{1}{n} \| \mathbf{b} - Ax \|^2_2 : x \in \mathbb{R}^p \right\}.
\]

Observations:

- The LS estimator is the ML estimator for the Gaussian linear model.
- The loss function is the quadratic loss.
Statistical learning with ML estimators

- A visual summary: From parametric models to learning machines

\[(a_i, b_i)_{i=1}^n \xrightarrow{\text{modeling parameter } x} P(b_i|a_i, x) \xrightarrow{\text{indepedency identical dist.}} p_x(b) := \prod_{i=1}^nP(b_i|a_i, x)\]

\[\downarrow \text{maximizing w.r.t } x\]

\[a \xrightarrow{\text{maximizing prediction}} x^*_\text{ML} \]

Observations:
- Recall \(x^*_{\text{ML}} \in \arg \min_{x \in X} \{L(h_x(a), b) := -\log p_x(b)\} \).
- Maximizing \(p_x(b)\) gives the ML estimator.
- Maximizing \(p_x(b)\) and minimizing \(-\log p_x(b)\) result in the same solution set.

- See Recitation 1 for more examples in classification, imaging, and quantum tomography
Learning machines result in optimization problems

**Definition (\(M\)-Estimator)**

The learning machine typically has to solve an optimization problem of the following form:

\[
x^*_M \in \arg \min_{x \in X} \{ F(x) \}
\]

for some function \(F\) depending on the sample space \(X\), class of probability distributions \(P_X\), and sample \(b\). The term “\(M\)-estimator” denotes “maximum-likelihood-type estimator” [2].

**Example: The least-absolute deviation estimator (LAD)**

The least-absolute deviation estimator is given by

\[
x^*_{\text{LAD}} \in \arg \min \left\{ \frac{1}{n} \sum_{i=1}^{n} |b_i - \langle a_i, x \rangle| : x \in \mathbb{R}^p \right\}.
\]

**Remark:**

- The LAD estimator is more robust to outliers than the LS estimator.
Practical Issues

Given an estimator $x^* \in \arg\min_{x \in \mathcal{X}} \{F(x)\}$ of $x^\dagger$, we have two questions:

1. Is the formulation reasonable?
2. What is the role of the data size?
Standard approach to checking the fidelity

Standard approach

1. Specify a performance criterion or a (pseudo)metric \( d(x^*, x^\dagger) \) that should be small if \( x^* = x^\dagger \).
2. Show that \( d \) is actually \textit{small in some sense} when \textit{some condition} is satisfied.

Example

Take the \( \ell_2 \)-error \( d(x^*, x^\dagger) := \|x^* - x^\dagger\|_2^2 \) as an example. Then we may verify the fidelity via one of the following ways, where \( \varepsilon \) denotes a small enough number:

1. \( \mathbb{E}[d(x^*, x^\dagger)] \leq \varepsilon \) (expected error),
2. \( P \left( d(x^*, x^\dagger) > t \right) \leq \varepsilon \) for any \( t > 0 \) (consistency),
3. \( \sqrt{n}(x^* - x^\dagger) \) converges in distribution to \( \mathcal{N}(0, I) \) (asymptotic normality),
4. \( \sqrt{n}(x^* - x^\dagger) \) converges in distribution to \( \mathcal{N}(0, I) \) in a local neighborhood (local asymptotic normality).

if \textit{some condition} is satisfied. Such conditions typically revolve around the data size.

○ Recitation 1 explains these concepts in detail.
**Expected error**

**Gaussian linear model**

Let $\mathbf{x}^\natural \in \mathbb{R}^p$ and let $\mathbf{A} \in \mathbb{R}^{n \times p}$. The samples are given by $\mathbf{b} = \mathbf{A}\mathbf{x}^\natural + \mathbf{w}$, where $\mathbf{w}$ is a sample of a Gaussian random vector $\mathbf{w} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$.

What is the performance of the ML estimator

$$\mathbf{x}^\star_{\text{ML}} \in \arg \min_{\mathbf{x} \in \mathbb{R}^p} \left\{ \frac{1}{n} \| \mathbf{b} - \mathbf{A}\mathbf{x} \|^2_2 \right\}.$$  

**Theorem (Performance of the LS estimator [6])**

*If $\mathbf{A}$ is a matrix of independent and identically distributed (i.i.d.) standard Gaussian distributed entries, and if $n > p + 1$, then*

$$\mathbb{E} \left[ \| \mathbf{x}^\star_{\text{ML}} - \mathbf{x}^\natural \|^2_2 \right] = \frac{p}{n - p - 1} \sigma^2 \rightarrow 0 \text{ as } \frac{n}{p} \rightarrow \infty.$$
Performance of the ML estimator

Problem

Let $x^\natural \in \mathbb{R}^p$ be unknown and $b_1, \ldots, b_n$ be i.i.d. samples of a random variable $B$ with p.d.f. $p_{x^\natural}(b) \in P := \{p_X(b) : x \in \mathbb{R}^p\}$. Estimate $x^\natural$ from $b_1, \ldots, b_n$.

Optimization formulation (ML estimator)

$$x^*_{ML} := \arg \min_{x \in \mathbb{R}^p} \left\{ - \frac{1}{n} \sum_{i=1}^{n} \log [p_X(b_i)] \right\} = \arg \min_{x \in \mathbb{R}^p} f(x)$$
Performance of the ML estimator

Problem
Let \( x^\natural \in \mathbb{R}^p \) be unknown and \( b_1, \ldots, b_n \) be i.i.d. samples of a random variable \( B \) with p.d.f. \( p_{x^\natural}(b) \in \mathcal{P} := \{ p_x(b) : x \in \mathbb{R}^p \} \). Estimate \( x^\natural \) from \( b_1, \ldots, b_n \).

Optimization formulation (ML estimator)

\[
 x^\star_{ML} := \arg \min_{x \in \mathbb{R}^p} \left\{ -\frac{1}{n} \sum_{i=1}^{n} \log [p_x(b_i)] \right\} = \arg \min_{x \in \mathbb{R}^p} f(x)
\]

Theorem (Performance of the ML estimator [4, 7])
Under some technical conditions, the random variable \( x^\star_{ML} \) satisfies

\[
 \lim_{n \to \infty} \sqrt{n} \mathbf{J}^{-1/2} \left( x^\star_{ML} - x^\natural \right) \overset{d}{=} Z \sim \mathcal{N}(0, \mathbf{I}), \quad \text{where} \quad \mathbf{J} := -\mathbb{E} \left[ \nabla^2_x \log [p_x(B)] \right] \bigg|_{x=x^\natural}
\]

is the Fisher information matrix associated with one sample.
Performance of the ML estimator

Problem

Let $x^\natural \in \mathbb{R}^p$ be unknown and $b_1, \ldots, b_n$ be i.i.d. samples of a random variable $B$ with p.d.f. $p_{x^\natural}(b) \in \mathcal{P} := \{p_x(b) : x \in \mathbb{R}^p\}$. Estimate $x^\natural$ from $b_1, \ldots, b_n$.

Optimization formulation (ML estimator)

$$x_{ML}^* := \arg \min_{x \in \mathbb{R}^p} \left\{ -\frac{1}{n} \sum_{i=1}^{n} \log [p_x(b_i)] \right\} = \arg \min_{x \in \mathbb{R}^p} f(x)$$

Theorem (Performance of the ML estimator [4, 7])

Under some technical conditions, the random variable $x_{ML}^*$ satisfies

$$\lim_{n \to \infty} \sqrt{n} J^{-1/2} \left( x_{ML}^* - x^\natural \right) \overset{d}{=} Z \sim \mathcal{N}(0, I), \quad \text{where} \quad J := -\mathbb{E} \left[ \nabla^2_x \log [p_x(B)] \right] \bigg|_{x = x^\natural}$$

is the Fisher information matrix associated with one sample. Roughly speaking,

$$\left\| \sqrt{n} J^{-1/2} \left( x_{ML}^* - x^\natural \right) \right\|_2^2 \sim \text{Tr} (I) = p \quad \Rightarrow \quad \left\| x_{ML}^* - x^\natural \right\|_2^2 = O(p/n).$$
Example: ML estimation for quantum tomography

Problem (Quantum tomography)

A quantum system of \(q\) qubits can be characterized by a \textit{density operator}, i.e., a Hermitian positive semidefinite \(X^\# \in \mathbb{C}^{p \times p}\) with \(p = 2^q\).

Let \(b_1, \ldots, b_n\) be samples of independent random variables \(B_1, \ldots, B_n\), with probability distribution

\[
P(\{b_i = k\}) = \text{Tr}\left(A_k X^\#\right), \quad k = 1, \ldots, m,
\]

where \(\{A_1, \ldots, A_m\} \subseteq \mathbb{C}^{p \times p}\) is a \textit{positive operator-valued measure}, i.e., a set of Hermitian positive semidefinite matrices summing to \(I\).

How do we estimate \(X^\#\) given \(\{A_1, \ldots, A_m\}\) and \(b_1, \ldots, b_n\)?

The ML estimator

\[
X^*_{\text{ML}} \in \arg \min_{X \in \mathbb{C}^{p \times p}} \left\{ -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{m} \mathbb{I}\{b_i = k\} \ln \text{Tr} (A_k X) : X = X^H, X \succeq 0 \right\}.
\]
Example: ML estimation for quantum tomography

Performance of ML estimator for quantum tomography with 3 qubits

Numerical result

$\frac{4.5}{\sqrt{n}}$
Caveat Emptor: The ML estimator does not always yield the optimal performance!

Problem
Let \( x^\natural \in \mathbb{R}^p \). Let \( b_i = \langle a_i, x^\natural \rangle + w_i \) for \( i = 1, \ldots, n \), where \( w_i \sim \mathcal{N}(0, 1) \).
Let \( a_i = [0 \ldots 0 1 0 \ldots 0]^T \) be the unit coordinate vector at the \( i^{th} \) coordinate. How do we estimate \( x^\natural \) given \( b \)?

The ML solution
Since \( b \sim \mathcal{N}(x^\natural, I) \), the ML estimator is given by \( x^\natural_{\text{ML}} := b \).

James-Stein estimator [3]
For all \( p \geq 3 \), the James-Stein estimator is given by
\[
    x^\natural_{\text{JS}} := \left( 1 - \frac{p - 2}{\|b\|^2_2} \right) b,
\]
where \( (a)_+ = \max(a, 0) \).

Theorem (Performance comparison: ML vs. James-Stein [3])
For all \( x^\natural \in \mathbb{R}^p \) with \( p \geq 3 \), we have
\[
    \mathbb{E} \left[ \|x^\natural_{\text{JS}} - x^\natural\|^2_2 \right] < \mathbb{E} \left[ \|x^\natural_{\text{ML}} - x^\natural\|^2_2 \right].
\]
In expectation, the performance of the ML estimator is uniformly dominated by the performance of the James-Stein estimator!
Elephant in the room: What happens when $n < p$?

The linear model and the LS estimator when $n < p$

Let $\mathbf{x}^\dagger \in \mathbb{R}^p$ and $\mathbf{A} \in \mathbb{R}^{n \times p}$. The samples are given by $\mathbf{b} = \mathbf{A}\mathbf{x}^\dagger + \mathbf{w}$, where $\mathbf{w}$ denotes the unknown noise.

The LS estimator for $\mathbf{x}^\dagger$ given $\mathbf{A}$ and $\mathbf{b}$ is defined as

$$
\mathbf{x}^\star_{LS} \in \arg \min_{\mathbf{x} \in \mathbb{R}^p} \{ \| \mathbf{b} - \mathbf{A}\mathbf{x} \|_2^2 \}.
$$

The estimation error $\| \mathbf{x}^\star_{LS} - \mathbf{x}^\dagger \|_2$ can be arbitrarily large!

Proposition (The amount of overfitting [1])

Suppose that $\mathbf{A} \in \mathbb{R}^{n \times p}$ is a matrix of i.i.d. standard Gaussian random variables, and $\mathbf{w} = 0$. We have

$$(1 - \epsilon) \left(1 - \frac{n}{p}\right) \| \mathbf{x}^\dagger \|_2^2 \leq \| \mathbf{x}^\star_{\text{candidate}} - \mathbf{x}^\dagger \|_2^2 \leq (1 - \epsilon)^{-1} \left(1 - \frac{n}{p}\right) \| \mathbf{x}^\dagger \|_2^2$$

with probability at least $1 - 2 \exp \left[-(1/4)(p - n)\epsilon^2\right] - 2 \exp \left[-(1/4)p\epsilon^2\right]$, for all $\epsilon > 0$ and $\mathbf{x}^\dagger \in \mathbb{R}^p$. 

Role of computation

Observations:

○ The estimator $x^*$'s performance, e.g., $\|x^* - x^\dagger\|^2_2$, depends on the data size $n$.

○ Evaluating $\|x^* - x^\dagger\|^2_2$ is not enough for evaluating the performance of a Learning Machine.

  ▶ We can only *numerically approximate* the solution of

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

○ We use algorithms to *numerically approximate* $x^*$.

Practical performance

Denote the numerical approximation by an algorithm at time $t$ by $x^t$.

The practical performance at time $t$ using $n$ data samples is determined by

$$\|x^t - x^\dagger\|^2_2 \leq \underbrace{\|x^t - x^*\|^2_2}_{\varepsilon(t,n)} + \underbrace{\|x^* - x^\dagger\|^2_2}_{\varepsilon(n)},$$

where $\varepsilon(n)$ denotes the statistical error, $\varepsilon(t)$ is the numerical error, and $\overline{\varepsilon}(t,n)$ denotes the total error of the Learning Machine.
Role of computation

**Observations:**
- The estimator $x^*$'s performance, e.g., $\|x^* - x\|^2$, depends on the data size $n$.
- Evaluating $\|x^* - x^\dagger\|^2$ is not enough for evaluating the performance of a Learning Machine.
  - We can only *numerically approximate* the solution of
    $$x^* \in \arg \min_{x \in \mathbb{R}^p} \{ F(x) \} .$$
- We use algorithms to *numerically approximate* $x^*$.

**Practical performance**

Denote the numerical approximation by an algorithm at time $t$ by $x_t$.

The practical performance at time $t$ using $n$ data samples is determined by

$$\|x_t - x^\dagger\|^2 \leq \|x_t - x^*\|^2 + \|x^* - x^\dagger\|^2,$$

where $\varepsilon(n)$ denotes the statistical error, $\epsilon(t)$ is the numerical error, and $\bar{\varepsilon}(t,n)$ denotes the total error of the Learning Machine.
Role of computation

Observations:

- The estimator $x^*$'s performance, e.g., $\|x^* - x^\dagger\|^2$, depends on the data size $n$.
- Evaluating $\|x^* - x^\dagger\|^2$ is not enough for evaluating the performance of a Learning Machine.
  - We can only \textit{numerically approximate} the solution of
    
    \[ x^* \in \arg \min_{x \in \mathbb{R}^p} \left\{ F(x) \right\}. \]
  - We use algorithms to \textit{numerically approximate} $x^*$.

Practical performance

Denote the numerical approximation by an algorithm at time $t$ by $x^t$.

The practical performance at time $t$ using $n$ data samples is determined by

\[
\|x^t - x^\dagger\|^2 \leq \underbrace{\|x^t - x^*\|^2}_{\varepsilon(t)} + \underbrace{\|x^* - x^\dagger\|^2}_{\bar{\varepsilon}(t,n)},
\]

where $\varepsilon(n)$ denotes the statistical error, $\varepsilon(t)$ is the numerical error, and $\bar{\varepsilon}(t,n)$ denotes the total error of the Learning Machine.
Peeling the onion

Models
Let \( d(\cdot, \cdot) : \mathcal{H}^0 \times \mathcal{H}^0 \to \mathbb{R}^+ \) be a metric in an extended function space \( \mathcal{H}^0 \) that includes \( \mathcal{H} \); i.e., \( \mathcal{H} \subseteq \mathcal{H}^0 \). Let

1. \( h^0 \in \mathcal{H}^0 \) be the true, expected risk minimizing model
2. \( h^\natural \in \mathcal{H} \) be the solution under the assumed function class \( \mathcal{H} \subseteq \mathcal{H}^0 \)
3. \( h^* \in \mathcal{H} \) be the estimator solution
4. \( h^t \in \mathcal{H} \) be the numerical approximation of the algorithm at time \( t \)

Practical performance

\[
\bar{\varepsilon}(t, n) \leq d(h^t, h^0) - d(h^t, h^*) + d(h^*, h^\natural) + d(h^\natural, h^0),
\]
where \( \bar{\varepsilon}(t, n) \) denotes the total error of the Learning Machine. We can try to

1. reduce the optimization error with computation
2. reduce the statistical error with more data samples, with better estimators, and with prior information
3. reduce the model error with flexible or universal representations
Estimation of parameters vs estimation of risk

Recall the general setting

Let \( R(h_x) = \mathbb{E}L(h_x(a), b) \) be the risk function and \( R_n(h_x) = \frac{1}{n} \sum_{i=1}^{n} L(h_x(a_i), b_i) \) be the empirical estimate. Let \( \mathcal{X} \subseteq \mathcal{X}^0 \) be parameter domains, where \( \mathcal{X} \) is known. Define

1. \( x^o \in \arg\min_{x \in \mathcal{X}^o} R(h_x) \): true minimum risk model
2. \( x^\natural \in \arg\min_{x \in \mathcal{X}} R(h_x) \): assumed minimum risk model
3. \( x^* \in \arg\min_{x \in \mathcal{X}} R_n(h_x) \): ERM solution
4. \( x^t \): numerical approximation of \( x^* \) at time \( t \)

Nomenclature

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( R_n(\cdot) )</td>
<td>training error</td>
</tr>
<tr>
<td>( R(\cdot) )</td>
<td>test error</td>
</tr>
<tr>
<td>( R(x^\natural) - R(x^o) )</td>
<td>modeling error</td>
</tr>
<tr>
<td>( R(x^*) - R(x^\natural) )</td>
<td>excess risk</td>
</tr>
<tr>
<td>( \sup_{x \in \mathcal{X}}</td>
<td>R(x) - R_n(x)</td>
</tr>
<tr>
<td>( R_n(x^t) - R_n(x^*) )</td>
<td>optimization error</td>
</tr>
</tbody>
</table>

\[
\begin{array}{cccc}
\mathcal{X} & \rightarrow & \mathcal{X}^0 & \uparrow \quad n \uparrow \quad p \uparrow \\
\text{Training error} & \searrow & \searrow & \searrow \\
\text{Excess risk} & \searrow & \searrow & \searrow \\
\text{Generalization error} & \searrow & \searrow & \searrow \\
\text{Modeling error} & \searrow & \searrow & \leftrightarrow \\
\text{Time} & \searrow & \uparrow & \uparrow \\
\end{array}
\]
Peeling the onion (risk minimization setting)

Models
Let $\mathcal{X} \subseteq \mathcal{X}^\circ$ be parameter domains, where $\mathcal{X}$ is known. Define

1. $x^\circ \in \arg\min_{x \in \mathcal{X}^\circ} R(h_x)$: true minimum risk model
2. $x^\natural \in \arg\min_{x \in \mathcal{X}} R(h_x)$: assumed minimum risk model
3. $x^* \in \arg\min_{x \in \mathcal{X}} R_n(h_x)$: ERM solution
4. $x^t$: numerical approximation of $x^*$ at time $t$

Practical performance

$$R(x^t) - R(x^\circ) \leq R_n(x^t) - R_n(x^*) + 2 \sup_{x \in \mathcal{X}} |R(x) - R_n(x)| + R(x^\natural) - R(x^\circ)$$

where $\bar{\epsilon}(t, n)$ denotes the total error of the Learning Machine. We can try to

1. reduce the optimization error with computation
2. reduce the generalization error with regularization or more data
3. reduce the model error with flexible or universal representations
How does the generalization error depend on the data size and dimension?

**Theorem ([5])**

Let \( h_x : \mathbb{R}^p \rightarrow \mathbb{R}, h_x(a) = x^T a \) and let \( L(h_x(a), b) = \max(0, 1 - b \cdot x^T a) \) be the hinge loss. Let \( \mathcal{X} := \{ x \in \mathbb{R}^p : ||x|| \leq \lambda \} \). Suppose that \( ||a|| \leq \sqrt{p} \) almost surely (boundedness).

Roughly speaking, with some probability that we can control, the following holds:

\[
\sup_{x \in \mathcal{X}} |R_n(x) - R(x)| = \mathcal{O} \left( \sqrt{\frac{p\lambda}{n}} \right)
\]
Wrap up!

- See you at Recitation 1 on Friday!
*Peeling the onion (risk minimization setting) - Decomposition details

\[ R(x^t) - R(x^\natural) = R(x^t) - R_n(x^t) + R_n(x^t) - R_n(x^\star) + R_n(x^\star) - R_n(x^\natural) + R_n(x^\natural) - R(x^\natural) \leq 0 \]

\[ \leq R_n(x^t) - R_n(x^\star) + \underbrace{R(x^t) - R_n(x^t) + R_n(x^\natural) - R(x^\natural)}_{\leq 0} \]

\[ 2 \sup_{x \in X} |R_n(x) - R(x)| \]

\[ R(x^t) - R(x^\circ) = R(x^t) - R(x^\natural) + R(x^\natural) - R(x^\circ) \]

\[ \leq R_n(x^t) - R_n(x^\star) + 2 \sup_{x \in X} |R_n(x) - R(x)| + R(x^\natural) - R(x^\circ) \]
References

Compressible distributions for high-dimensional statistics.

*Robust Statistics*.

Estimation with quadratic loss.

*Asymptotic methods in Statistical Decision Theory*.

*Foundations of Machine Learning*.

The squared-error of generalized LASSO: A precise analysis.
2013.
arXiv:1311.0830v2 [cs.IT].
*Asymptotic Statistics.*  

An overview of statistical learning theory.  