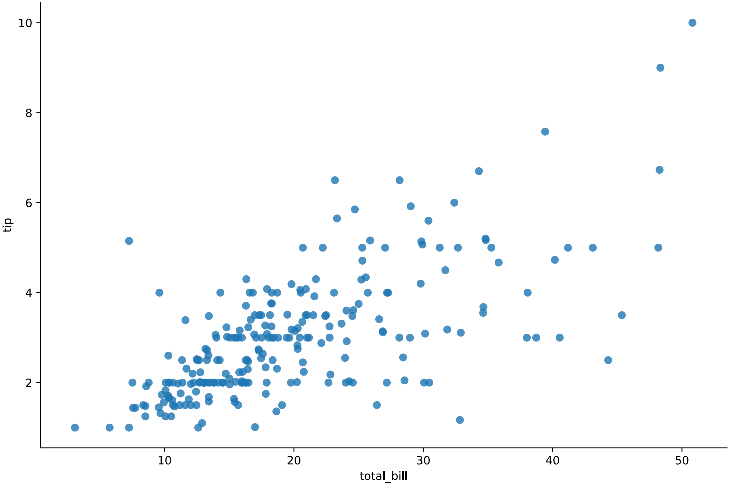
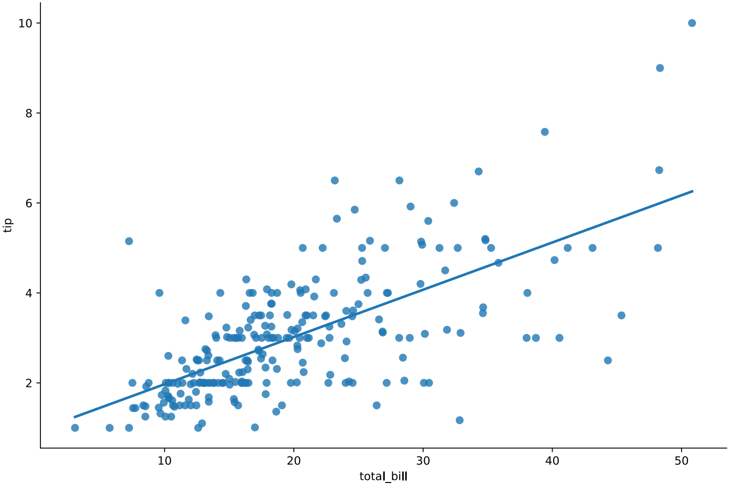
SD TSIA 204
Linear Models
Intro to linear models

Ekhiñe Irurozki
Télécom Paris

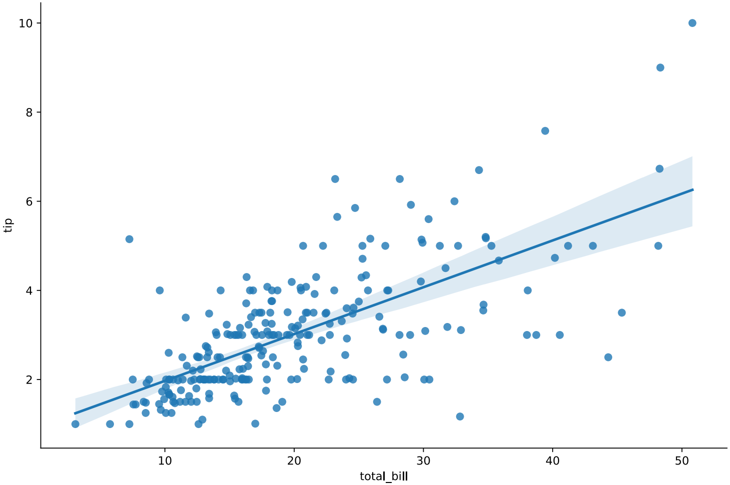
A 2D starting example



A 2D starting example



A 2D starting example



Notation interpretation

- $n = 244$
- $p = 1$
- y_i : tip let by the i-th customer
- x_i : total bill payed by the i-th customer
- y : the observation is the tips, dependent variable
- x : the feature/covariate, price of the bill, independent variable

Linear model / Linear regression hypothesis : assume that the price of the bill and the tip let are linearly correlated

Exo : use `describe()` from `Pandas` to get a rough data summary

Three questions to be covered : modeling, learning and predicting

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression
# Generate example data
np.random.seed(42)
X = np.random.rand(20, 1)*10 # Independent variable
y = 2 * X + 3 + np.random.randn(20, 1) # Dependent variable
# Fit linear regression model
model = LinearRegression()
model.fit(X, y)
# Predict y values using the model
X_new = np.linspace(0, 10, 100).reshape(-1, 1)
y_pred = model.predict(X_new)
# Create a scatter plot of the data points
plt.scatter(X, y, label='Data Points')
# Plot the linear regression line
plt.plot(X_new, y_pred, color='red', label='Linear Regression Line')
plt.xlabel('X')
plt.ylabel('y')
```

Modeling I, the 1D case

$$\text{Given a sample : } \quad (y_i, x_i), \text{ for } i = 1, \dots, n$$

Linear model or linear regression hypothesis assume :

$$\boxed{y_i \approx \theta_0^\star + \theta_1^\star x_i}$$

Model coefficients

- $\theta_0^\star$: intercept (unknown)
- $\theta_1^\star$: slope (unknown)

<u>Rem</u>: both parameters are unknown from the statistician

Data

- y is an **observation** or a variable to explain
- x is a **feature** or a covariate

Modeling II

Probabilistic model. Let us give a precise meaning to the sign $\approx$:

$$
\begin{aligned}
& y_i = \theta_0^\star + \theta_1^\star x_i + \varepsilon_i, \\
& \varepsilon_i \overset{i.i.d}{\sim} \varepsilon, \text{ for } i = 1, \ldots, n \\
& \mathbb{E}(\varepsilon) = 0
\end{aligned}
$$

where i.i.d. means "independent and identically distributed"

Interpretation : $\varepsilon_i = y_i - \theta_0^\star - \theta_1^\star x_i$: represent the error between the theoretical model and the observations, represented by random variables ε_i centered (often referred to as **white noise**).

<u>Rem</u>: motivation for the random nature of the noise – measurement noise, transmission noise, in-population variability, etc.

Modeling III

$$y_i = \theta_0^\star + \theta_1^\star x_i + \varepsilon_i$$

We call

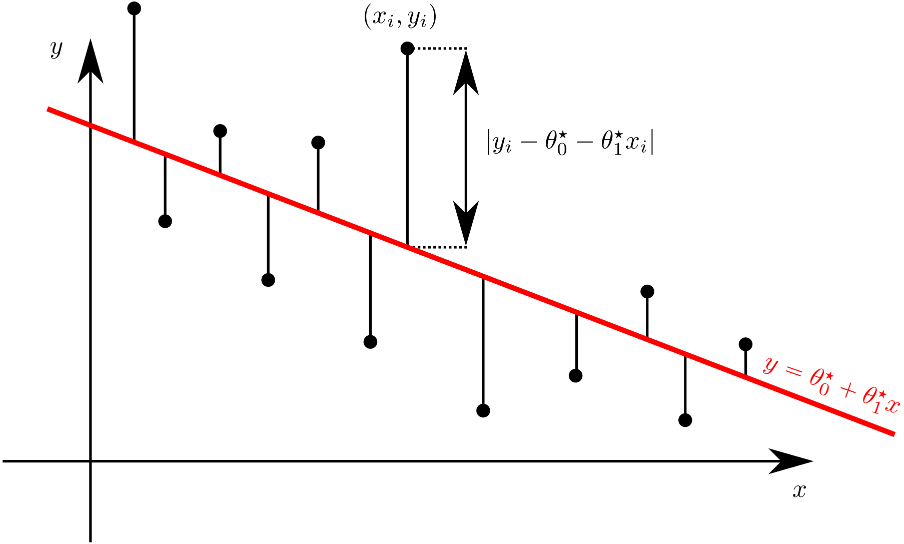
- **intercept** the scalar $\theta_0^\star$ (: *ordonnée à l'origine*)
- **slope** the scalar $\theta_1^\star$ (: *pente*)

Our **goal in the learning stage** is to estimate $\theta_0^\star$ and $\theta_1^\star$ (unknown) by $\widehat{\theta}_0$ and $\widehat{\theta}_1$ relying on observations (y_i, x_i) for $i = 1, \dots, n$

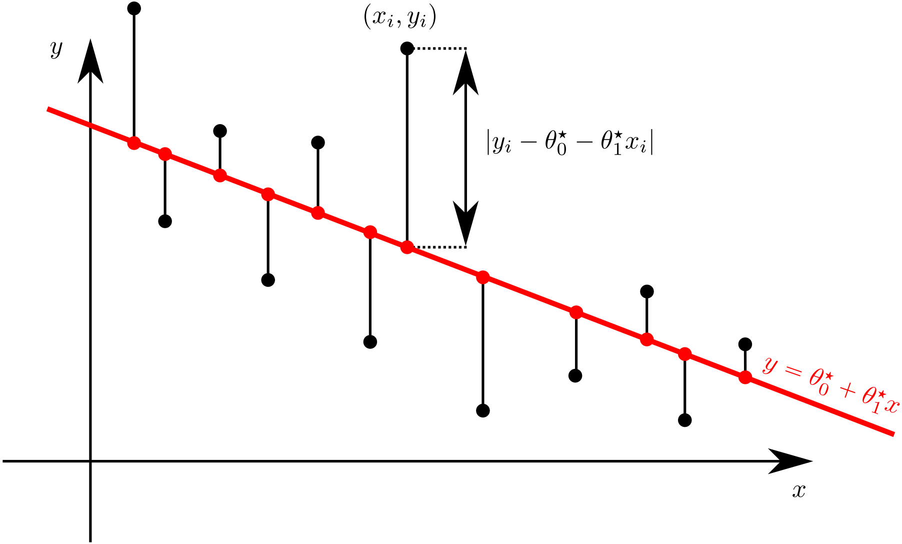
<u>Rem</u>: The "hat" notation is classical in statistics for referring to estimators

In **prediction time** $\widehat{y}_i = \widehat{\theta}_0 + \widehat{\theta}_1 x_i$

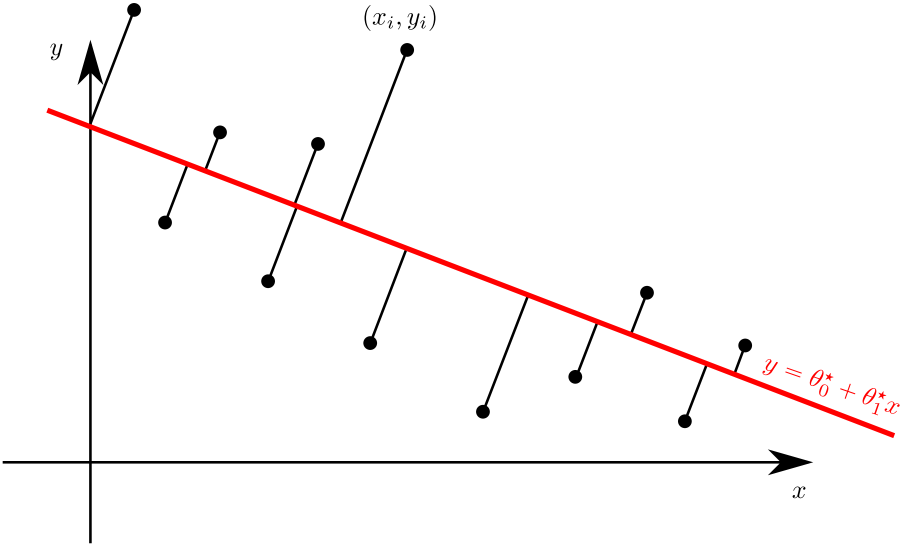
Least squares : visualization



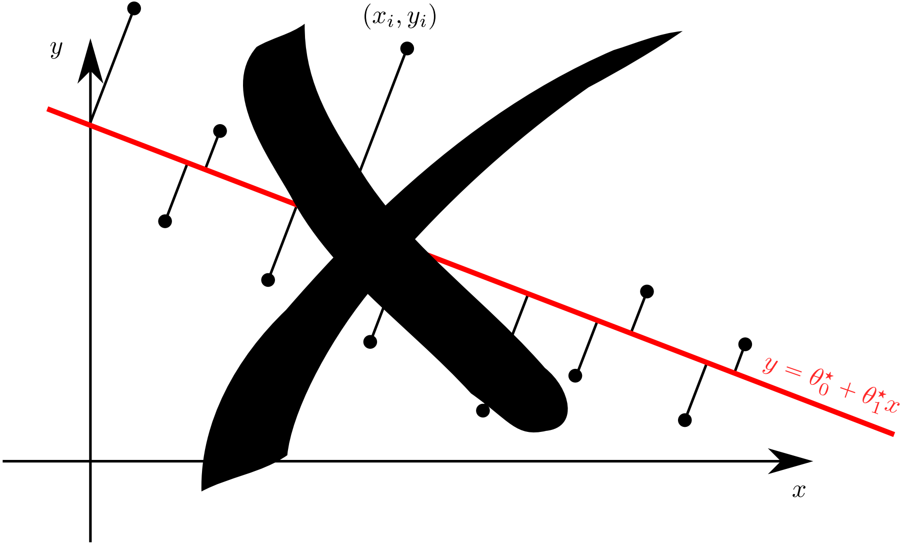
Least squares : visualization



(Total) Least squares : visualization



(Total) Least squares : visualization



Learning : mathematical formulation of Least squares

The **least squares** estimator is defined as :

$$(\widehat{\theta}_0, \widehat{\theta}_1) \in \operatorname{argmin}_{(\theta_0,\theta_1)\in\mathbb{R}^2} \sum_{i=1}^{n} \left(y_i - \theta_0 - \theta_1 x_i\right)^2$$

- Differentiate between θ^*, θ and $\widehat{\theta}$!!!!!
- it is also referred to as “ordinary least squares” (OLS)
- an original motivation for the squares is computational : first order conditions only require solving a linear system
- a solution always exists : minimizing a **coercive** continuous function (coercive : $\lim_{\|x\|\to+\infty} f(x) = +\infty$)

<u>Rem</u>: write « $\in$ argmin » as long as you do not know if the solution is unique

Least square authorship (controversial)

FIGURE – Adrien-Marie Legendre and Carl Friedrich Gauss

Historical / robust detour

The **least absolute deviation** (LAD) estimator reads :

$$(\widehat{\theta}_0, \widehat{\theta}_1) \in \operatorname{argmin}_{(\theta_0,\theta_1)\in\mathbb{R}^2} \sum_{i=1}^{n} |y_i - \theta_0 - \theta_1 x_i|$$

Rem: hard to compute without computer ; requires an optimization solver for non-smooth function (or a Linear Programming solver)

Rem: more robust to outliers (: *données aberrantes*)

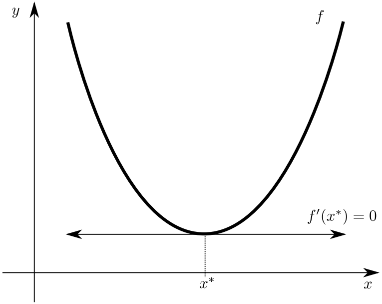
Least absolute deviation authorship

FIGURE – Ruđer Josip Bošković and Pierre-Simon de Laplace

Existence and uniqueness of the solution

Existence of a Local minimum : first order condition

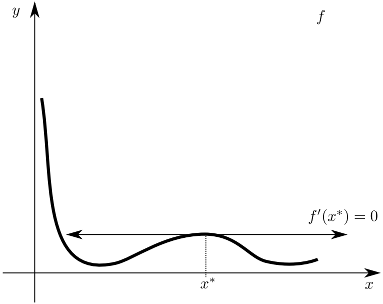
Fermat's rule Theorem If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, *i.e.* $\nabla f(x^*) = 0$.



Existence and uniqueness of the solution

Existence of a Local minimum : first order condition

Fermat's rule Theorem If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, *i.e.* $\nabla f(x^*) = 0$.

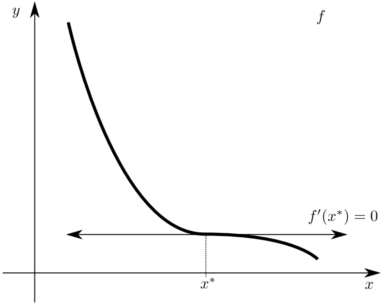


Rem: sufficient condition when f is strongly convex!

Existence and uniqueness of the solution

Existence of a Local minimum : first order condition

Fermat's rule Theorem If f is differentiable, then at a local minimum x^* the gradient of f vanishes at x^*, *i.e.* $\nabla f(x^*) = 0$.



Rem: sufficient condition when f is strongly convex !

The Hessian Matrix and Gradients

The **gradient** ∇f is a vector of first-order partial derivatives :

$$\nabla f = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \vdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

The **Hessian Matrix H** of f is a square matrix of second-order partial derivatives :

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

The minimizer is unique $\Leftrightarrow$ f its strictly convex

f is quadratic $\implies$ f is convex.

$f(\widehat{\boldsymbol{\theta}})$ strictly convex $\Leftrightarrow \nabla^2 f(\widehat{\boldsymbol{\theta}})$ positive definite $\Leftrightarrow det(\nabla^2 f(\widehat{\boldsymbol{\theta}})) > 0$

Back to least squares

$$\widehat{\boldsymbol{\theta}} = (\widehat{\theta}_0, \widehat{\theta}_1) \in \operatorname{argmin}_{(\theta_0,\theta_1)\in\mathbb{R}^2} \frac{1}{2}\sum_{i=1}^{n}(y_i - \theta_0 - \theta_1 x_i)^2$$

For least squares, minimize the function of two variables :

$$f(\theta_0, \theta_1) = f(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \theta_0 - \theta_1 x_i)^2$$

First order condition / Fermat's rule :

$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \widehat{\theta}_0 - \widehat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \widehat{\theta}_0 - \widehat{\theta}_1 x_i) x_i = 0 \end{cases}$$

Calculus continued

$$\text{Usual mean notation : } \overline{x}_n = \frac{1}{n}\sum_{i=1}^{n} x_i \text{ and } \overline{y}_n = \frac{1}{n}\sum_{i=1}^{n} y_i$$

With that, Fermat's rule states (dividing by n) :

$$\begin{cases} \frac{\partial f}{\partial \theta_0}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \widehat{\theta}_0 - \widehat{\theta}_1 x_i) = 0 \\ \frac{\partial f}{\partial \theta_1}(\widehat{\boldsymbol{\theta}}) = \sum_{i=1}^{n}(y_i - \widehat{\theta}_0 - \widehat{\theta}_1 x_i)x_i = 0 \end{cases}$$

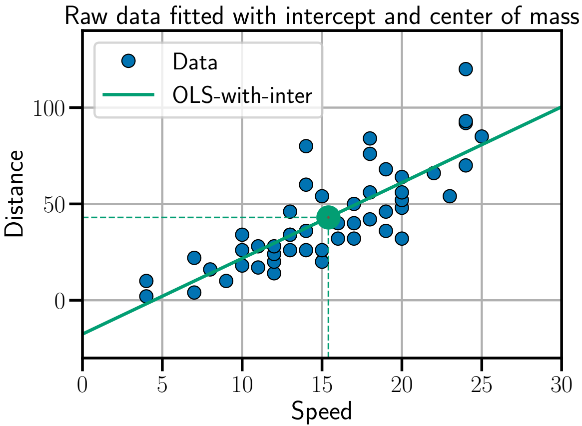
$$\Leftrightarrow$$

$$\begin{cases} \widehat{\theta}_0 = \overline{y}_n - \widehat{\theta}_1 \overline{x}_n & \text{(CNO1)} \\ \widehat{\theta}_1 = \dfrac{\sum_{i=1}^{n}(x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^{n}(x_i - \overline{x}_n)^2} & \text{(CNO2)} \end{cases}$$

Exo : Show that the solution to the OLS is unique iff $Var(x) \neq 0$

Center of gravity and interpretation

$$(\text{CNO1}) \Leftrightarrow (\overline{x}_n, \overline{y}_n) \in \{(x,y) \in \mathbb{R}^2 : y = \widehat{\theta}_0 + \widehat{\theta}_1 x\}$$



- $\overline{speed} = 15.4$
- $\overline{dist} = 42.98$
- $\widehat{\theta}_0 = -17.579095$ intercept (negative !)
- $\widehat{\theta}_1 = 3.932409$ slope

Physical interpretation : the cloud of points' center of gravity belongs to the (estimated) regression line

Vector formulation

<u>Notation</u> : $\mathbf{x} = (x_1, \ldots, x_n)^\top$ and $\mathbf{y} = (y_1, \ldots, y_n)^\top$

$$\text{(CNO2)} \Leftrightarrow \widehat{\theta}_1 = \frac{\sum_{i=1}^n (x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sum_{i=1}^n (x_i - \overline{x}_n)^2}$$

$$\text{(CNO2)} \Leftrightarrow \widehat{\theta}_1 = \mathrm{corr}_n(\mathbf{x}, \mathbf{y}) \cdot \frac{\sqrt{\mathrm{var}_n(\mathbf{y})}}{\sqrt{\mathrm{var}_n(\mathbf{x})}}$$

where $$\mathrm{corr}_n(\mathbf{x}, \mathbf{y}) = \frac{\frac{1}{n}\sum_{i=1}^n (x_i - \overline{x}_n)(y_i - \overline{y}_n)}{\sqrt{\mathrm{var}_n(\mathbf{x})}\sqrt{\mathrm{var}_n(\mathbf{y})}}$$

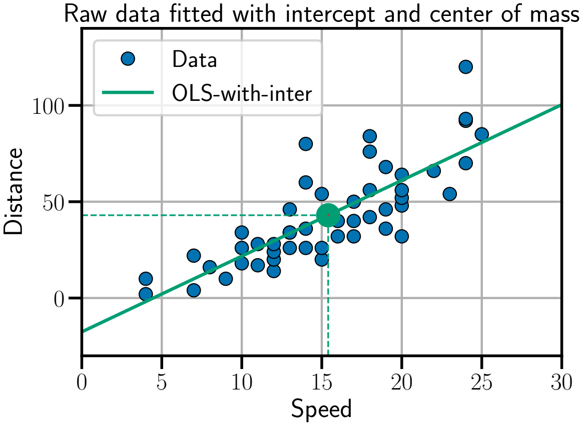
and $$\mathrm{var}_n(\mathbf{z}) = \frac{1}{n}\sum_{i=1}^n (z_i - \overline{z}_n)^2 \text{ (for any } \mathbf{z} = (z_1, \ldots, z_n)^\top\text{)}$$

respectively **empirical correlation**, **empirical variances**

cars example

Braking distance for cars as a sunction of the speed

Line slope : $\mathrm{corr}_n(\mathbf{x},\mathbf{y}) \cdot \frac{\sqrt{\mathrm{var}_n(\mathbf{y})}}{\sqrt{\mathrm{var}_n(\mathbf{x})}} = 3.932409.$



Centering

Centered model :

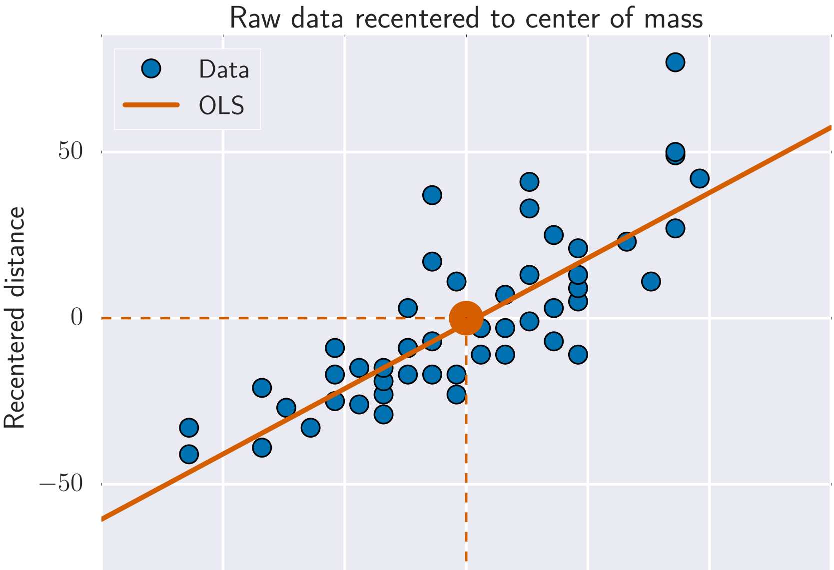
$$\text{Write for any } i = 1, \dots, n : \begin{cases} x'_i = x_i - \overline{x}_n \\ y'_i = y_i - \overline{y}_n \end{cases} \quad \Leftrightarrow \begin{cases} \mathbf{x}' = \mathbf{x} - \overline{x}_n \mathbf{1}_n \\ \mathbf{y}' = \mathbf{y} - \overline{y}_n \mathbf{1}_n \end{cases}$$

and $\mathbf{1}_n = (1, \dots, 1)^\top \in \mathbb{R}^n$, then solving the OLS with $(\mathbf{x}', \mathbf{y}')$ leads to

$$\begin{cases} \widehat{\theta}'_0 = 0 \\ \widehat{\theta}'_1 = \dfrac{\dfrac{1}{n}\displaystyle\sum_{i=1}^n x'_i y'_i}{\dfrac{1}{n}\displaystyle\sum_{i=1}^n x'^2_i} \end{cases}$$

<u>Rem</u>: equivalent to choosing the cloud of points' center of mass as origin, *i.e.* $(\overline{x}'_n, \overline{y}'_n) = (0, 0)$

Centering (II)



Centering and interpretation

Consider the coefficient $\widehat{\theta}'_1$ ($\widehat{\theta}'_0 = 0$) for centered points $\mathbf{y}', \mathbf{x}'$, then :

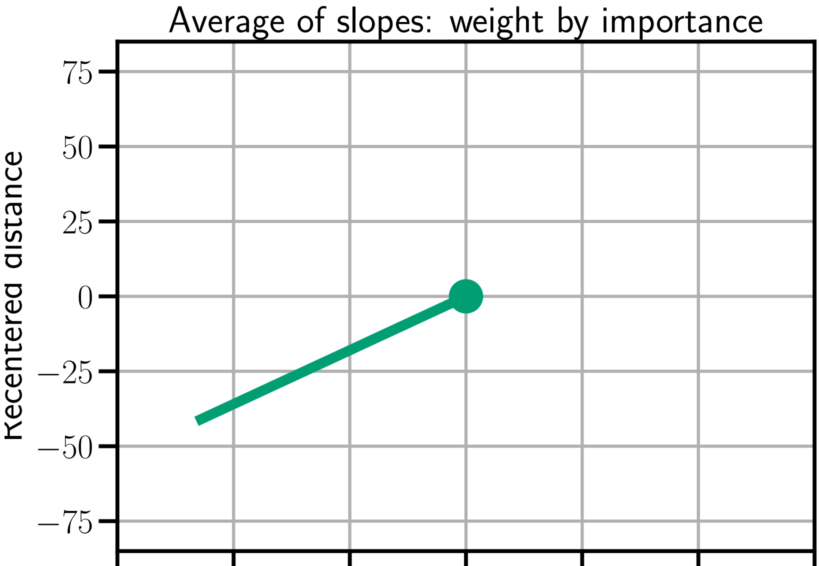
$$\widehat{\theta}'_1 \in \text{argmin}_{\theta_1} \sum_{i=1}^{n} (y'_i - \theta_1 x'_i)^2 = \text{argmin}_{\theta_1} \sum_{i=1}^{n} x'^2_i \left(\frac{y'_i}{x'_i} - \theta_1 \right)^2$$

<u>Interpretation</u> : $\widehat{\theta}'_1$ is a weighted average of the slopes $\frac{y'_i}{x'_i}$

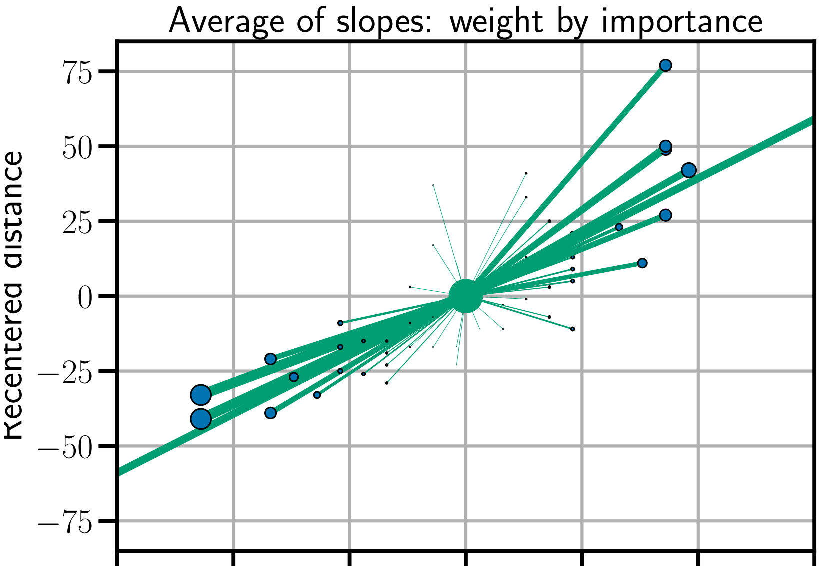
$$\widehat{\theta}'_1 = \frac{\displaystyle\sum_{i=1}^{n} x'^2_i \frac{y'_i}{x'_i}}{\displaystyle\sum_{j=1}^{n} x'^2_j}$$

<u>Influence of extreme points</u> : weights proportional to x'^2_i ; connected to the **leverage** (🇫🇷 : *levier*) effect

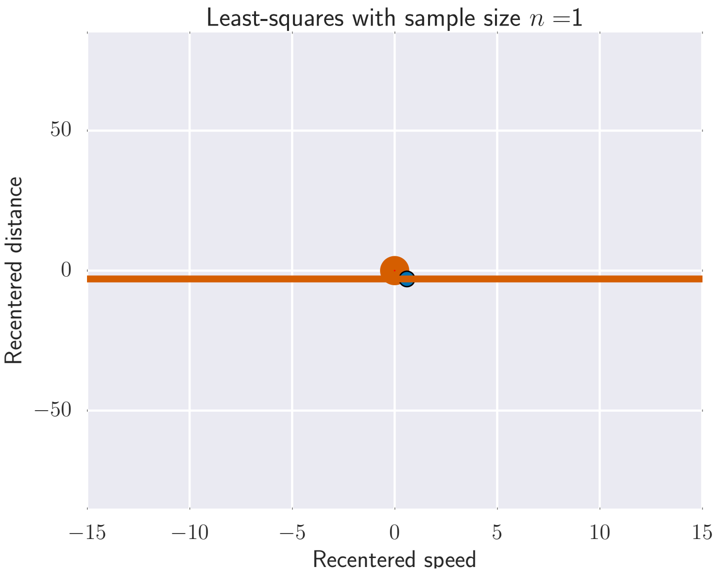
Extreme points – leverage effect



Extreme points – leverage effect



Extreme points – leverage effect (II)



Centering + scaling (standardization)

Centered-scaled model :

$$\forall i = 1, \dots, n : \begin{cases} x_i'' = (x_i - \overline{x}_n)/\sqrt{\mathrm{var}_n(\mathbf{x})} \\ y_i'' = (y_i - \overline{y}_n)/\sqrt{\mathrm{var}_n(\mathbf{y})} \end{cases} \quad \Leftrightarrow \begin{cases} \mathbf{x}'' = \dfrac{\mathbf{x} - \overline{x}_n \mathbf{1}_n}{\sqrt{\mathrm{var}_n(\mathbf{x})}} \\ \mathbf{y}'' = \dfrac{\mathbf{y} - \overline{y}_n \mathbf{1}_n}{\sqrt{\mathrm{var}_n(\mathbf{y})}} \end{cases}$$

Solving OLS with $(\mathbf{x}'', \mathbf{y}'')$ then

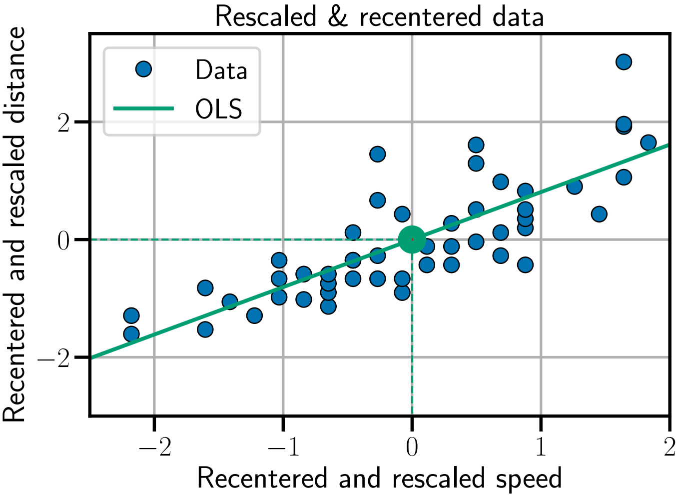
$$\begin{cases} \widehat{\theta}_0'' = 0 \\ \widehat{\theta}_1'' = \dfrac{1}{n} \displaystyle\sum_{i=1}^{n} x_i'' y_i'' \end{cases}$$

Rem: equivalent to choosing the points cloud center of mass as origin and normalize $\mathbf{x}$ and $\mathbf{y}$ to have unit **empirical norm** $\|\cdot\|_n$:

$$\|\mathbf{x}''\|_n^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i'')^2 = 1$$

$$\|\mathbf{y}''\|_n^2 = \frac{1}{n} \sum_{i=1}^{n} (y_i'')^2 = 1$$

Centering + scaling



When/why preprocessing ?

Centering **y** or using an intercept (or adding a constant feature) is equivalent

Rem: for sparse (🇫🇷 : *creux*) cases centering **y** adding a constant feature could be preferred

Scaling features is important :

- if you want to interpret the coefficients' amplitude in regression (better solution : t-tests)
- if you want to penalize or regularize coefficients (*c.f.* Lasso, Ridge, etc.) a single scale is needed
- for computing reasons (*e.g.* store scaling to improve efficiency, etc.)

Rem: in practice centering/scaling is useful for **estimation** not so much for **prediction** (see next courses)

What happens with the logarithm scaling ?

Centering with `Python`

Use centering classes from `sklearn`, see `preprocessing` :

http://scikit-learn.org/stable/modules/preprocessing.html

```
from sklearn import preprocessing

scaler = preprocessing.StandardScaler().fit(X)

print(np.isclose(scaler.mean_, np.mean(X)))

print(np.array_equal(scaler.std_, np.std(X)))

print(np.array_equal(scaler.transform(X),
                     (X - np.mean(X)) / np.std(X)))

print(np.array_equal(scaler.transform([26]),
                     (26 - np.mean(X)) / np.std(X)))
```

Rem:most valuable with `pipeline`

http://scikit-learn.org/stable/modules/pipeline.html

Prediction

We call **prediction** function the function that associates an estimation of the variable of interest to a new sample. For least squares the prediction is given by :

$$\mathrm{pred}(x_{n+1}) = \widehat{\theta}_0 + \widehat{\theta}_1 x_{n+1}$$

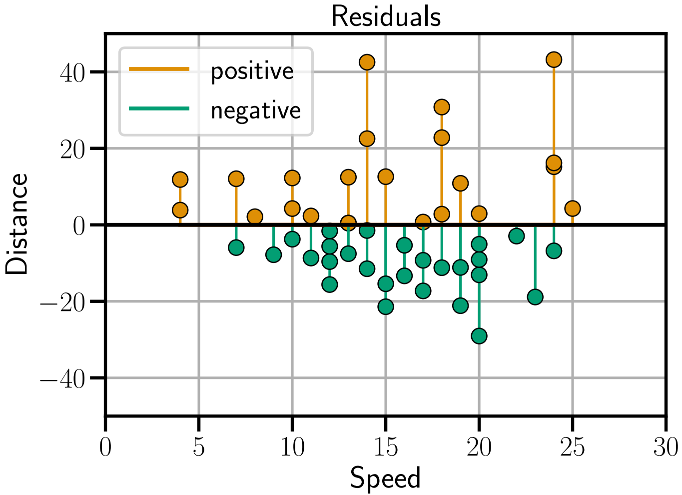
<u>Rem</u>: often written $\widehat{y}_{n+1}$ (implicit dependence on x_{n+1})

The **residual** : difference between observations and predicted values

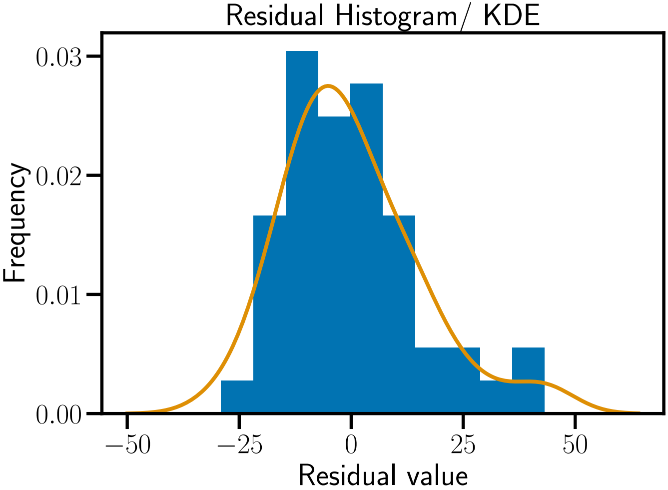
$$\epsilon_i = y_i - \mathrm{pred}(x_i) = y_i - \widehat{y}_i = y_i - (\widehat{\theta}_0 + \widehat{\theta}_1 x_i)$$

<u>Rem</u>: observable estimate of the unobservable statistical error

Residuals (on cars)



Residual histograms



Least squares motivation

- Computing advantage : computationally heavy methods avoided before computers (*e.g.* iterative methods)
- Theoretical advantage : least square analysis easy under simple hypothesis
- Interpretability : how much does the regressor increase with the features

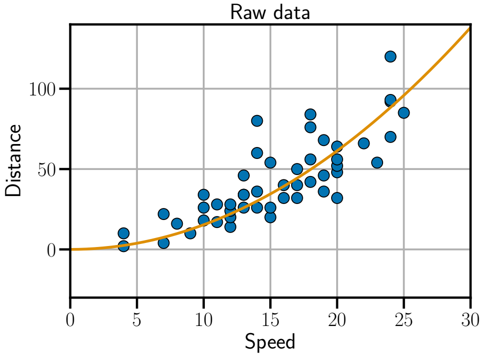
Example : under additive white Gaussian noise assumption *i.e.*, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ the maximum likelihood is equivalent to solving least squares to estimate $(\theta_0^\star, \theta_1^\star)$

Rem: for another noise model and/or to limit outliers influence one can solve (see *e.g.* `QuantReg` in `statsmodels`)

$$\boxed{\widehat{\boldsymbol{\theta}} = (\widehat{\theta}_0, \widehat{\theta}_1) \in \mathrm{argmin}_{(\theta_0,\theta_1)\in\mathbb{R}^2} \sum_{i=1}^{n} |y_i - \theta_0 - \theta_1 x_i|}$$

Discussion : toward multivariate cases

Physical laws (or your driving school memories) would lead to rather pick a **quadratic** model instead of a **linear** one : the OLS can be applied by choosing x_i^2 as features instead of x_i :

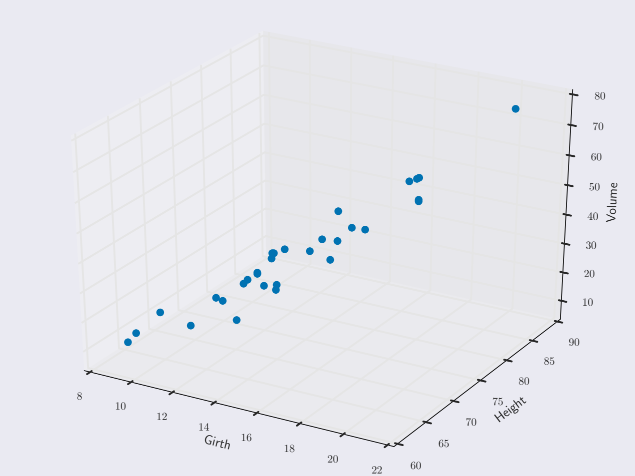


Web sites and books to go further

- Datascience in general : Blog + videos by Jake Vanderplas `http://jakevdp.github.io/`
  Homework for next lesson : watch the following videos `http://jakevdp.github.io/blog/2017/03/03/reproducible-data-analysis-in-jupyter/`
- A few notebooks of OLS with `statsmodels`
- McKinney (2012) about `Python` for statistics
- Lejeune (2010) about linear models (in French)
- Regression course by B. Delyon (in French, more technical)

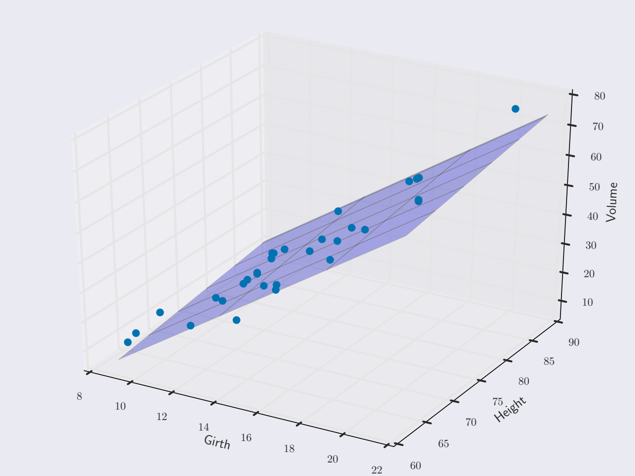
Toward multivariate models

Tree volume as a function of height / girth (🇫🇷 : *circonférence*)



Toward multivariate models

Tree volume as a function of height / girth (🇫🇷 : *circonférence*)



Python commands

```python
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LinearRegression

# Generate example data
...

# Fit linear regression model
model = LinearRegression()
model.fit(X, y)
```

Model

One observes p features $(\mathbf{x}_1, \ldots, \mathbf{x}_p)$. Model in dimension p

$$\begin{aligned}
y_i &= \theta_0^\star + \sum_{j=1}^{p} \theta_j^\star x_{i,j} + \varepsilon_i \\
\varepsilon_i &\overset{i.i.d}{\sim} \varepsilon, \text{ pour } i = 1, \ldots, n \\
\mathbb{E}[\varepsilon] &= 0
\end{aligned}$$

Rem: we assume (frequentist point of view) there exists a “true” parameter $\boldsymbol{\theta}^\star = (\theta_0^\star, \ldots, \theta_p^\star)^\top \in \mathbb{R}^{p+1}$

Dimension p

Matrix model

$$\underbrace{\begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}}_{\mathbf{y}} = \underbrace{\begin{pmatrix} 1 & x_{1,1} & \dots & x_{1,p} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n,1} & \dots & x_{n,p} \end{pmatrix}}_{X} \underbrace{\begin{pmatrix} \theta_0^\star \\ \vdots \\ \theta_p^\star \end{pmatrix}}_{\boldsymbol{\theta}^\star} + \underbrace{\begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_n \end{pmatrix}}_{\boldsymbol{\epsilon}}$$

$$\text{Equivalently :} \quad \boxed{\mathbf{y} = X\boldsymbol{\theta}^\star + \boldsymbol{\epsilon}} \tag{1}$$

Column notation : $X = (\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_p)$ with $\mathbf{x}_0 = \mathbf{1}_n = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix}$.

Line notation : $X = \begin{pmatrix} x_1^\top \\ \vdots \\ x_n^\top \end{pmatrix} = (x_1, \dots, x_n)^\top$

Matrix Notation and L_2 Norm

Matrix notation is a powerful way to represent mathematical operations involving vectors and matrices.

The **Inner Product** (dot product) of two vectors $\mathbf{u}$ and $\mathbf{v}$ is defined as :

$$\langle \mathbf{u}, \mathbf{v} \rangle = \sum_{i=1}^{n} u_i v_i = \mathbf{u} \cdot \mathbf{v}^T$$

Let $\mathbf{A}$ be an $m \times n$ matrix and $\mathbf{B}$ be an $n \times p$ matrix. The **matrix product** $\mathbf{C} = \mathbf{AB}$ is an $m \times p$ matrix with elements :

$$c_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$$

The L_2 **Norm** (Euclidean norm) of a vector $\mathbf{v}$ is defined as :

$$\|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2}$$

Matrix notation simplifies operations and equations involving vectors and matrices.

Vocabulary

$$\boxed{\mathbf{y} = X\boldsymbol{\theta}^\star + \boldsymbol{\epsilon}}$$

- $\mathbf{y} \in \mathbb{R}^n$: observations vector
- $X \in \mathbb{R}^{n\times(p+1)}$: **design** matrix (with features as columns and a first column of 1s)
- $\tilde{X} \in \mathbb{R}^{n\times(p)}$: **reduced design** matrix (with features as columns and NO column of ones)
- $\boldsymbol{\theta}^\star \in \mathbb{R}^{p+1}$: (unknown) **true** parameter to be estimated
- $\boldsymbol{\epsilon} \in \mathbb{R}^n$: noise vector

(Ordinary) Least squares

A least square estimator is **any** solution of the following problem :

$$\widehat{\boldsymbol{\theta}} \in \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^{p+1}} \left(\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 \right)$$

$$\widehat{\boldsymbol{\theta}} \in \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^{p+1}} \sum_{i=1}^{n} \left[y_i - \left(\theta_0 + \sum_{j=1}^{p} \theta_j x_{i,j} \right) \right]^2$$

$$\widehat{\boldsymbol{\theta}} \in \operatorname{argmin}_{\boldsymbol{\theta} \in \mathbb{R}^{p+1}} \sum_{i=1}^{n} \left[y_i - \langle x_i, \boldsymbol{\theta} \rangle \right]^2$$

- Does the solution exist ? A solution always exists, as we are minimizing a coercive continuous function (**coercive** : $\lim_{\|x\| \to +\infty} f(x) = +\infty$)
- Is the solution unique ? not guaranteed

Exo how do we make the prediction ?

Row / column interpretation

Row interpretation

Let $\tilde{x}_1^\top, \ldots, \tilde{x}_{p+1}^\top$ be the rows of X. The residuals are $r_i = \tilde{x}_i \boldsymbol{\theta} - y_i$ and the OLS is equivalent to minimizing the sum of squares residuals

Column interpretation

Let $x_1, \ldots, x_{p+1}$ be the columns of X. Then $\|\mathbf{y} - X\boldsymbol{\theta}\|_2^2 = \|(\theta_0 x_0, \ldots, \theta_p x_p) - \mathbf{y}\|_2^2$, so OLS is to find a linear combination of columns of X that is closest to $\mathbf{y}$.

Vocabulary (and abuse of terms)

We call **Gram matrix** the matrix

$$X^\top X$$

whose general term is $[X^\top X]_{i,j} = \langle \mathbf{x}_i, \mathbf{x}_j \rangle$

If the design matrix X is centered and scaled, the Gram matrix is proportional to the correlation between columns. $X^\top X$ is often referred to as the feature correlation matrix

<u>Rem</u>: when columns are scaled such that $\forall j \in [\![0, p]\!], \|\mathbf{x}_j\|^2 = n$, the Gramian diagonal is $(n, \dots, n)$

The vector $X^\top \mathbf{y} = \begin{pmatrix} \langle \mathbf{x}_0, \mathbf{y} \rangle \\ \vdots \\ \langle \mathbf{x}_p, \mathbf{y} \rangle \end{pmatrix}$ represents the correlation between the observations and the features

Hilbert projection theorem (HPT)

Let $C \subset \mathbb{R}^d, Y \in \mathbb{R}^d$. Let $\widehat{z} = \arg\min_{z \in C} \|Y - z\|_2^2$. Then $\widehat{z}$ always exists and is given by

$$\boxed{< Y - \widehat{z}, z >= 0 \qquad \forall z \in C}$$

Hilbert projection theorem (HPT) and application to OLS

$$\widehat{\boldsymbol{\theta}} \in \text{argmin}_{\theta\in\mathbb{R}^{p+1}} \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$$

Note $col(X) = span([x_0, ..., x_p]) = \sum_{j=0}^{p} x_j\theta_j = X\boldsymbol{\theta}$ OLS :
$\widehat{W} \in \text{argmin}_{W\in col(X)} \left(\|\mathbf{y} - W\|_2^2\right)$

$$\begin{aligned}
< \mathbf{y} - \widehat{W}, W > &= 0 \\
(\mathbf{y} - \widehat{W})^\top W &= 0 \\
(\mathbf{y} - \widehat{W})^\top X\boldsymbol{\theta} &= 0 \\
(\mathbf{y} - \widehat{W})^\top X &= 0 \\
(\mathbf{y} - X\widehat{\boldsymbol{\theta}})^\top X &= 0 \\
X^\top(\mathbf{y} - X\widehat{\boldsymbol{\theta}}) &= 0 \\
X^\top X\widehat{\boldsymbol{\theta}} &= X^\top\mathbf{y}
\end{aligned} \tag{2}$$

OLS normal equations

The solution to the OLS problem is given by the solution to the normal equation

$$\textbf{Normal equation :} \boxed{X^\top X \widehat{\boldsymbol{\theta}} = X^\top \mathbf{y}}$$

As a consequence,

- a solution always exists.
- its unique if the solution to the normal equations is unique

Hilbert projection theorem

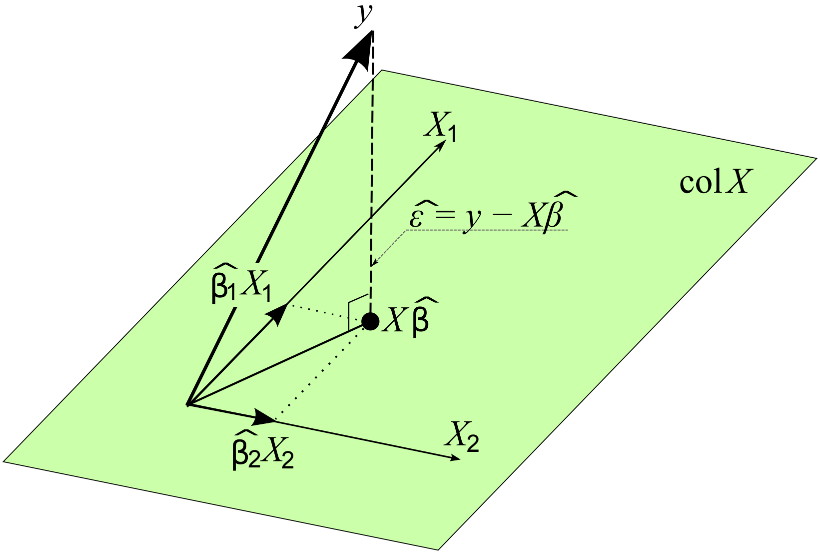


FIGURE – Souce : Wikipedia

Least squares and uniqueness

Let $\widehat{\boldsymbol{\theta}}$ be a solution of $\boxed{X^\top X\widehat{\boldsymbol{\theta}} = X^\top \mathbf{y}}$

Non uniqueness : happens for non trivial kernel,*i.e.* when $\ker(X) = \{\boldsymbol{\theta} \in \mathbb{R}^{p+1} : X\boldsymbol{\theta} = 0\} \neq \{0\}$

Assume $\boldsymbol{\theta}_K \in \ker(X)$ with $\boldsymbol{\theta}_K \neq 0$, then

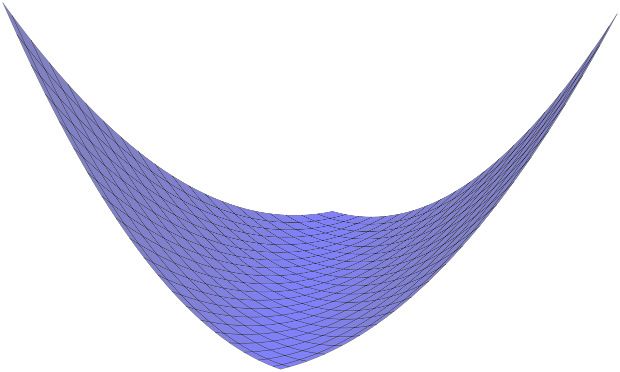
$$\begin{aligned} X(\widehat{\boldsymbol{\theta}} + \boldsymbol{\theta}_K) &= X\widehat{\boldsymbol{\theta}} \\ \text{and then} \quad (X^\top X)(\widehat{\boldsymbol{\theta}} + \boldsymbol{\theta}_K) &= X^\top \mathbf{y} \end{aligned}$$

<u>Conclusion</u> : the set of least squares solutions is an affine sub-space

$$\boxed{\widehat{\boldsymbol{\theta}} + \ker(X)}$$

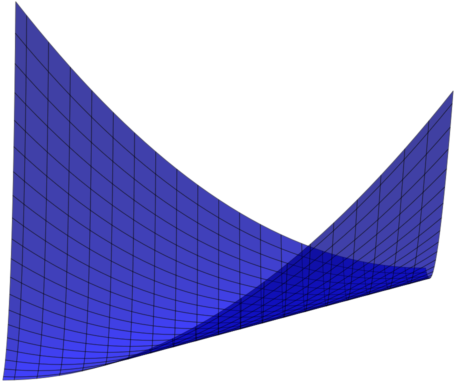
Optimization in $\mathbb{R}^d$

Convex case, $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$, where the set of minimizers is non-unique :



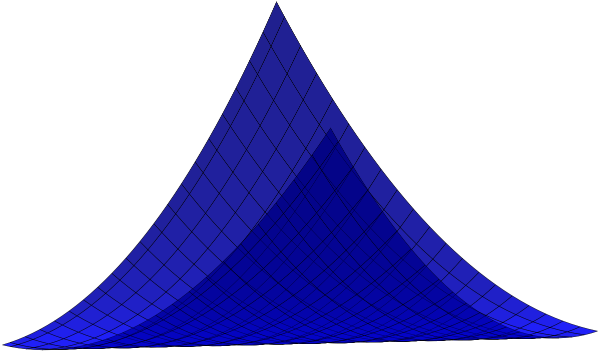
Optimization in $\mathbb{R}^d$

Convex case, $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$, where the set of minimizers is non-unique :



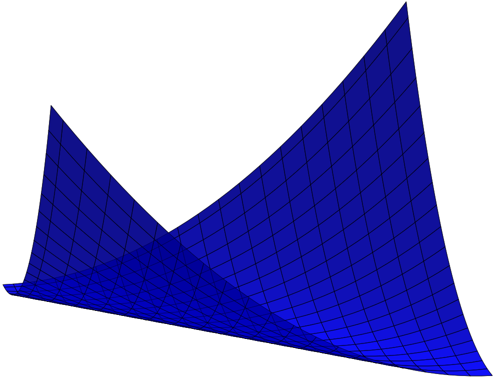
Optimization in $\mathbb{R}^d$

Convex case, $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$, where the set of minimizers is non-unique :



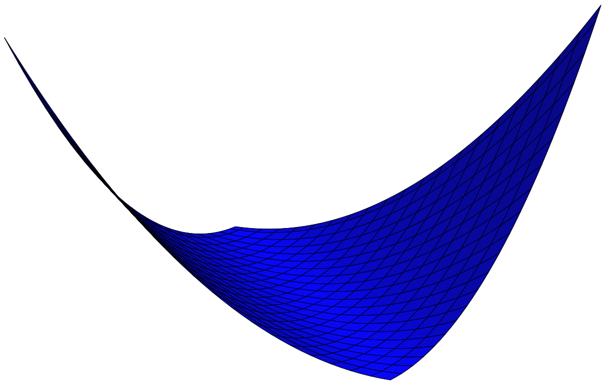
Optimization in $\mathbb{R}^d$

Convex case, $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$, where the set of minimizers is non-unique :



Optimization in $\mathbb{R}^d$

Convex case, $f(\boldsymbol{\theta}) = \|\mathbf{y} - X\boldsymbol{\theta}\|_2^2$, where the set of minimizers is non-unique :



Non uniqueness : single feature case

Reminder : $$X = \begin{pmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}$$

If $\ker(X) = \{\boldsymbol{\theta} \in \mathbb{R}^2 : X\boldsymbol{\theta} = 0\} \neq \{0\}$ there exists $(\theta_0, \theta_1) \neq (0, 0)$:

$$\begin{cases} \theta_0 + \theta_1 x_1 & = 0 \\ \vdots \qquad \vdots & = \vdots \\ \theta_0 + \theta_1 x_n & = 0 \end{cases} \qquad (\star)$$

1. If $\theta_1 = 0$: $(\star) \Rightarrow \theta_0 = 0$, so $(\theta_0, \theta_1) = (0, 0)$, **contradiction**
2. If $\theta_1 \neq 0$:
   2.1 If $\forall i, x_i = 0$ then $X = (\mathbf{1}_n, 0)$ and $\theta_0 = 0$
   2.2 Otherwise there exists $x_{i_0} \neq 0$ and $\forall i, x_i = -\theta_0/\theta_1 = x_{i_0}$, *i.e.* $X = [\mathbf{1}_n \quad x_{i_0} \cdot \mathbf{1}_n]$

Interpretation : $\mathbf{x}_1 \propto \mathbf{1}_n$, *i.e.* $\mathbf{x}_1$ is constant

Interpretation for multivariate cases

Reminder : we write $X = (\mathbf{1}_n, \mathbf{x}_1, \ldots, \mathbf{x}_p)$, the features being column-wise (each are of length n)

The property $\ker(X) = \{\boldsymbol{\theta} \in \mathbb{R}^{p+1} : X\boldsymbol{\theta} = 0\} \neq \{0\}$ means that there exists a linear dependence between the features $\mathbf{1}_n, \mathbf{x}_1, \ldots, \mathbf{x}_p$,

Reformulation : $\exists \boldsymbol{\theta} = (\theta_0, \ldots, \theta_p)^\top \in \mathbb{R}^{p+1} \setminus \{0\}$ s.t.

$$\theta_0 \mathbf{1}_n + \sum_{j=1}^{p} \theta_j \mathbf{x}_j = 0$$

Algebra reminder

Rank of a matrix : $\quad \mathrm{rank}(X) = \dim(\mathrm{span}(\mathbf{1}_n, \mathbf{x}_1, \ldots, \mathbf{x}_p))$; $\mathrm{span}(\cdot)$: the space generated by $\cdot$

Property : $\mathrm{rank}(X) = \mathrm{rank}(X^\top)$

Rank–nullity theorem :

- $\mathrm{rank}(X) + \dim(\ker(X)) = p + 1$
- $\mathrm{rank}(X^\top) + \dim(\ker(X^\top)) = n$

Property : $\boxed{\mathrm{rank}(X) \leq \min(n, p+1)}$

See Golub and Van Loan (1996) for details

Algebra reminder (continued)

Matrix inversion : A square matrix $A \in \mathbb{R}^{m \times m}$ is invertible

- if and only if its kernel is trivial : $\ker(A) = \{0\}$
- if and only if it is full rank $\text{rank}(A) = m$

OLS is unique iff $X^\top X$ is invertible

$\Leftrightarrow \ker(X^\top X) = \{0\}$

$\Leftrightarrow \ker(X) = \{0\}$

$\Leftrightarrow X$ has full rank

Exo: $\ker(X) = \ker(X^\top X)$

Prediction

$$\textbf{Prediction vector :} \qquad \widehat{\mathbf{y}} = X\widehat{\boldsymbol{\theta}}$$

Rem: $\widehat{\mathbf{y}}$ depends linearly on the observation vector $\mathbf{y}$

Rem: an **orthogonal projector** is a matrix H such that

1. H is symmetric : $H^\top = H$
2. H is idempotent : $H^2 = H$

Proposition Writing H_X the orthogonal projector onto the space span by the columns of X, one gets $\widehat{\mathbf{y}} = H_X\mathbf{y}$

If X is full (column) rank, then $H_X = X(X^\top X)^{-1}X^\top$ is called the **hat matrix**

Exo: Show that H_X is an orthogonal projector

Prediction (continued)

If a new observation $x_{n+1} = (x_{n+1,1}, \ldots, x_{n+1,p})$ is provided, the associated prediction is :

$$\widehat{y}_{n+1} = \langle \widehat{\boldsymbol{\theta}}, (1, x_{n+1,1}, \ldots, x_{n+1,p})^\top \rangle$$
$$\widehat{y}_{n+1} = \widehat{\theta}_0 + \sum_{j=1}^{p} \widehat{\theta}_j x_{n+1,j}$$

<u>Rem</u>: the normal equation ensures **equi-correlation** between observations and features :

$$\begin{aligned}(X^\top X)\widehat{\boldsymbol{\theta}} = X^\top \mathbf{y} \Leftrightarrow X^\top \widehat{\mathbf{y}} &= X^\top \mathbf{y} \\ \Leftrightarrow \begin{pmatrix} \langle \mathbf{x}_0, \widehat{\mathbf{y}} \rangle \\ \vdots \\ \langle \mathbf{x}_p, \widehat{\mathbf{y}} \rangle \end{pmatrix} &= \begin{pmatrix} \langle \mathbf{x}_0, \mathbf{y} \rangle \\ \vdots \\ \langle \mathbf{x}_p, \mathbf{y} \rangle \end{pmatrix}\end{aligned}$$

Residuals and normal equation

$$\textbf{Residual(s)}: \quad \widehat{\boldsymbol{\varepsilon}} = \mathbf{y} - \widehat{\mathbf{y}} = \mathbf{y} - X\widehat{\boldsymbol{\theta}} = (\mathrm{Id}_n - H_X)\mathbf{y}$$

Proposition

$$\begin{aligned} < \widehat{\boldsymbol{\varepsilon}}, X >&= 0_n \\ < \widehat{\boldsymbol{\varepsilon}}, \widehat{\mathbf{y}} >&= 0 \\ < \widehat{\boldsymbol{\varepsilon}}, \bar{\mathbf{y}}\mathbf{1}_n >&= 0 \end{aligned} \tag{3}$$

<u>Rem</u>: The Normal equation is $(X^\top X)\widehat{\boldsymbol{\theta}} = X^\top \mathbf{y}$. It follows that $X^\top(X\widehat{\boldsymbol{\theta}} - \mathbf{y}) = 0 \Leftrightarrow X^\top \widehat{\boldsymbol{\varepsilon}} = 0 \Leftrightarrow \widehat{\boldsymbol{\varepsilon}}^\top X = 0$

With $X = (\mathbf{1}_n, \mathbf{x}_1, \ldots, \mathbf{x}_p)$, this can be rewritten

$$\forall j = 1, \ldots, p : \langle \widehat{\boldsymbol{\varepsilon}}, \mathbf{x}_j \rangle = 0 \text{ and } \overline{r}_n = 0$$

<u>Interpretation</u> : (1,2) residuals are $\perp$ to features and (3) $\widehat{\boldsymbol{\varepsilon}}$ is centered ($\sum \widehat{\epsilon}_i = 0$)

How good is our model ? RSS and the determination coefficient R^2

The ratio of the variation explained by the model and the total variation of the data $R^2 = \frac{\|\widehat{\mathbf{y}}-\bar{\mathbf{y}}\mathbf{1}_n\|^2}{\|\mathbf{y}-\bar{\mathbf{y}}\mathbf{1}_n\|^2}$ We can write also, by orthogonality :

$$\|\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}_n\|^2 = \|\mathbf{y} - \widehat{\mathbf{y}}\|^2 + \|\widehat{\mathbf{y}} - \bar{\mathbf{y}}\mathbf{1}_n\|^2 \quad (4)$$

Reordering

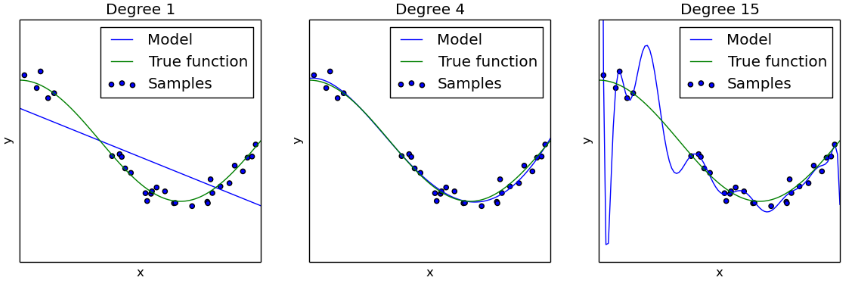
$$\|\widehat{\mathbf{y}} - \bar{\mathbf{y}}\mathbf{1}_n\|^2 = \|\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}_n\|^2 - \|\mathbf{y} - \widehat{\mathbf{y}}\|^2 \quad (5)$$

So

$$R^2 = 1 - \frac{\|\mathbf{y} - \widehat{\mathbf{y}}\|^2}{\|\mathbf{y} - \bar{\mathbf{y}}\mathbf{1}_n\|^2} \quad (6)$$

Exo: Show that $0 \leq R^2 \leq 1$

Polynomial regression and overfitting



Source : sklearn

References I

B. Delyon.
Régression, 2015.
https://perso.univ-rennes1.fr/bernard.delyon/regression.pdf.

G. H. Golub and C. F. van Loan.
Matrix computations.
Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

M. Lejeune.
Statistiques, la théorie et ses applications.
Springer, 2010.

W. McKinney.
Python for Data Analysis : Data Wrangling with Pandas, NumPy, and IPython.
O'Reilly Media, 2012.