X/HEC Datascience for Business
2025/2026 **Mathurin Massias**

Lab : Introduction: Python, Numpy, Pandas

- Evaluation -

The Lab is done by pairs. **Each student in the pair** must upload his/her notebook, with the name constructed as `fistname1_lastname1_firstname2_lastname2.ipynb`, where you have substituted your first and last names. **There is no evaluation if you don't respect this.** If code is shared between groups, both groups get 0. Read the intro slides on what's expected of you in the labs.

Exercice 1. (Time series analysis with pandas)

Let us use the dataset[1] **Individual household electric power consumption Data Set**.

First, execute the following commands to download the data (this snippet is also in `download_hpc.py` on Moodle if copy pasting fails):

```python
import sys
import urllib
import zipfile
from os import path

import pandas as pd

url = u'https://archive.ics.uci.edu/ml/machine-learning-databases/00235/'
filename = 'household_power_consumption'
zipfilename = filename + '.zip'
location = url + zipfilename

if not path.isfile(zipfilename): urllib.request.urlretrieve(location, zipfilename)

zipfile.ZipFile(zipfilename).extractall()

na_values = ['?', '']
fields = ['Date', 'Time', 'Global_active_power']
df = pd.read_csv(filename + '.txt', sep=';', nrows=200000,
                na_values=na_values, usecols=fields)
```

We only focus on the `Global_active_power` feature for the moment.

1) Count the number of lines with missing values. Erase all such lines.

2) Use the `pandas` functions `to_datetime` and `set_index` to set a DateTimeIndex to `df`. You should first preserve the full date information, i.e. keep the hour, minute, seconds information in your newly created DateTime. **Beware**, when reading dates, that the international dates format that is different from the French standard.
What's the advantage of having a DateTimeIndex compared to a regular index?

3) Display the graphic of daily averages, between January 1 2007 and April 30 2007. Propose an explanation for the consumption behavior between February and early April.

Let us now add some temperature information for our study. Such information can be found on Moodle in "`TG_STAID011249.txt`" [2]. Here the temperatures available are the one in Orly.

[1] https://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption

[2] or online at http://eca.knmi.nl/dailydata/predefinedseries.php

4) Load the dataset with `pandas`, and keep only the `DATE` and `TG` columns. Divide by 10 the `TG` column to get Celsius temperature. Treat missing values as NaNs.

5) Create a `pandas` Time Series with the daily temperatures between January 1 2007 and April 30 2007. Display on the same graph the temperature and the `Global_active_power` Time Series. Using a twinx axis might help to display 2 series of values with different magnitudes in a readable fashion.

Exercice 2. (Linear algebra)

We remind the following linear algebra fact: for any $X \in \mathbb{R}^{n\times p}$ and $y \in \mathbb{R}^n$ the following equation holds true:

$$X^\top (XX^\top + \lambda \mathrm{Id}_n)^{-1} y = (X^\top X + \lambda \mathrm{Id}_p)^{-1} X^\top y \tag{1}$$

6) Check this property numerically for $\lambda = 10^{-5}$ without inverting any matrix. To compute $A^{-1}b$, never invert directly A: solve the linear system $Ax = b$ with `np.linalg.solve`[3], for a matrix X whose entries are generated randomly (iid) according to a Gaussian distribution with mean zero and variance 5, and for a vector y with coordinates generated randomly (iid) according to a uniform distribution over $[-1, 1]$,

   (a) check it for $n = 100$ and $p = 2000$,

   (b) check it for $n = 2000$ and $p = 100$.

7) For a few scenarios similar to (a) and (b) ($n \ll p, p \ll n$), do a short numerical/graphical study to compare (according to n and p) when it is more time efficient to compute the quantity in (1) using the left hand side formulation or right hand side formulation. Explain the results.

Exercice 3. (Random matrix spectrum)

8) Choose three non-Gaussian probability distributions, with mean 0 and variance 2, and write a function that takes as input n, p and the distribution name, and creates a matrix $X \in \mathbb{R}^{n\times p}$ with entries generated (iid) according to this distribution.

9) Display on one single graph the singular values of X for $n = 1000$, and $p = 200, 500, 1000, 2000$ for the three distributions chosen.

10) Display on one single graph the spectrum (i.e. the set of eigen values) of $X^\top X/n$ for $n = 1000$, and $p = 200, 500, 1000, 2000$. Comment.

Exercice 4. (Power method)

We consider a matrix $X \in \mathbb{R}^{n\times p}$ generated as in Exercise 2, question 1), for $(n, p) = (200, 400)$.

11) Write a function coding Algorithm 1.

```
Algorithme 1 : Power method
  Input    : Matrix $X$; maximal number of iterations: $T$
  Choose $v_0 \in \mathbb{R}^p$ at random
  for $k = 1, \dots, T$ do
      $u_k = Xv_{k-1}/\|Xv_{k-1}\|$
      $v_k \leftarrow X^\top u_k/\|X^\top u_k\|$
  Output : $u_T, v_T$
```

12) Modify the implementation of the algorithm to store all iterates of u and v. Let u^* (resp. v^*) be the leading left (resp. right) singular vector of X. Compute them using `np.linalg.svd`. Plot the norm of $u_k - u^*$ as a function of k. Is it true that the output u, v from the algorithm converge to u^*, v^*? Run your code several times. *Bonus*: can you show it mathematically?

[3] this is mathematically equivalent, but more stable numerically: see https://stackoverflow.com/questions/31256252/why-does-numpy-linalg-solve-offer-more-precise-matrix-inversions-than-numpy-li

13) Provide two initialization vectors v_0 leading to different limits for this algorithm; explain how they are related.

14) Provide a way to approximate the largest singular value of X using the power method.

15) Build upon the power method to provide an algorithm that can approximate the second largest singular value of X (without using an SVD function).

Exercice 5. (Analysis of the `auto-mpg` dataset)

Here, we consider the `auto-mpg.data`. We aim at predicting cars consumption based on several characteristics: cylinders, displacement, horsepower, weight, acceleration, year, country and cars name. The output coding cars consumption (more precisely the "mpg", i.e. the distance ridden in miles for a gallon of oil) is written y;

16) Import the dataset from https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.data-original with Pandas. Add columns name using the `name` parameter of `read_csv` and consulting: https://archive.ics.uci.edu/ml/machine-learning-databases/auto-mpg/auto-mpg.names. You can check the impact of using `sep=r"\s+"`. Is there a marker for missing values in this dataset? If needed, remove the corresponding lines. The last column, `car name`, is not useful for our study: drop it.

17) Add two or three binary features to meaningfully encode the three origins (`'origin'` feature, for which, initially, 1 stands for USA, 2 for Europe and 3 for Japan)[4].

18) Select (manually) 9 rows of the dataset such that all 3 origins are represented, and model year is not constant. Get the least-squares estimator $\hat{\theta}$ (with intercept) and the prediction vector $\hat{y}$, considering only these 9 lines. What do you observe? Why?

19) Now, get the least-squares estimator $\hat{\theta}$ and the prediction vector $\hat{y}$ (with intercept) over the whole dataset, after performing scaling/centering (the columns must have unit standard deviation and zero mean). Which variables seem to best explain gasoline consumption *according to your model*? Why wouldn't this answer make sense if the columns were not normalized?

20) Assume you observe a new car with the following values features:

cylinders	displacement	horsepower	weight	acceleration	year	origin
6	225	100	3233	15.4	2017	1

Can you predict its consumption in this model? Beware of the year encoding.

Use a pipeline http://scikit-learn.org/stable/modules/generated/sklearn.pipeline.Pipeline.html for performing the rescaling and the least-squares step consecutively. Comment on the predicted y value for this car.

[4] cf http://lib.stat.cmu.edu/datasets/cars.desc