TP n° 1 : Introduction: Python, Numpy, Pandas, *et al.*

Some remarks prior starting

We are going to use `jupyter notebook` for all practical sessions. For those using Telecom's machines, choose the Anaconda versions (the other version is outdated).

Some important points to remember :

- Loading -

```
import math                          # import a package
import numpy as np                   # import a package with a nickname
from sklearn import linear_model     # import whole module
from os import mkdir                 # import a function
```

- Using standard help -

```
help(mkdir)                          # to get some help on mkdir
linear_model.LinearRegression?       # to get some help on LinearRegression
```

- Package versions, function localization -

```
print(np.__version__)                # to get a package version
from inspect import getsourcelines   # to get a function source code
getsourcelines(linear_model.LinearRegression)
```

Introduction to Python, Numpy and Scipy

This section can be skipped for those already familiar with Python, Numpy and Scipy. For the others, it could be useful to browse the following documents :

http://perso.telecom-paristech.fr/~gramfort/liesse_python/1-Intro-Python.html
http://perso.telecom-paristech.fr/~gramfort/liesse_python/2-Numpy.html
http://perso.telecom-paristech.fr/~gramfort/liesse_python/3-Scipy.html

The associated questions are below :

1) Write a function `nextpower` that outputs the next power of 2 for a given (float or int) number. Check that your output is of type `int`, test it for instance with `type`.
2) From a word containing all the alphabet letters, generate with a string *slicing* the string `cfilorux`. Propose two versions. Do the same for the string `vxz`.
3) Display the number π with 9 digits.
4) Count the number of occurrences of each character in the string `s="HelLo WorLd!!"`. Output a dictionary that for each character associate the number of occurrences.
5) Write a function performing the Cesar code : each character is replaced by another one (and only one). You can use the `shuffle` function on the string containing the whole alphabet for instance.

6) Compute $2\prod_{k=1}^{\infty}\frac{4k^2}{4k^2-1}$ efficiently. Use for instance `time` to determine a fast version. Propose a version without loop, using `Numpy`

7) Create a function `quicksort` based on the following pseudo-code :

```
function quicksort('array')
 if length('array') <= 1
     return 'array'
 select and remove a pivot value 'pivot' from 'array'
 create empty lists 'less' and 'greater'
 for each 'x' in 'array'
    if 'x' <= 'pivot' then append 'x' to 'less'
    else append 'x' to 'greater'
 return concatenate(quicksort('less'), 'pivot', quicksort('greater'))
```

Hint : the length of a list is given by `len(l)` and two lists can be concatenated with `l1 + l2` and `l.pop()` remove the last element from the list l.

8) Without using `for` / `while` loops, create a random matrix $M \in \mathbb{R}^{5\times 6}$ with coefficients taken uniformly (and independently) in $[-1,1]$, then replace every other column by its value minus twice the value of the next column. Replace the negative values by 0 using a binary mask.

9) Create a random matrix $M \in 5\times 20$ with coefficients taken uniformly (and independently) in $[-1,1]$. Test whether $G = M^\top M$ is symmetric (semi-) definite positive, and that its eigenvalues are positive. What is the rank of G ?

Hing : one could use `np.allclose`, `np.logical_not`, `np.all` for instance.

Introduction to Pandas, Matplotlib, etc.

One can start browsing the following tutorial on `pandas` :
http://pandas.pydata.org/pandas-docs/stable/tutorials.html

- Data loading -

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

10) First, execute the following commands :

```
from os path
import pandas as pd
import urllib
import zipfile

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.urlretrieve(Location, 'zipfilename')
zip = zipfile.ZipFile(zipfilename)
zip.extractall()

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

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

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

11) Detect and count the number of line with missing values.
12) Erase all such lines.
13) Use `to_datetime` and `set_index` to create a Time Series (beware of the international dates format that is different from the French standard).
14) 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. On top of `matplotlib` you could use the `seaborn` package for nicer display.

Let us now add some temperature information for our study. Such information can be found on EOLE in "`TG_STAID011249.txt`"[2]. Here the temperatures available are the one in Orly (note that the place were the consumption was collected is unknown in the previous dataset).

15) 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.
16) 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.

Next...

— Consider the dataset in `airparif_abae1bd78def4fe8a409ab8c95fc4608.zip` available on EOLE and propose a visualization of the pollution (per year). Provide a monthly analysis, and find which month is the more polluted.

— http://blog.yhat.com/posts/aggregating-and-plotting-time-series-in-python.html

— http://www.math.univ-toulouse.fr/~besse/Wikistat/pdf/st-tutor2-python-pandas.pdf

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