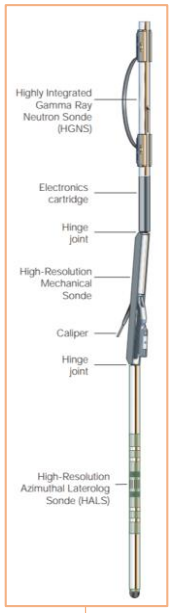
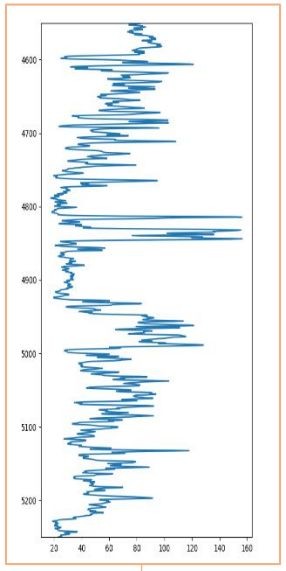
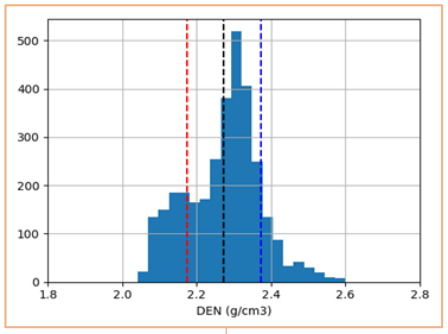
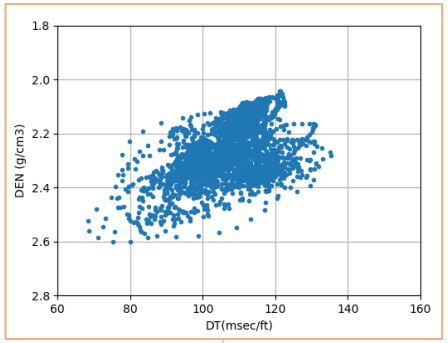



```
~VERSION INFORMATION
VERS.          2.0   :CWLS Log ASCII Standard - VERSION 2.0
WRAP.           NO   :One Line per depth step
PROD.  Schlumberger  :LAS Producer
PROG.  DLIS to ASCII 2.1                          :LAS Program name and version
CREA.                                             :LAS Creation date {YYYY/MM/DD hh:mm}
SOURCE.     ait_DSI.dlis                          :DLIS File Name
FILE-ID.     HAIT .018                            :File Identific
#---------------------------------------------
~WELL INFORMATION
#MNEM.UNIT      DATA            DESCRIPTION
#---- ------ --------------   ----------------------------
STRT .F      7672.0           :START DEPTH
STOP .F      13310.0          :STOP DEPTH
STEP .F       0.5             :STEP
NULL .         -999.25        :NULL VALUE
COMP .                                            :COMPANY
CNTY .                                            :COUNTY
STAT .                                            :STATE
CTRY .                                            :COUNTRY
UWI  .                                            :UNIQUE WELL ID
DATE .       24-FEB-2000                          :LOG DATE {DD-M
SRVC .       Schlumberger                         :SERVICE COMPAN
LATI .DEG                                         :LATITUDE
LONG .DEG                                         :LONGITUDE
GDAT .                                            :GeoDetic Datum
#---------------------------------------------------------------
~PARAMETER INFORMATION
#MNEM.UNIT    VALUE                  DESCRIPTION
#---- -----  -------------------     ------------------------
TD    .F      13320.00000            :Total Depth
TNDTYP.        CMTD-B/A              :Tension Device
TREF .LBF     1000.00000             :Reference Tension of the Cab
DIFF .F          1.00000             :Maximum Permitted Depth Diff
GGRD .DF/F       0.01000             :Geothermal Gradient
GDEV .DEG        0.00000             :Average Angular Deviation of
```

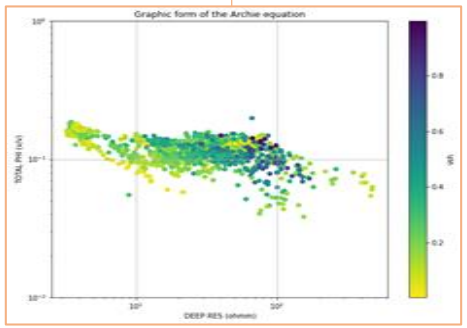
```
import matplotlib.pyplot as plt
import numpy as np

# Basic statistics parameters of logs

# Definition of the function

def basic_stat(l, label, fl):

    log_range= np.max(l) - np.min(l)

    file=open(label+'_stat.txt',"w")
    file.writelines("Statistics of "+label+" \n")
    file.writelines(" \n")
    file.writelines("Minimum value          :"+str(np.min(l))+" \n")
    file.writelines("Maximum value          :"+str(np.max(l))+" \n")
    file.writelines("Range                  :"+str(log_range)+" \n")
    file.writelines("Mean                   :"+str(np.mean(l))+" \n")
    file.writelines("Variance               :"+str(np.var(l))+" \n")
    file.writelines("Standard deviation     :"+str(np.std(l))+" \n")
    file.writelines(" \n")

    filter_log_range= np.max(fl) - np.min(fl)

    file.writelines("Statistics of filter "+label+" \n")
    file.writelines(" \n")
    file.writelines("Minimum value          :"+str(np.min(fl))+" \n")
    file.writelines("Maximum value          :"+str(np.max(fl))+" \n")
    file.writelines("Range                  :"+str(filter_log_range)+" \n")
    file.writelines("Mean                   :"+str(np.mean(fl))+" \n")
    file.writelines("Variance               :"+str(np.var(fl))+" \n")
    file.writelines("Standard deviation     :"+str(np.std(fl))+" \n")
```

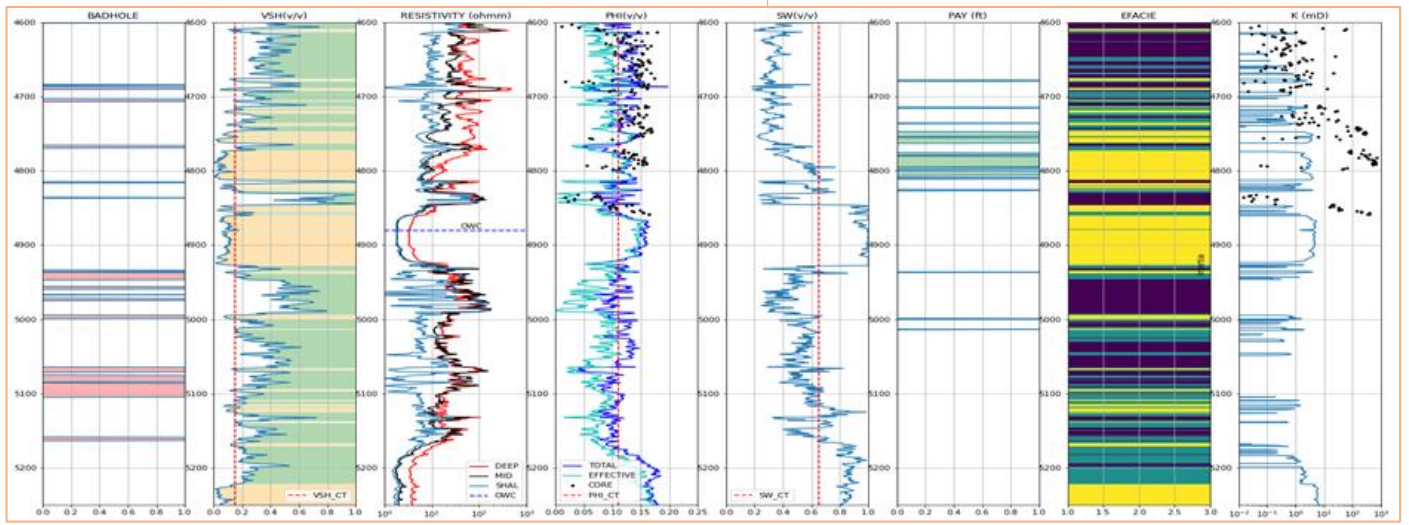






Basic Python for Log Analysis






```
SUMMATION RESULTS

        interval:  5220.0
             res:  486.5
       sand_ntg:  0.0931992337164751
             pay:  314.5
```

Course

Creator and speaker: Manuel David Soto. Geological Eng, UCV (1997). MSc in Geology, University of Texas at Austin (2007). 20 years of experience in operations, exploration and petrophysics. Now in the Petrophysical Specialist team, Repsol, Madrid.

Collaborator: Ulises Berman. Student Geophysical Eng, USB, Caracas, Venezuela.

Organizer: AAPG-USB Student Chapter, Caracas, Venezuela.

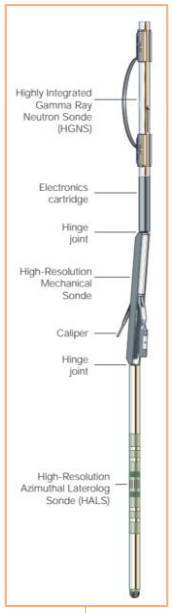
Content of the course

Introduction: Coding. Python and its packages. Installing Python, Jupyter and main libraries.

Session 1: Variables and data types (numeric, Boolean, dictionary, sequences). Arrays. Functions.

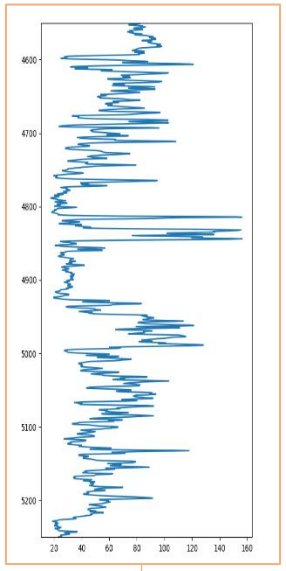
Session 2: Flow control. Reading & writing text and image files. Plots & multi plots. Univariate and bivariate statistical analysis.

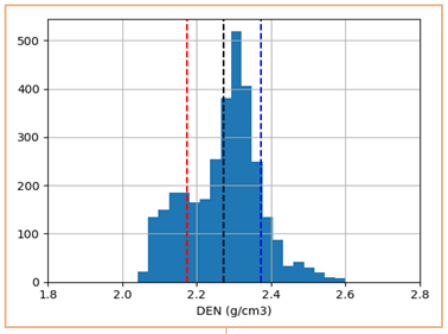
Session 3: Reading and displaying input logs. Parameter selection (function and mask).
Calculation (formulas), summation & displaying output logs. Typing and regression.

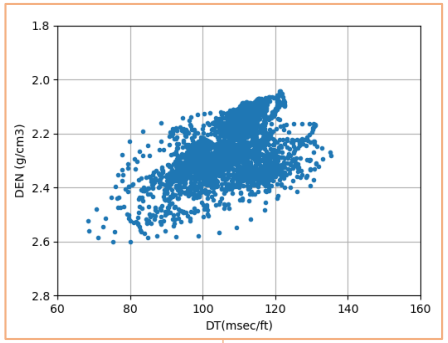



```
~VERSION INFORMATION
VERS.          2.0   :CWLS Log ASCII Standard - VERSION 2.0
WRAP.           NO   :One Line per depth step
PROD.  Schlumberger  :LAS Producer
PROG.  DLIS to ASCII 2.1                          :LAS Program name and version
CREA.                                             :LAS Creation date {YYYY/MM/DD hh:mm}
SOURCE.     ait_DSI.dlis                          :DLIS File Name
FILE-ID.     HAIT .018                            :File Identific
#---------------------------------------------
~WELL INFORMATION
#MNEM.UNIT      DATA            DESCRIPTION
#---- ------ --------------   ----------------------------
STRT .F      7672.0           :START DEPTH
STOP .F      13310.0          :STOP DEPTH
STEP .F       0.5             :STEP
NULL .         -999.25        :NULL VALUE
COMP .                                            :COMPANY
CNTY .                                            :COUNTY
STAT .                                            :STATE
CTRY .                                            :COUNTRY
UWI  .                                            :UNIQUE WELL ID
DATE .       24-FEB-2000                          :LOG DATE {DD-M
SRVC .       Schlumberger                         :SERVICE COMPAN
LATI .DEG                                         :LATITUDE
LONG .DEG                                         :LONGITUDE
GDAT .                                            :GeoDetic Datum
#---------------------------------------------------------------
~PARAMETER INFORMATION
#MNEM.UNIT    VALUE                  DESCRIPTION
#---- -----  -------------------     ------------------------
TD    .F      13320.00000            :Total Depth
TNDTYP.        CMTD-B/A              :Tension Device
TREF .LBF     1000.00000             :Reference Tension of the Cab
DIFF .F          1.00000             :Maximum Permitted Depth Diff
GGRD .DF/F       0.01000             :Geothermal Gradient
GDEV .DEG        0.00000             :Average Angular Deviation of
```

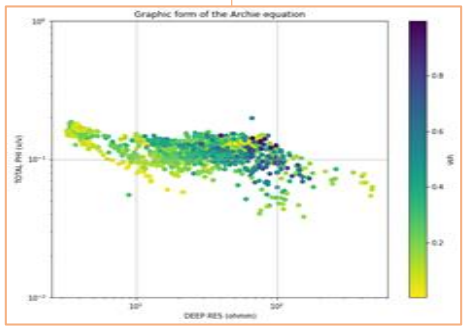
```
import matplotlib.pyplot as plt
import numpy as np

# Basic statistics parameters of logs

# Definition of the function

def basic_stat(l, label, fl):

    log_range= np.max(l) - np.min(l)

    file=open(label+'_stat.txt',"w")
    file.writelines("Statistics of "+label+" \n")
    file.writelines(" \n")
    file.writelines("Minimum value          :"+str(np.min(l))+" \n")
    file.writelines("Maximum value          :"+str(np.max(l))+" \n")
    file.writelines("Range                  :"+str(log_range)+" \n")
    file.writelines("Mean                   :"+str(np.mean(l))+" \n")
    file.writelines("Variance               :"+str(np.var(l))+" \n")
    file.writelines("Standard deviation     :"+str(np.std(l))+" \n")
    file.writelines(" \n")

    filter_log_range= np.max(fl) - np.min(fl)

    file.writelines("Statistics of filter "+label+" \n")
    file.writelines(" \n")
    file.writelines("Minimum value          :"+str(np.min(fl))+" \n")
    file.writelines("Maximum value          :"+str(np.max(fl))+" \n")
    file.writelines("Range                  :"+str(filter_log_range)+" \n")
    file.writelines("Mean                   :"+str(np.mean(fl))+" \n")
    file.writelines("Variance               :"+str(np.var(fl))+" \n")
    file.writelines("Standard deviation     :"+str(np.std(fl))+" \n")
```

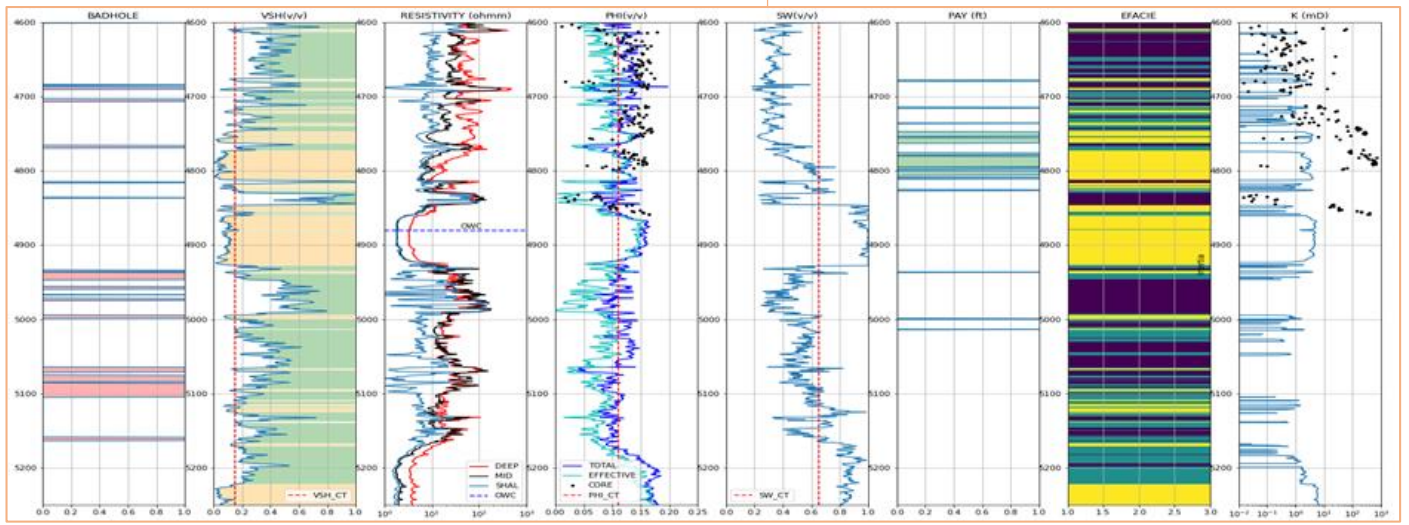






Introduction






```
SUMMATION RESULTS

        interval:  5220.0
             res:  486.5
       sand_ntg:  0.0931992337164751
             pay:  314.5
```

Coding and Python

Int 1. Why learn to code?

What is coding?

"Coding, also called computer programming, is the way to communicate with computers. Code tells a computer what actions to take, and writing code is like creating a set of instructions. By learning to write code, you can tell computers what to do or how to behave in a much faster way."

From: https://grasshopper.app

Int 1. Why learn to code?

Most of our jobs are focus on acquire and process data, and produce reports

Today data analysis is a fundamental part of all technical and not technical jobs.

Big data analysis, according to the World Economic Forum, will be in 10 years one of the world's most in demand professions.

People with basic data analysis skills and knowledge on basic coding and data manipulation will have a big advantage in the coming years.

In geosciences, there is no need to be an expert in the language, just the knowledge of the basic tools will allow you to: read data, produce graphs, statistics and write small reports.

Int 1. Why learn to code?

Advantages of learning basic coding

- Opens new job opportunities
- Is a fundamental skill to analyze data
- Experience in coding makes your job applications stand out
- Coding literacy will strengthen your understanding of the wider aspects of technology
- Coding can boost problem solving and logic skills
- Anyone can do it

Int 1. Why learn to code?

Coding is not difficult

https://iseprostorglive.blob.core.windows.net/user-assets/projects/AIhDsT0NGhVzHl2q9duLx3hVBPFfe6aW/assets/mc-cb5aedfdad8b2301cdaf3efd34de50de.mp4

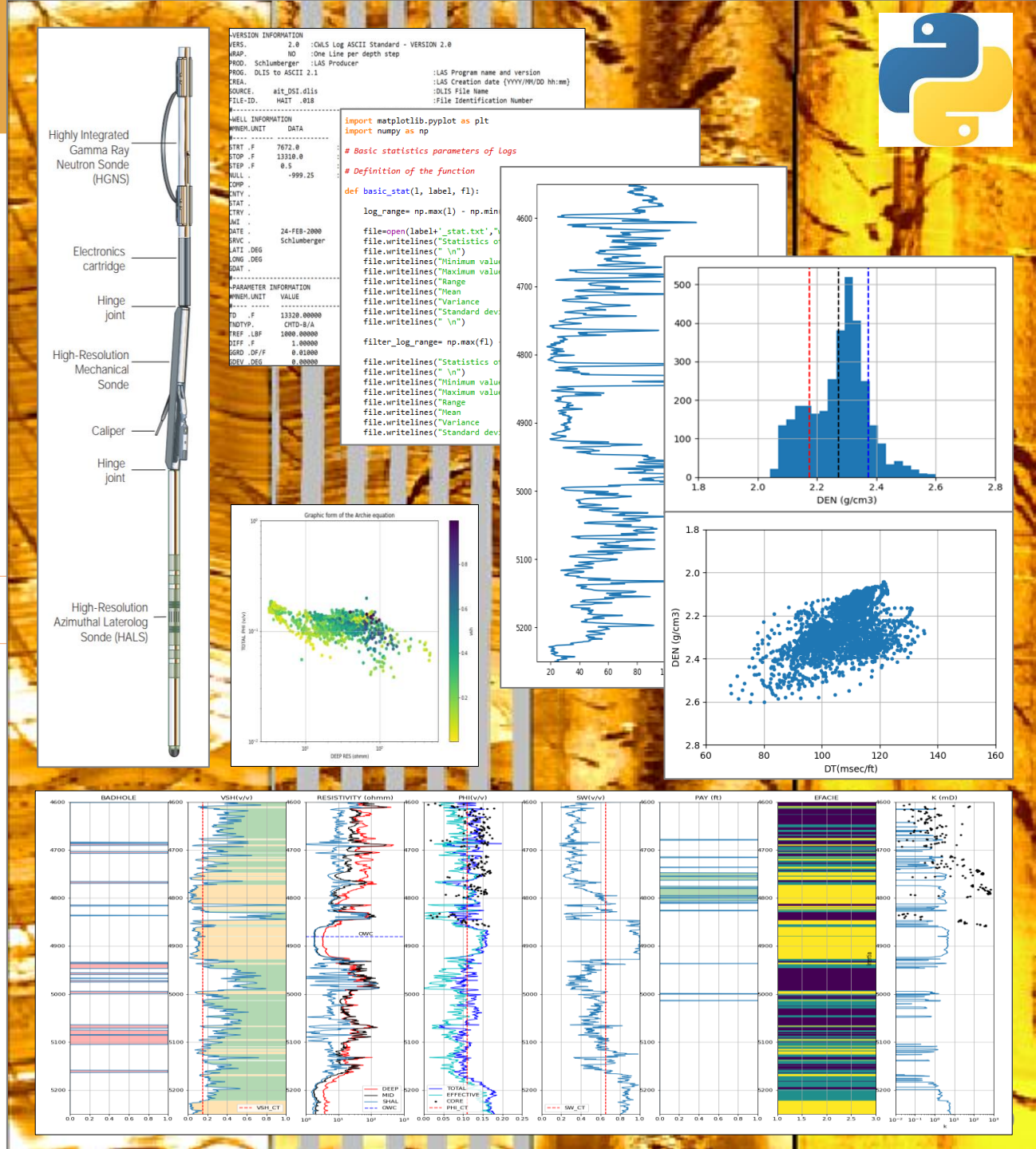
Int 1. Why learn to code?

Python in Log Analysis

A huge variety of critical data is acquired in the wells (open and case hole logs, pressure gradients, SWC, fluids, flow rates …)

In the last years Python has gained importance in the log analysis (LA) because:

- Python and its libraries are FREE
- It provides an excellent platform for teaching, developing and testing new procedures and algorithms
- Most important programs for LA incorporate utilities to code your own Python programs
- ML applications are gaining importance in LA, Python is one of the main programming languages for such advance applications.



Int 1. Why learn to code?

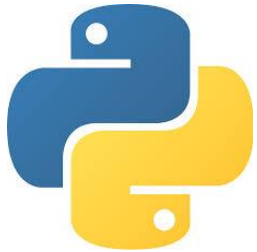
Python in Geophysical Operations

Common examples of Big Data are Seismic acquisition projects; hundreds of Gigabytes of data are acquired every day including seismic data, indicators, and ancillary data.

Python programming language has been used in Geophysical Operations for data analysis including:

- Reading and processing seismic data
- Source and receiver quality control analysis
- Statistics, attributes calculation
- Quality control

Int 1. Python programming language



Python is a widely used general-purpose, high-level programming language

- It was initially designed by Guido van Rossum in 1991 and then developed by Python Software Foundation
- It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code
- Main versions 2.X.X and 3.X.X

Advantages

- **Readability:** Python syntax is clear, making it easy to understand any piece of code.
- **Fast learning curve:** simple and intuitive syntax means it is easy to learn.
- **Open source:** Python is free and binaries are distributed by Python foundation.
- **Python is safe:** As there are no pointers like C, memory is protected, so the user will be able to see all errors and correct them.
- **Cross-platform:** Python can be run in Windows, Linux, or Mac.
- **All batteries inside:** standard and external libraries give to Python an unlimited power to tackle problems in different disciplines.

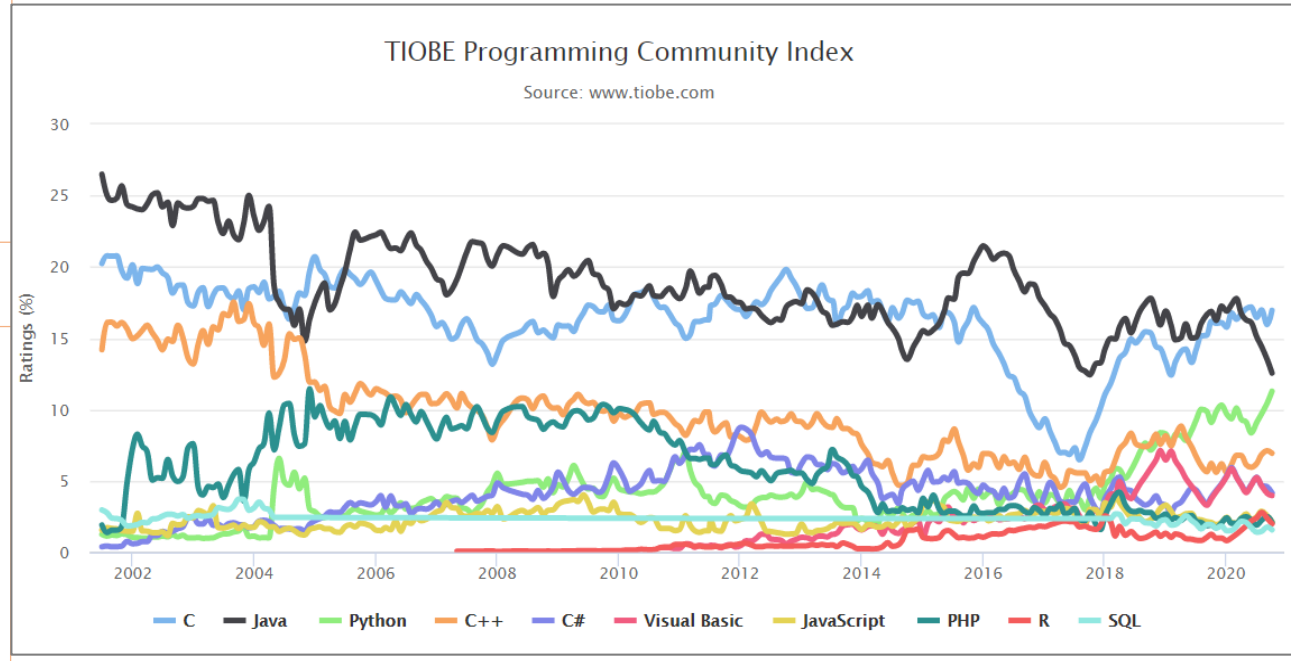
Int 1. Python programming language

Advantages

The TIOBE Programming Community index is an indicator of the popularity of programming languages. The index is updated once a month. The ratings are based on the number of skilled engineers world-wide, courses and third party vendors.

Python is among the top three languages in spite of been a general-purpose programming language.

From: https://www.tiobe.com/tiobe-index/



Int 1. Python programming language

Python libraries

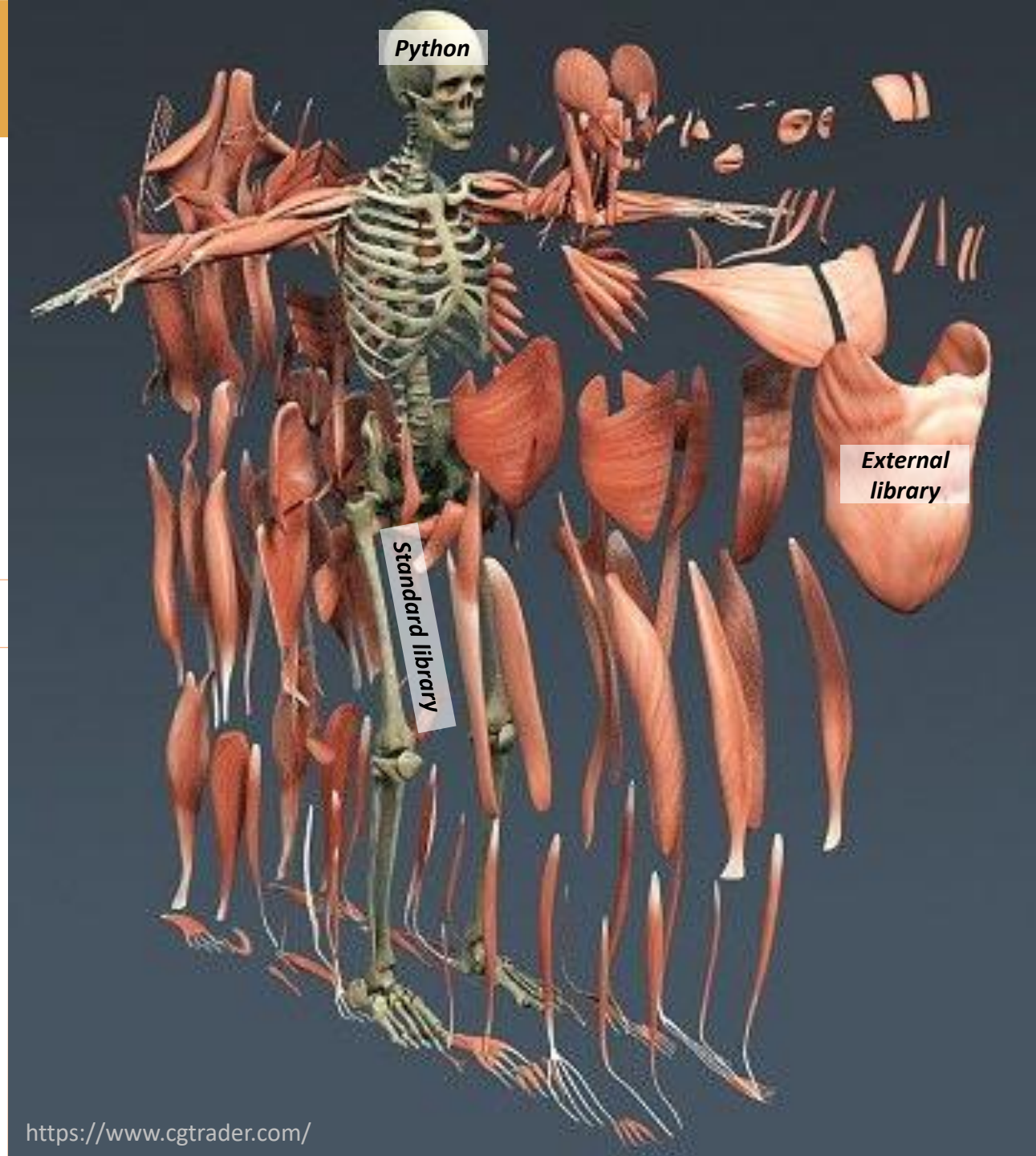
Python libraries or packages are the optional complements for the programming language (like muscles for the skeleton). There are two types of libraries:

Python's standard libraries (skull and other bones): They are installed with the program and include system, math, statistics, and similar components. Full list of SL at:

https://docs.python.org/3/library/

Python external libraries (muscles): They are developed by external groups and are maintained in parallel to the Python software. Hundred of thousand libraries which include arrays, plot, IA, data analysis, … Full list of EL at:

https://pypi.org/



Int 1. Python programming language

Python libraries

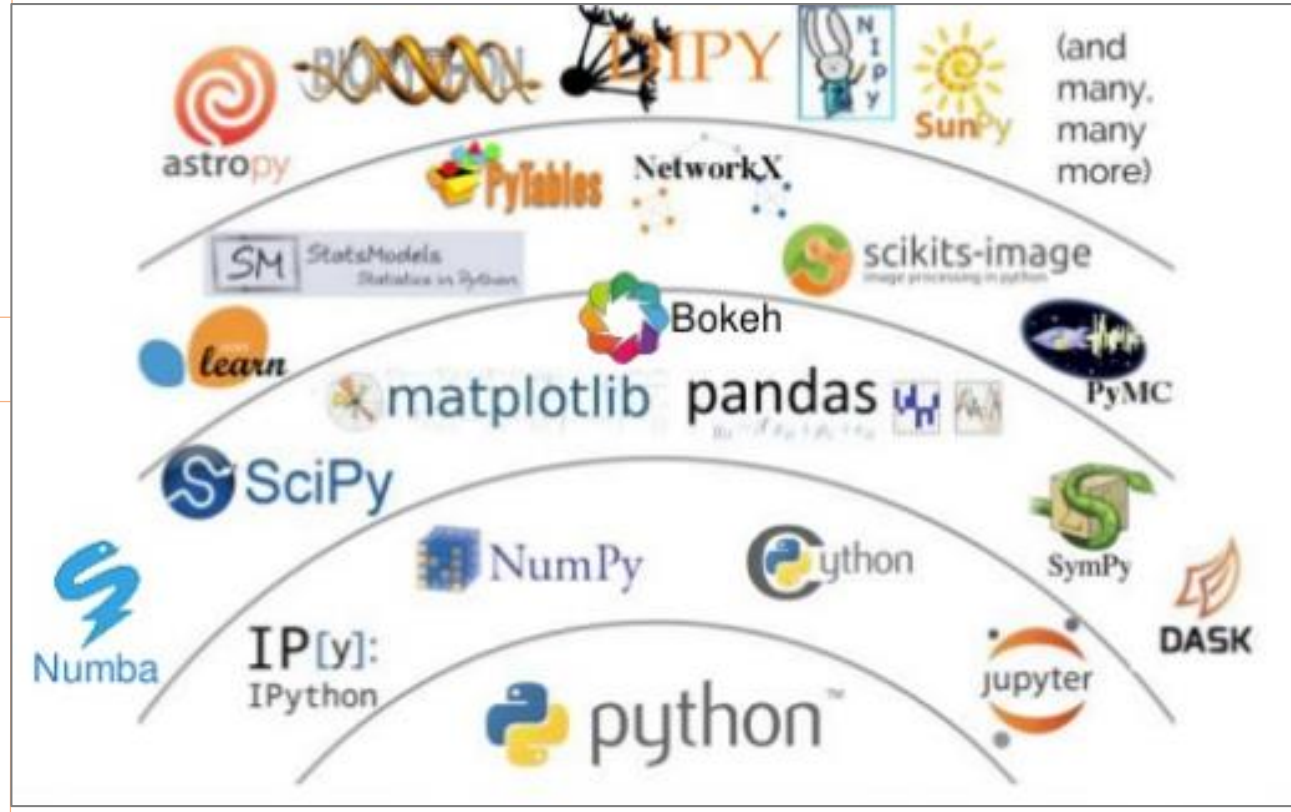
Python libraries or packages are the optional complements for the programming language (like muscles for the skeleton). There are two types of libraries:

Python's standard libraries (skull and other bones): They are installed with the program and include system, math, statistics, and similar components. Full list at:

https://docs.python.org/3/library/

Python external libraries (muscles): They are developed by external groups and are maintained in parallel to the Python software. They include arrays, plot, IA, data analysis, …

https://pypi.org/



Int 1. Python programming language

Success histories

- **Business:** Startup companies, data analysis, live stream analysis, Netflix, web development, Paypal, Uber.
- **Education:** Many universities use Python for teaching classes due to being free.
- **Engineering:** Has allowed development of rapid lab prototypes, as well as many software's using a Python shell.
- **Government:** Air traffic control, People data analysis
- **Scientific:** Reproducible research, data analysis.

Installing Python and the main libraries

Int 2. Installating Python

Two options

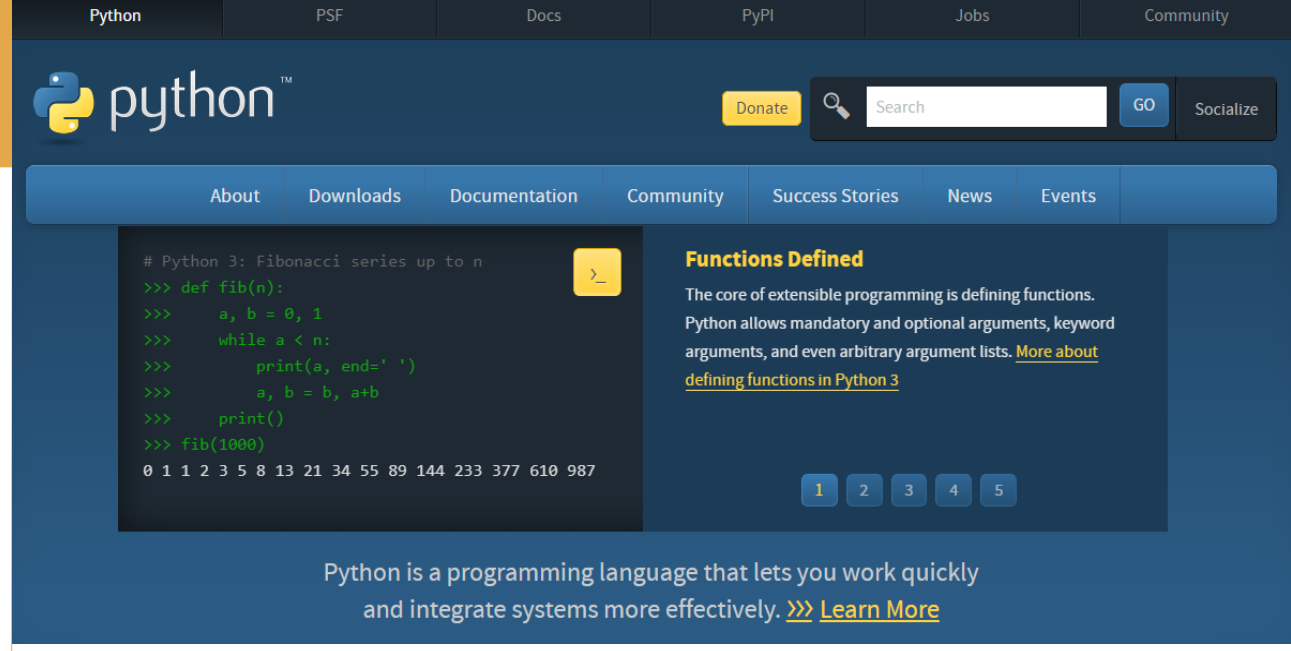
There are two ways to install Python in your computer:

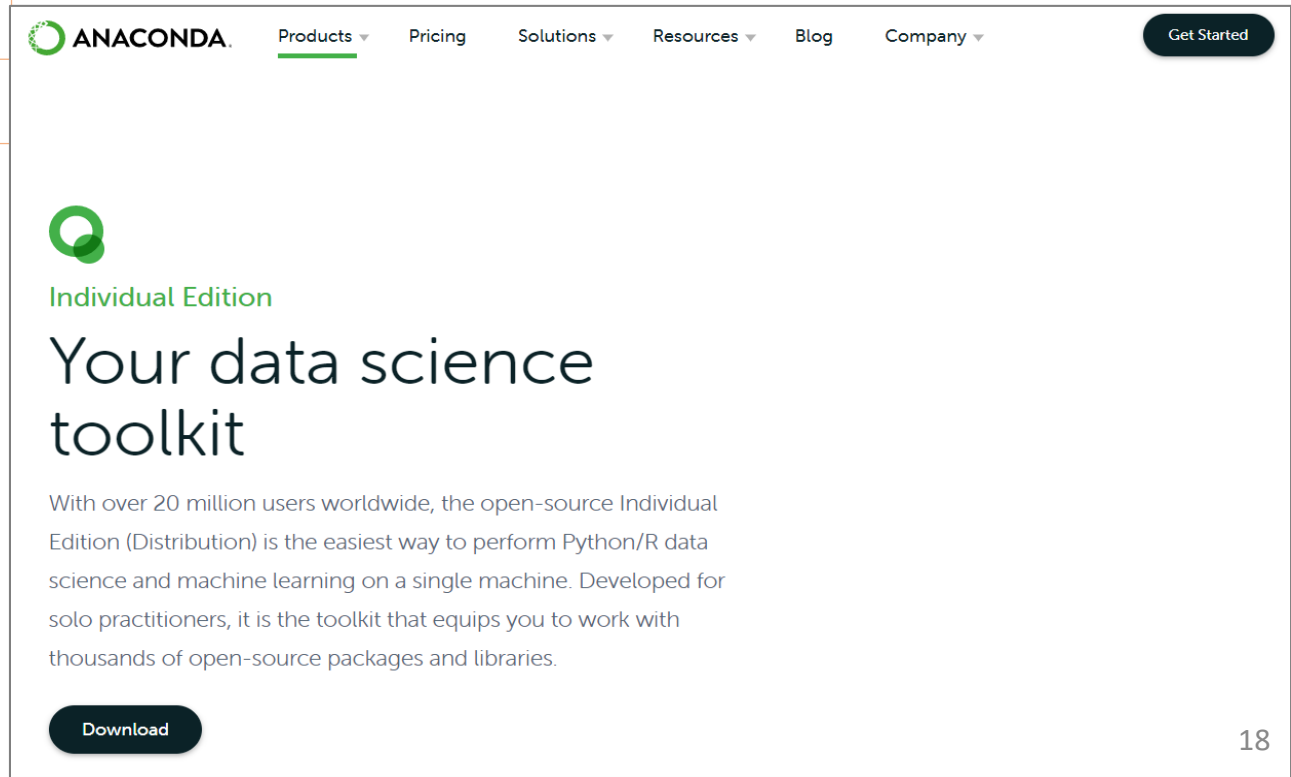
Native, pure Python software: Installing the pure open source Python software. **It is our preferred option and it will be covered in the next section.**

www.Python.org

Anaconda: This is a complete program which provide different environments for Python programming (recommended for Mac). It comes with several packages already installed. Depending on its use and version, it is covered by different licenses.

www.anaconda.com





Int 2. Standard Python installation

You can get different Python versions at:

https://www.Python.org/downloads/

Select the version required according to the operative system of your computer: Windows (32 or 64 bits), Linux, or Mac.

Tip: Do not install in the default Python directory, install in a simple directory under the root, where you have right of writing, for example: C:\Python386.

Due to issues with the main libraries (they are not ready for 3.9.1 yet), we are going to install an older version (3.8.6 for Windows 64 bits), you can get it at:

https://www.python.org/ftp/python/3.8.6/python-3.8.6-amd64.exe

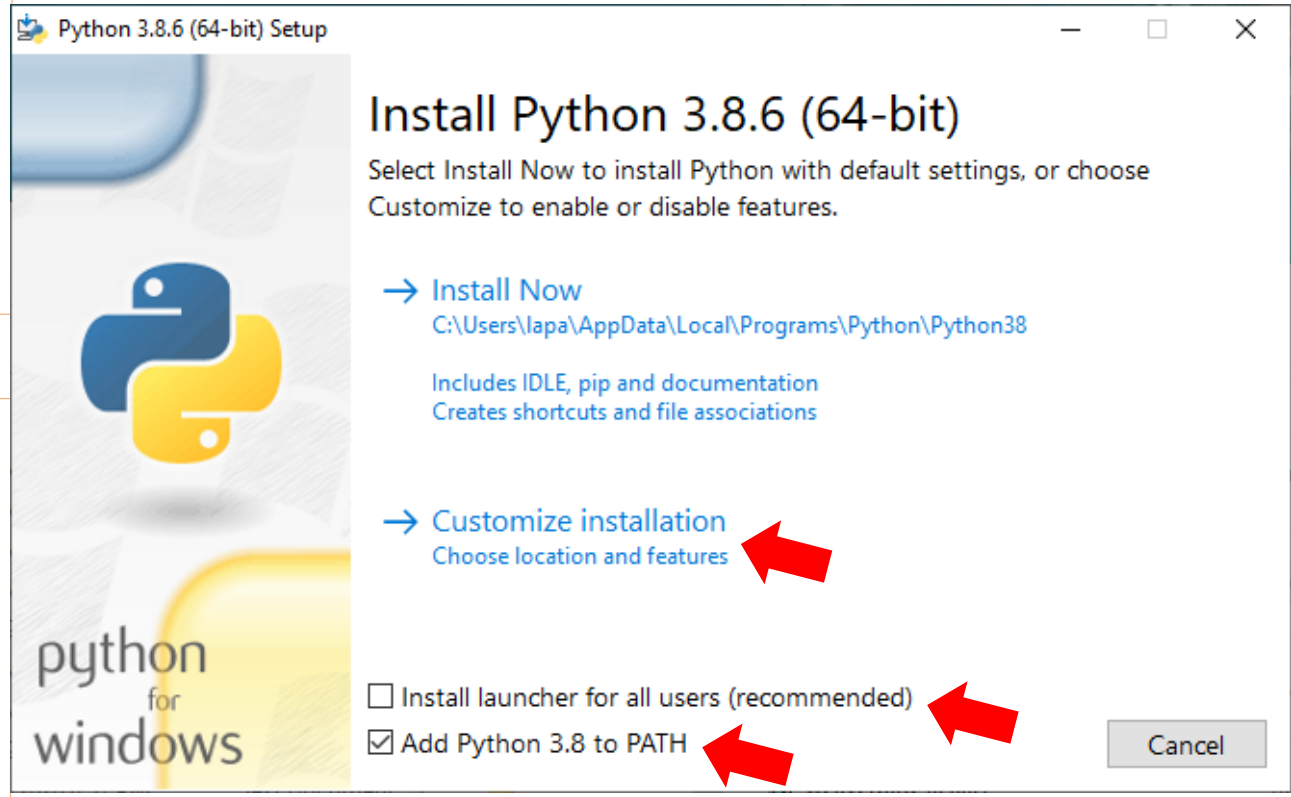
Int 2. Standard Python installation

Step 1

Click off: the Install launcher for all users to make the program only available for your user.

Click on: Add Python to PATH. This is especially important as later you will be able to run Python from any directory located in the computer structure.

Click in Customize installation.



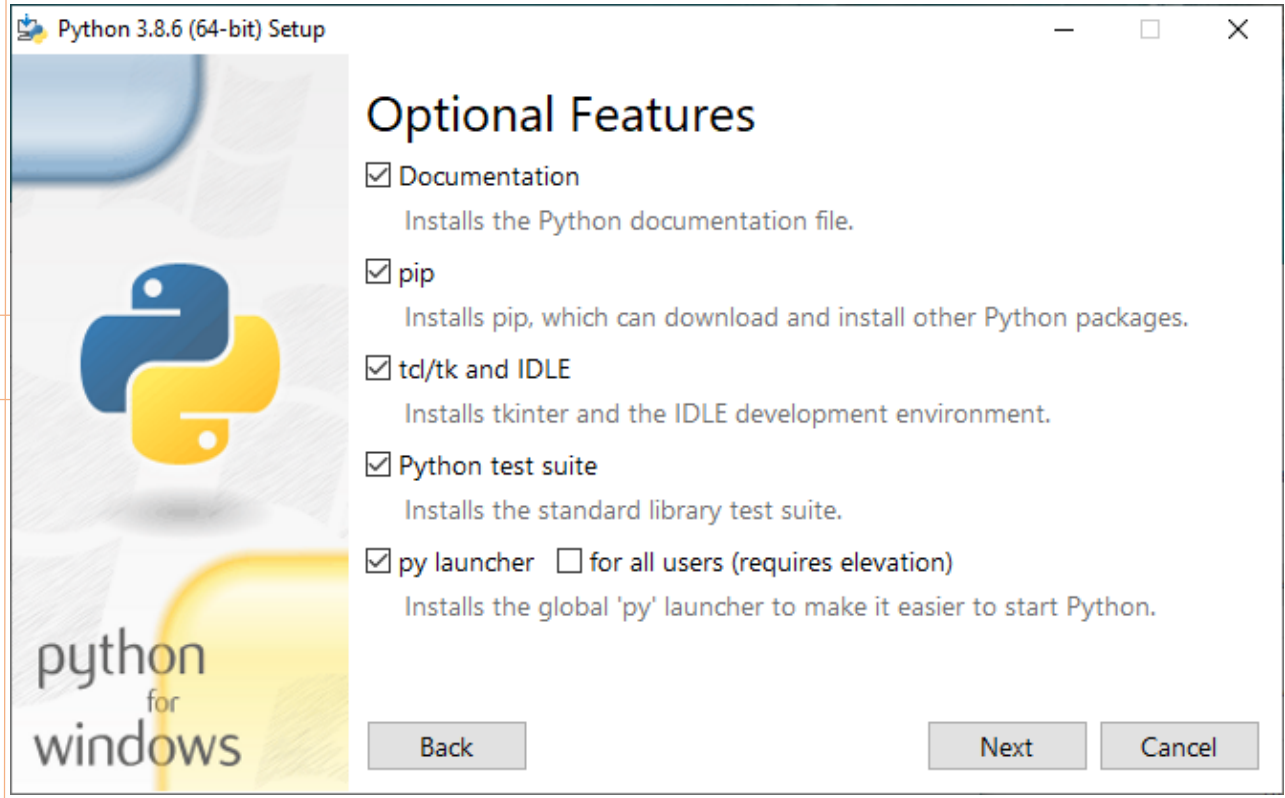
Int 2. Standard Python installation

Step 2

Click on all the options shows in this screen.

It is important to install pip because it allows you to install external Python packages.

Click on next button.



Int 2. Standard Python installation

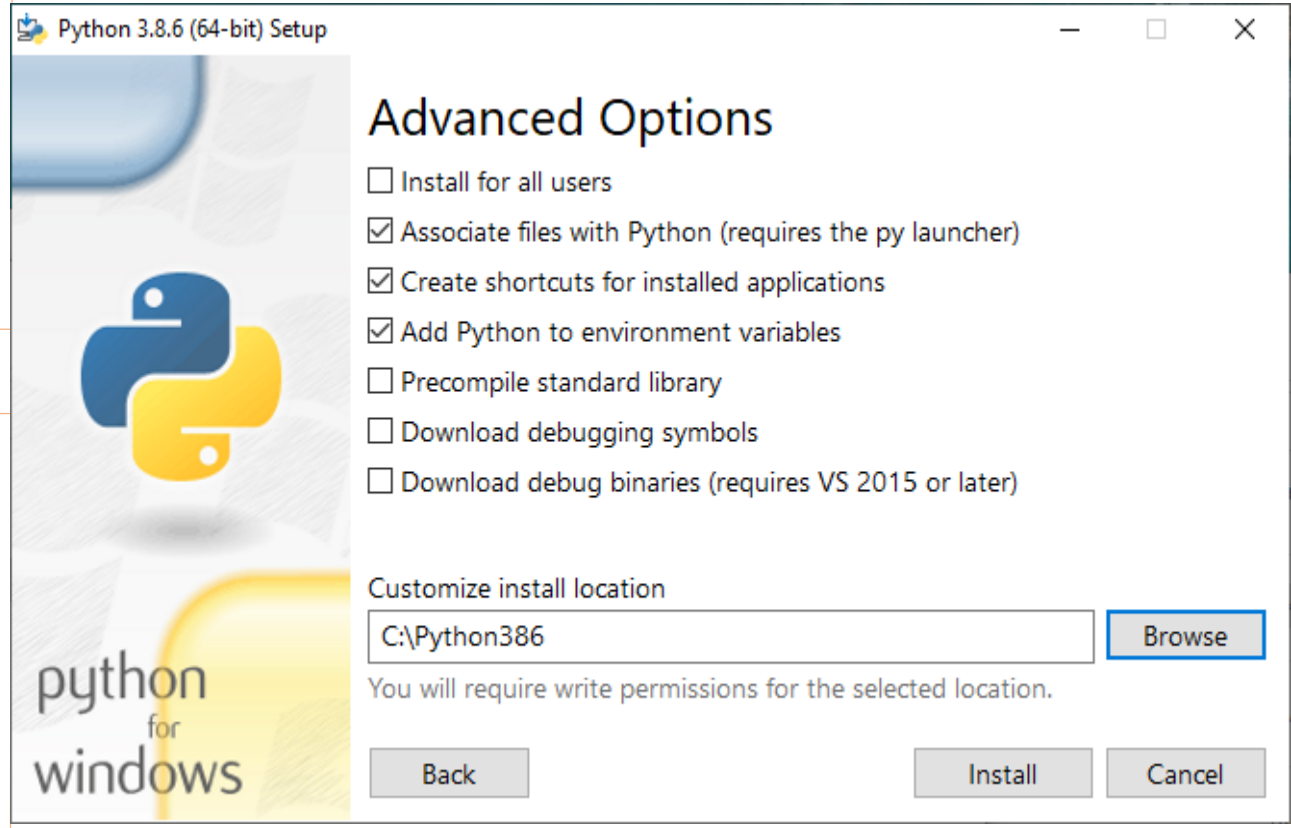
Step 3

Change the installation directory to a simple path in a place when the user has permission to write files and has enough space (at least 10 Gbs). Usually the place recommended is:

C:\Python386

Where number is the Python version to be installed.

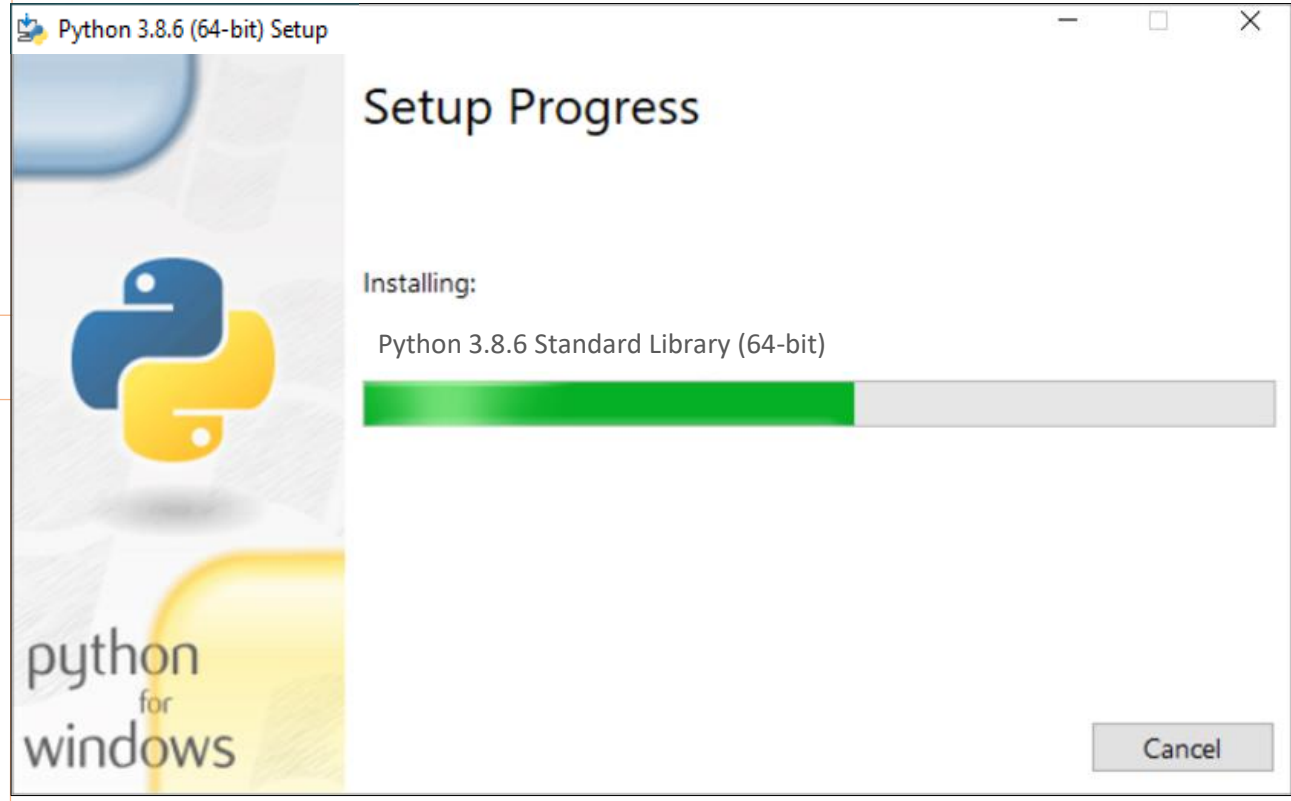
Then press the Install button.



Int 2. Standard Python installation

Step 4

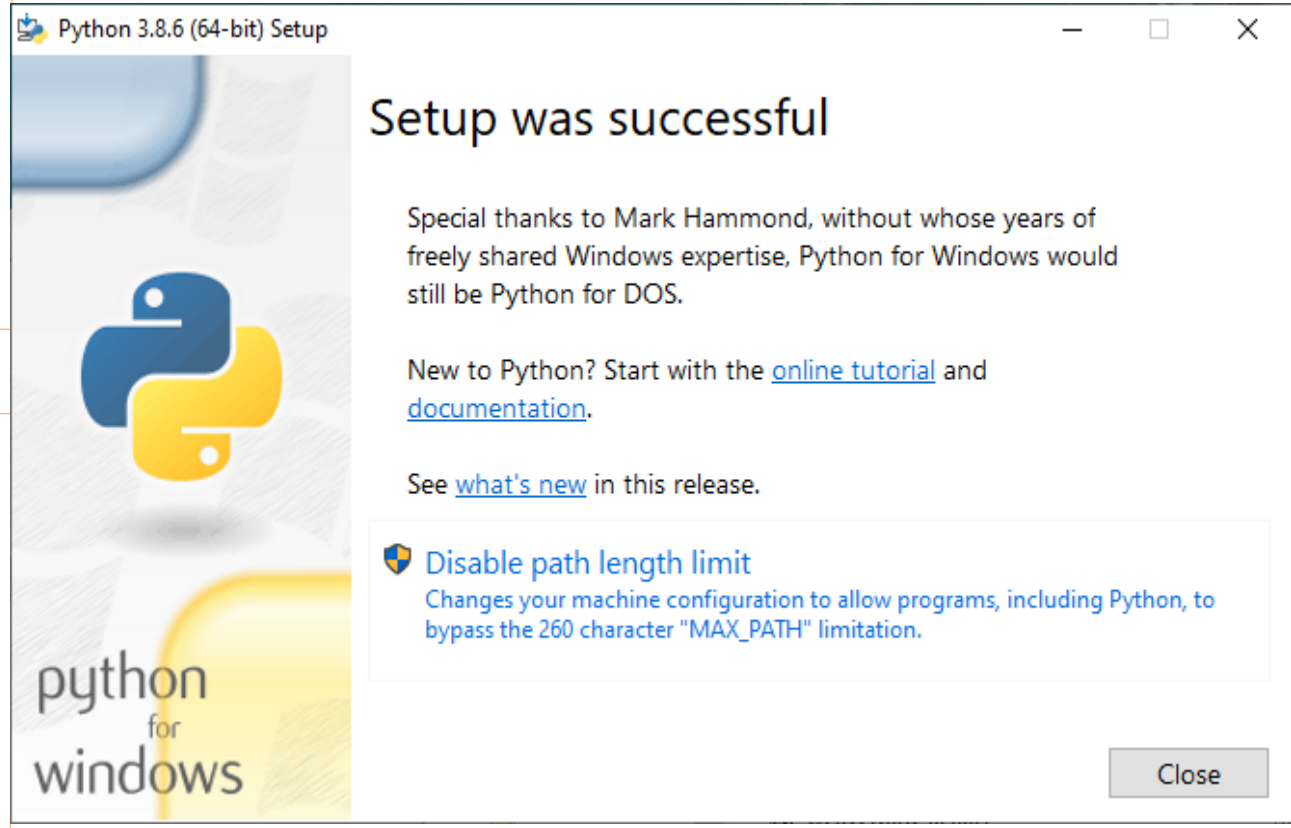
Wait for the program to install, it should take around five minutes.



Int 2. Standard Python installation

Step 5

Once the installation is complete, press the close button.



Int 2. Standard Python installation

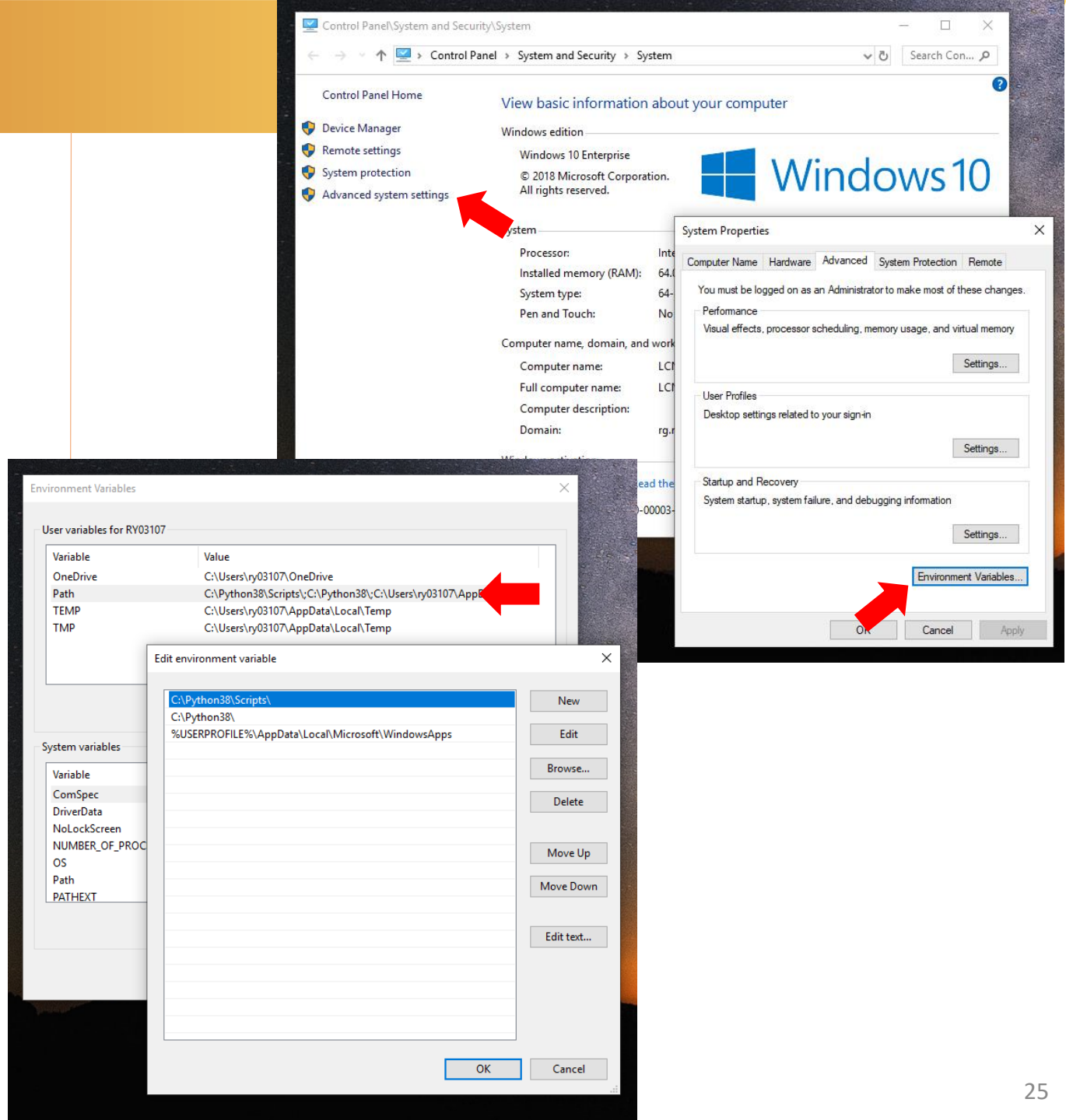
Step 6

It is very important to verify that the PATH environmental variable points to the following directories:

C:\Python386\Scripts

and

C:\Python386



Int 2. Standard Python installation

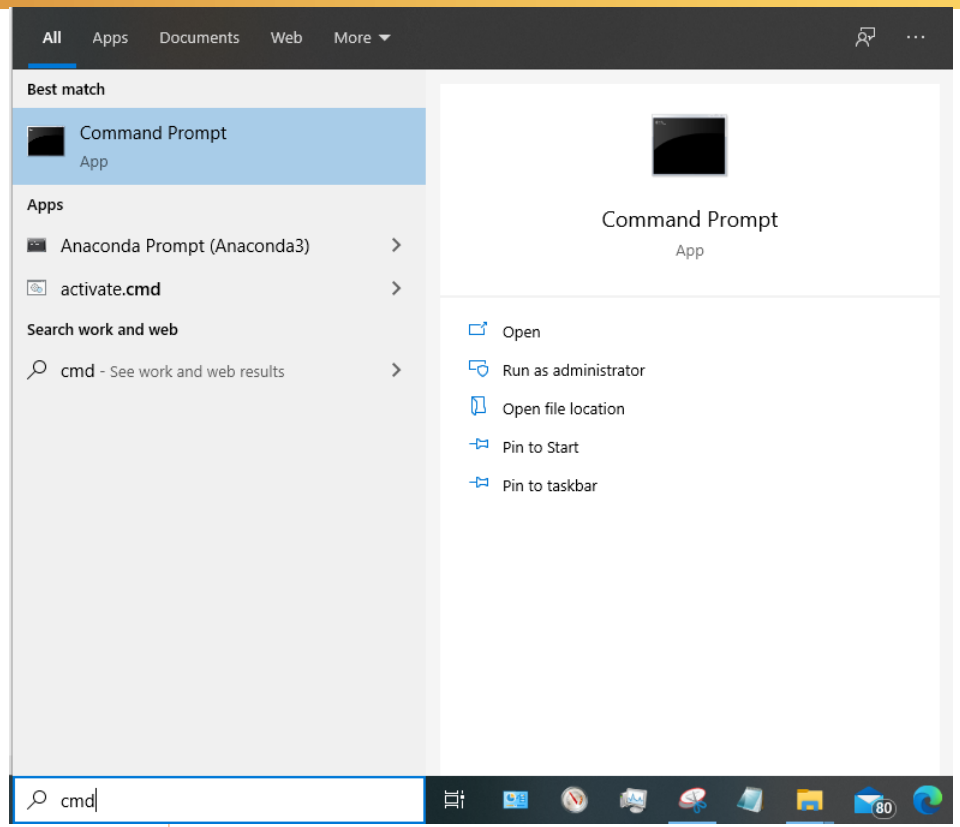
Step 7

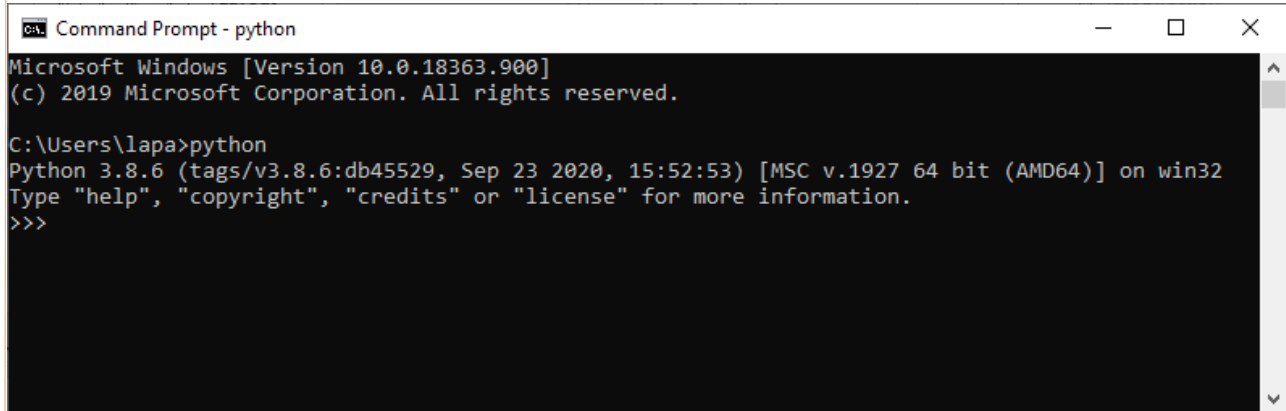
Finally, to verify that the program was installed:

In the windows search bar, type *cmd*, this will open a *Command Prompt* window (cmd).

Type *python* in the command line, this will enter inside the Python interpreter.

To leave the Python interpreter, before installing any library, by typing *exit() or Ctrl + z*





Int 2. Installing external packages

pip: makes everything easy for you

The command pip (package installer for Python) allows the installation of any package (libraries) available in the Python community, the command syntax is:

pip install package
pip install package2
pip install package3

Or in sequential manner:

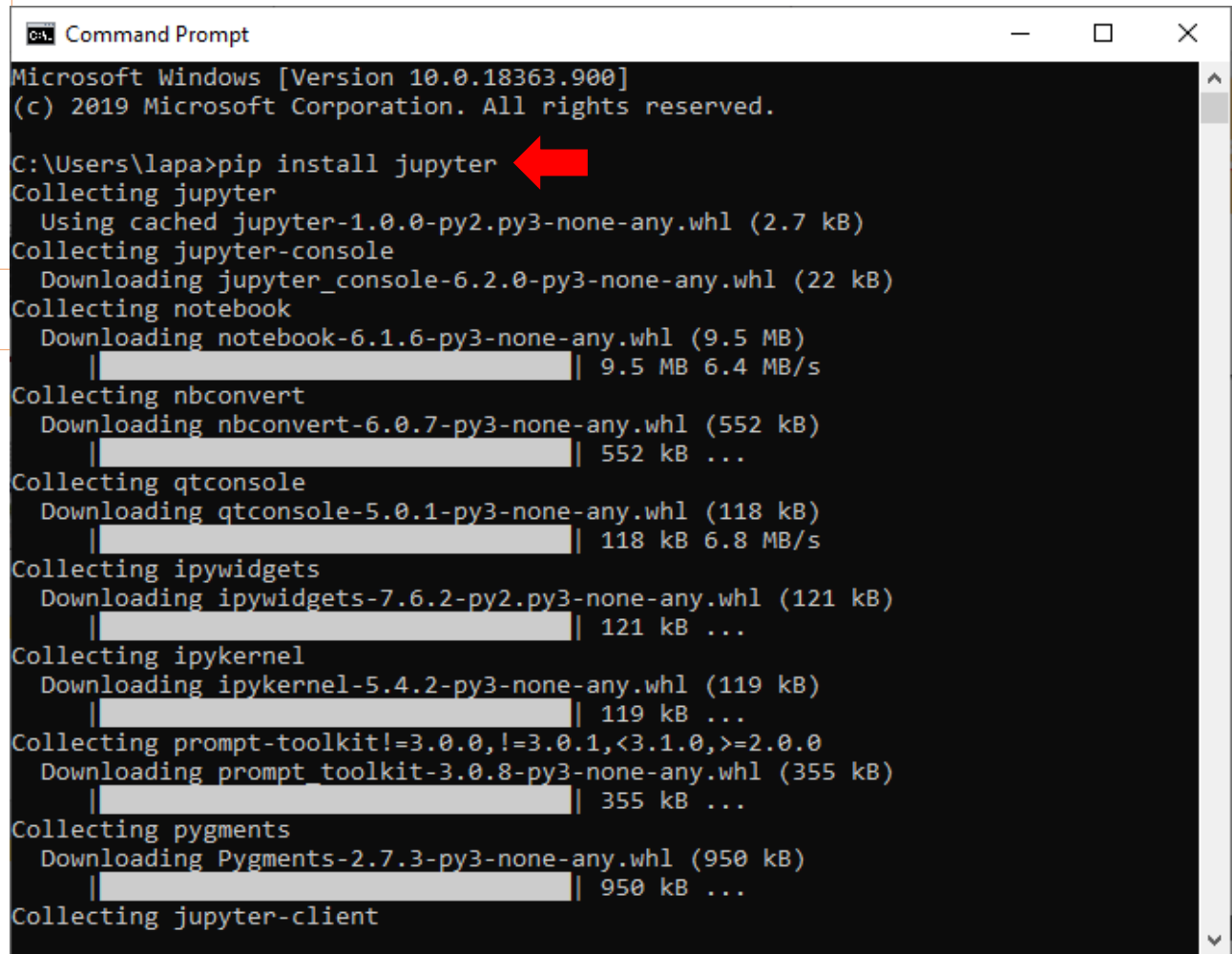
pip install package package2 package3

This command is executed in a command window while connecting to Internet.

All external packages can be found at: https://Pypi.org/

If you have another pip installations, be sure to execute these commands in the right location, such as:

C:\Python386\Scripts

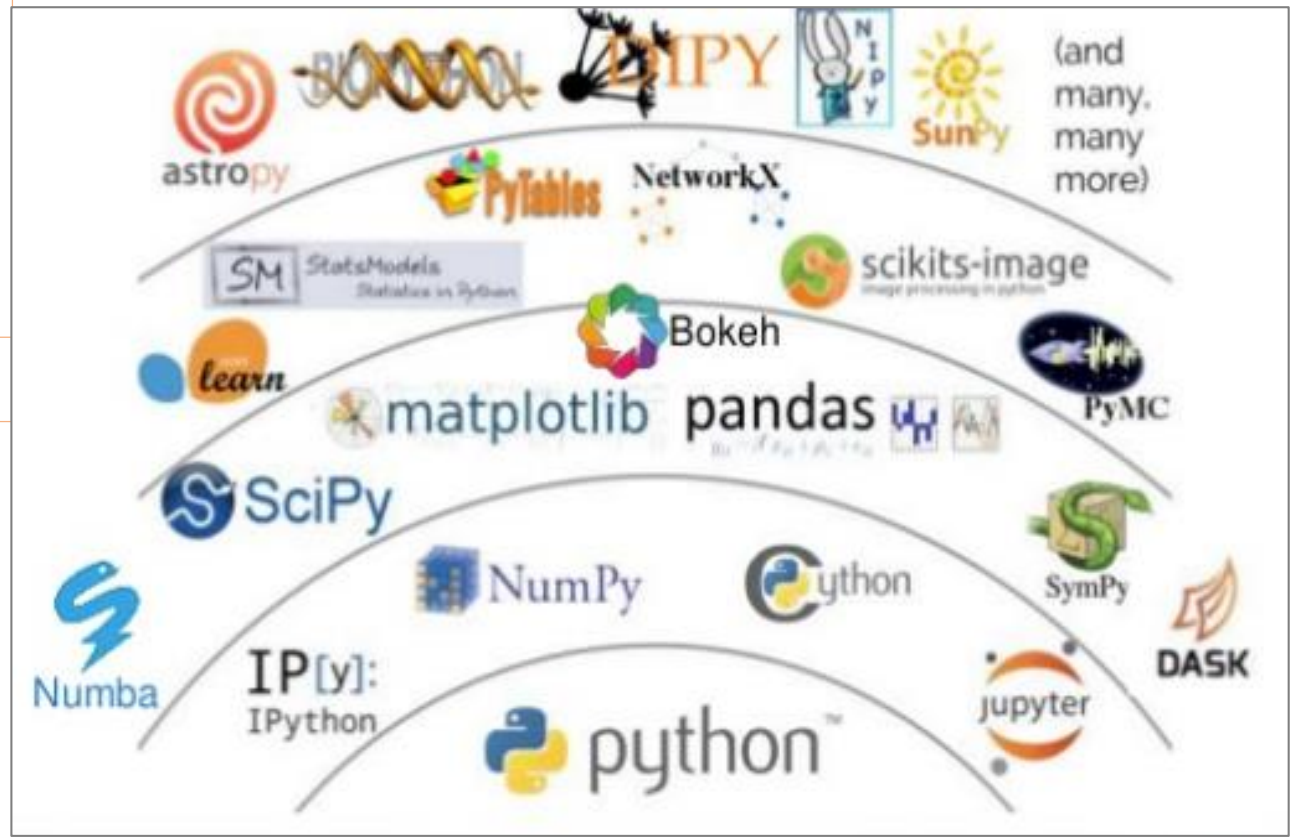


Int 2. Installing external packages

Important packages for geosciences

The most important packages or libraries for geosciences are:

Jupyter	**Development and documentation environment**
numPy	**Scientific computing (arrays, algebra, …)**
Matplotlib	**Matlab style charts and graphs**
Pandas	Data manipulation
sciPy	Statistics, algebra and digital processing
Pillow	Images manipulation
obsPy	Seismological toolbox
Pytorch	Neural networks
keras	Neural networks and AI
…	…



To install the libraries for session 1, execute the following command in a cmd window (do not activate Python):

pip install jupyter numpy matplotlib

Int 2. Python links

Tutorials

https://wiki.python.org/moin/BeginnersGuide

https://learnxinyminutes.com/docs/python/

https://www.stavros.io/tutorials/python/

https://realpython.com/

Beyond the basic

https://medium.com/

https://towardsdatascience.com/

https://www.youtube.com/c/PyCon2020s

https://www.youtube.com/c/PyCon2020

Reproducible research and Jupyter Notebook

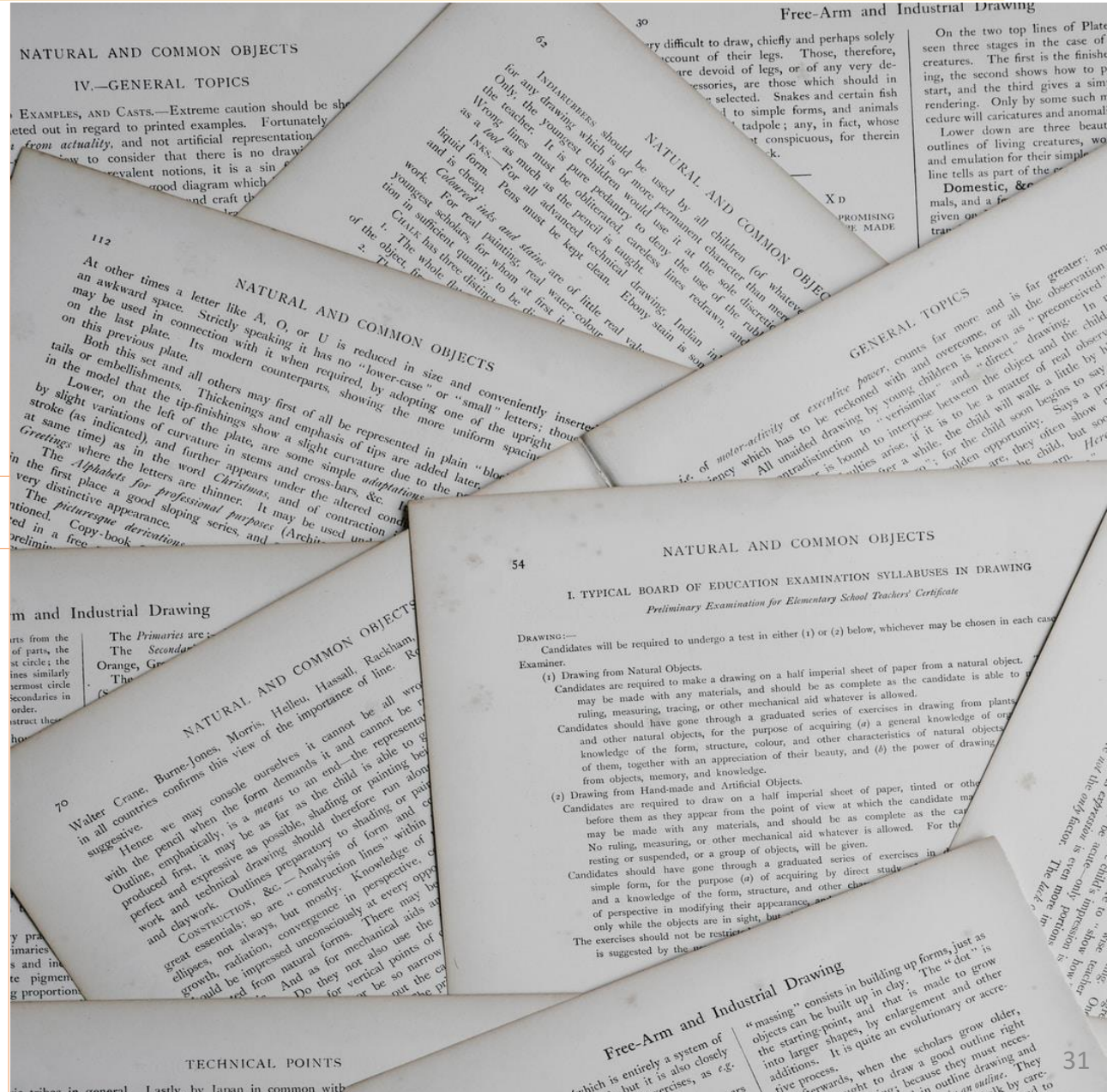
Int 3. Reproducible research

Many times you read a paper but find yourself unable to reproduce the results.
This is a common problem within scientific community!

We can find many examples of data analysis papers which cannot be replicated by other scientists, because the content of the papers usually demonstrates a theoretical background within the algorithm and results, but is apparent in the lack of actual code used.

Literate programming is the basic idea behind dynamic documents and was proposed by Donald Knuth in 1984. Originally, it was for mixing the source code and documentation of software development together.

Reproducible and replicable research refer to a process of research where researchers can share transparent and reliable work processes online so their work can be both repeated and replicated by others.



Int 3. Reproducible research

Why we need to document our programs?

Usually, when we start to code in our work several small personal scripts are generated for specific datasets which are later shared with our team or co-workers.

A computer code or script is an ASCII file with all the instructions provided in a sequential order to do a specific task by the computer. With time, scripts without any documentation start to build up in directories, leading for older scripts to be more difficult to adapt than creating a new one.

Jupyter Notebooks it is a program that has been used in the community for the last 10 years, allowing the combination of code and documentation within the same document.

Int 3. Jupyter Notebook

What is a Jupyter Notebook?

The Jupyter Notebooks is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations, and narrative text.

Uses include data cleaning and transformation, numerical simulation, statistical modelling, data visualization, machine learning, and many more.

In practice, Jupyter is a powerful programming environment for data analytics and is used with dozens of programming languages including Python.

5.3.2 Splitting the image file in individual channels

To analyze a color image it has to be separated in their RGB channels or bands, such as bellow:

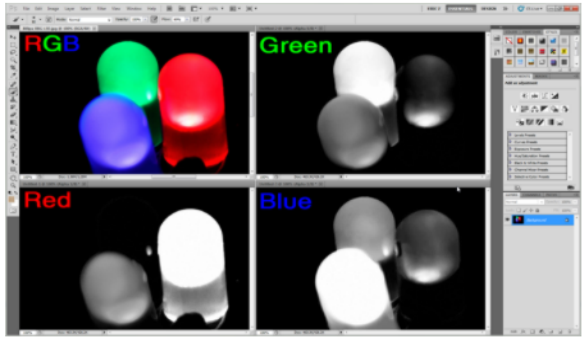


Image from: https://stackoverflow.com/questions/46139376/how-to-extract-red-green-and-blue-channels-of-bitmap-in-android

Each channel or band is composed by the same amount of pixels (716 x 1000 in our case) in which the amount of light is represented by a 8 bit scale, 0 in the pixels with no light (black) and 255 in pixels with maximum of light (white). The reconstruction of different colors is archived by the combine information of the three channels:

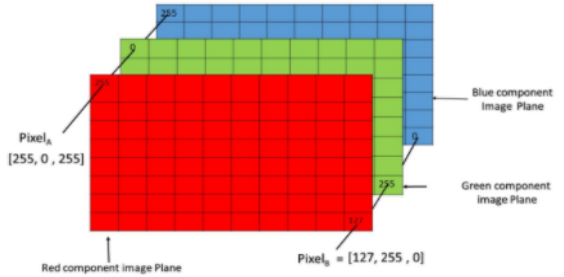


Image from: https://www.geeksforgeeks.org/matlab-rgb-image-representation/

At the following site there is a color calculator base on different scales: https://www.w3schools.com/colors/colors_rgb.asp

In [49]:

```
# Spliting the image in its RGB channels

rock_split = Image.Image.split(rock_class)

red = rock_split[0]
green = rock_split[1]
blue = rock_split[2]

print(rock_split)
print()
print(type(rock_split))

rock_split

# rock_split is not longer an image

# L: black & white image
```

```
(<PIL.Image.Image image mode=L size=1000x716 at 0x297387B29B0>, <PIL.Image.Image image mode=L size=1000x716 at 0x297387B26A0>,
<PIL.Image.Image image mode=L size=1000x716 at 0x297387B2A90>)

<class 'tuple'>
```

Out[49]:

```
(<PIL.Image.Image image mode=L size=1000x716 at 0x297387B29B0>,
 <PIL.Image.Image image mode=L size=1000x716 at 0x297387B26A0>,
 <PIL.Image.Image image mode=L size=1000x716 at 0x297387B2A90>)
```

Int 3. Jupyter Notebook

Example of a Jupyter Notebook

This is an example of a Jupyter Notebook. This document is a combination of the following features:

- Text with different font types
- Images and animations
- HyperLinks
- Equations
- Computer Code

Jupyter has many extensions including:

- Code automatic documentation
- Computer widgets

The SIR epidemic model

Ref: https://scipython.com/book/chapter-8-scipy/additional-examples/the-sir-epidemic-model/ (https://scipython.com/book/chapter-8-scipy/additional-examples/the-sir-epidemic-model/)

A simple mathematical description of the spread of a disease in a population is the so-called SIR model, which divides the (fixed) population of N individuals into three "compartments" which may vary as a function of time, t:

S(t) are those susceptible but not yet infected with the disease

I(t) is the number of infectious individuals

R(t) are those individuals who have recovered from the disease and now have immunity to it.

The SIR model describes the change in the population of each of these compartments in terms of two parameters, β and γ. β describes the effective contact rate of the disease: an infected individual comes into contact with βN other individuals per unit time (of which the fraction that are susceptible to contracting the disease is S/N). γ is the mean recovery rate: that is, 1/γ is the mean period of time during which an infected individual can pass it on.

The differential equations describing this model were first derived by Kermack and McKendrick [Proc. R. Soc. A, 115, 772 (1927)]:

$$\frac{\mathrm{d}S}{\mathrm{d}t} = -\frac{\beta S I}{N},$$
$$\frac{\mathrm{d}I}{\mathrm{d}t} = \frac{\beta S I}{N} - \gamma I,$$
$$\frac{\mathrm{d}R}{\mathrm{d}t} = \gamma I.$$

In [1]:

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
```

In [2]:

```
# Variable definition

# Total population, N.
N = 1000

# Initial number of infected and recovered individuals, I0 and R0.
I0, R0 = 1, 0
# Contact rate, beta, and mean recovery rate, gamma, (in 1/days).
beta, gamma = 0.2, 1./10
```

Int 3. Jupyter Notebook link

Why Jupyter Notebook is so popular among Data Scientists?

https://iseprostorglive.blob.core.windows.net/user-assets/projects/AIhDsT0NGhVzHl2q9duLx3hVBPFfe6aW/assets/mc-6a54e3f64a60e9a7a8cbf2d58713168c.mp4

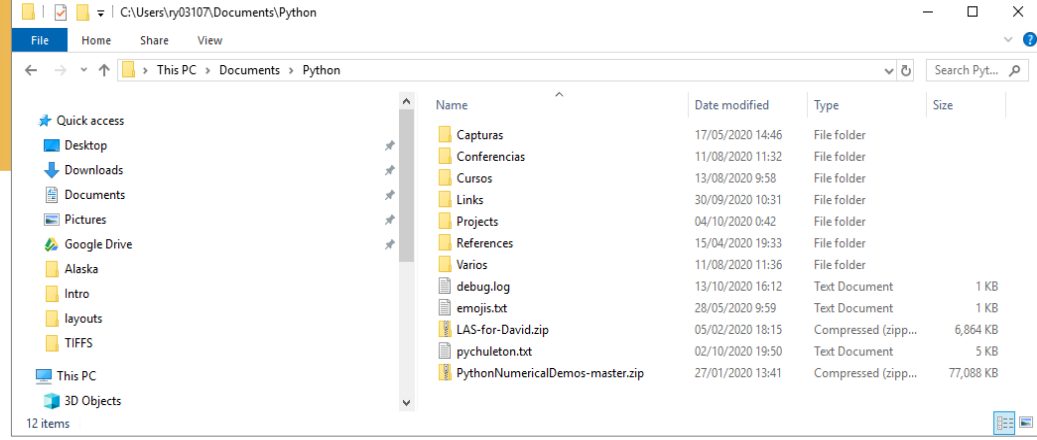
Int 3. Jupyter Notebook

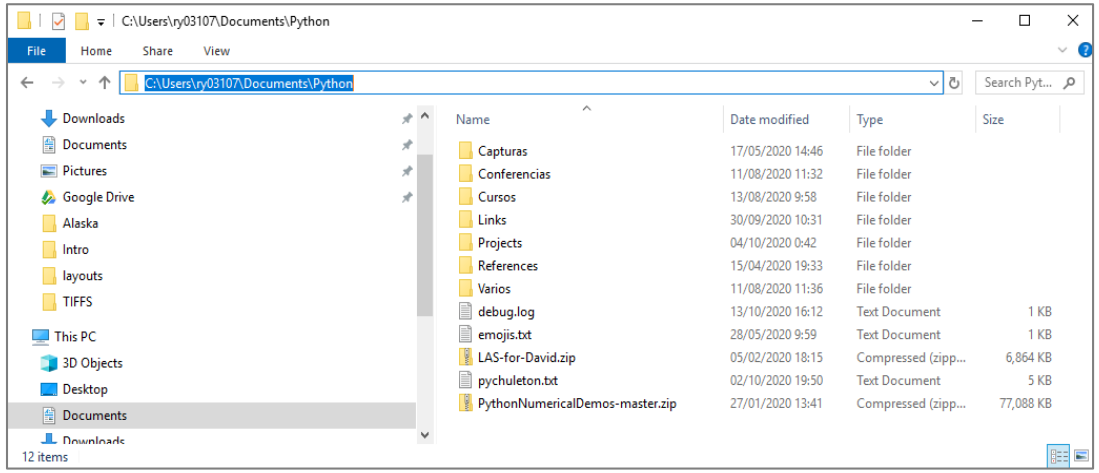
Running Jupyter

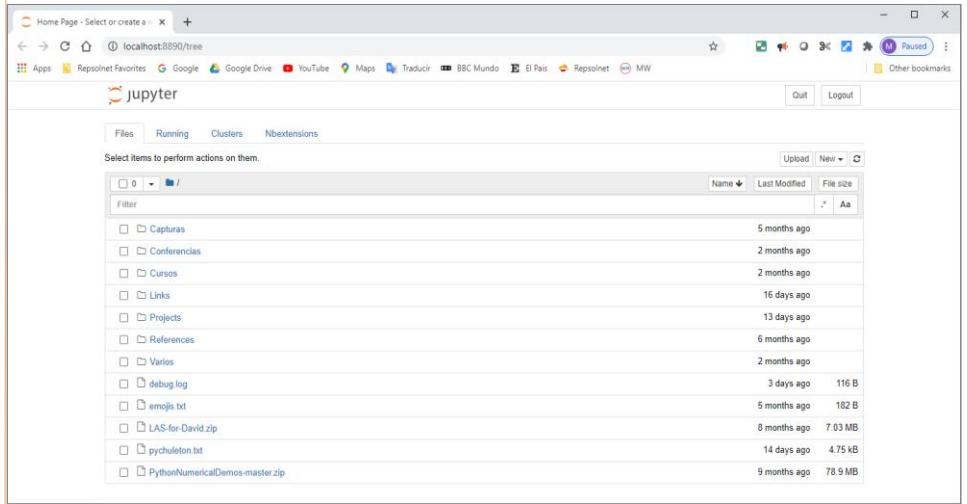
We already covered the installation of packages (Jupyter included) in page 28 and 29. To run Jupyter follow the following steps:

- Open a File explorer of the directory where you chose to store the notebooks.
- Select the name of the folder
- While the name is still selected (in blue) type *jupyter notebook*

This will open a command window (don't close it) as well as the Jupyter environment on your default web browser.







Int 3. Jupyter notebook links

Notebook examples:

https://juPyter-notebook.readthedocs.io/en/stable/examples/Notebook/examples_index.html

Notebooks by area:

https://github.com/juPyter/juPyter/wiki/A-gallery-of-interesting-Jupyter-Notebooks

Python in the Cloud

Int 4. Python in the Cloud

Free services for starting users

There are several services that allow users to run Python code in the Cloud. These services are free for starting users. The most common services are:

- Binder
- Kaggle kernels
- **Google Colaboratory (Colab)**
- Microsofft Azure notebooks
- CoCalc
- Datalore

Don´t use confidential data or codes in these services

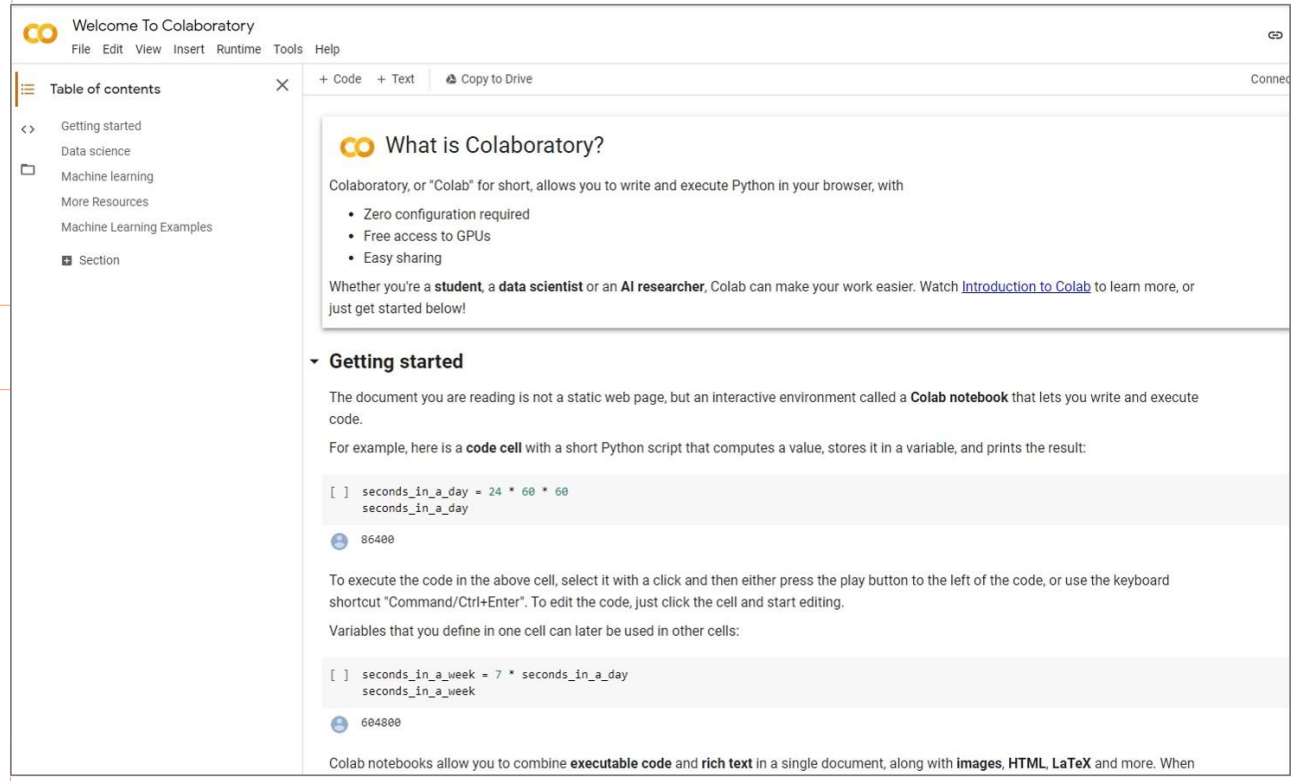
Int 4. Google Colaboratory

What is Google Colaboratory?

Google Colaboratory (Colab) is a platform from that allows you to run Python codes on a Jupyter Notebooks on the Cloud however, **in this course we preferred to work with Jupyter Notebook over the standard Python installation**.

It is a free service, with the only limitation being that users cannot be connected for more than 12 hours of run time. Sometimes the program will not run immediately due to service congestion

This system is good for practicing Python on the Web



Int 4. Google Colaboratory

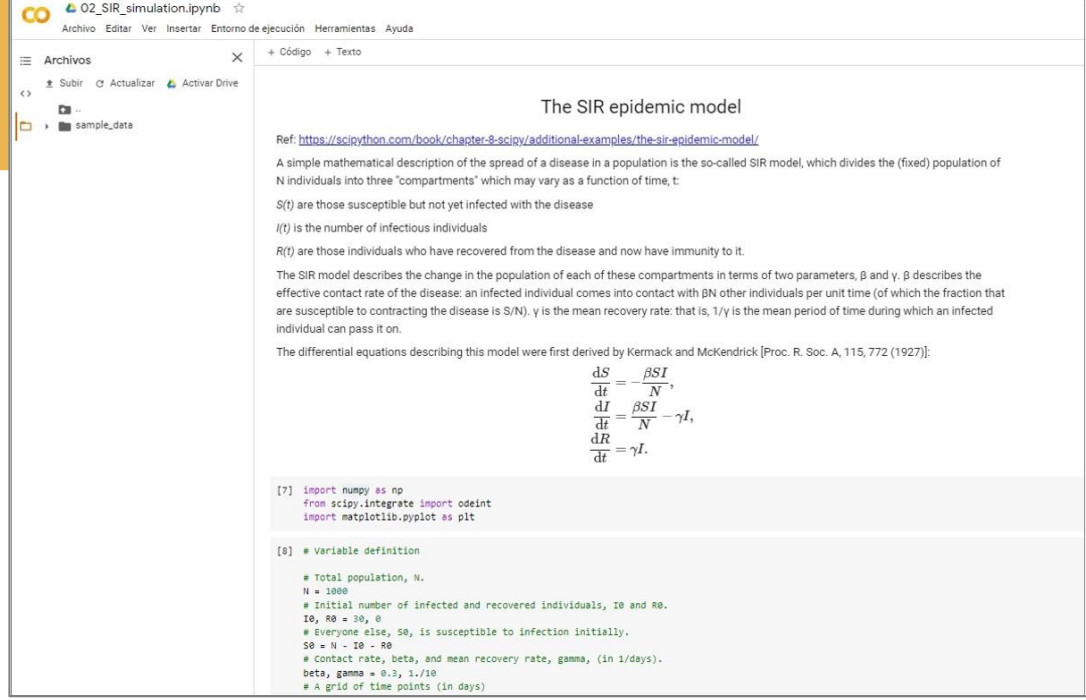
How to run Google Colab?

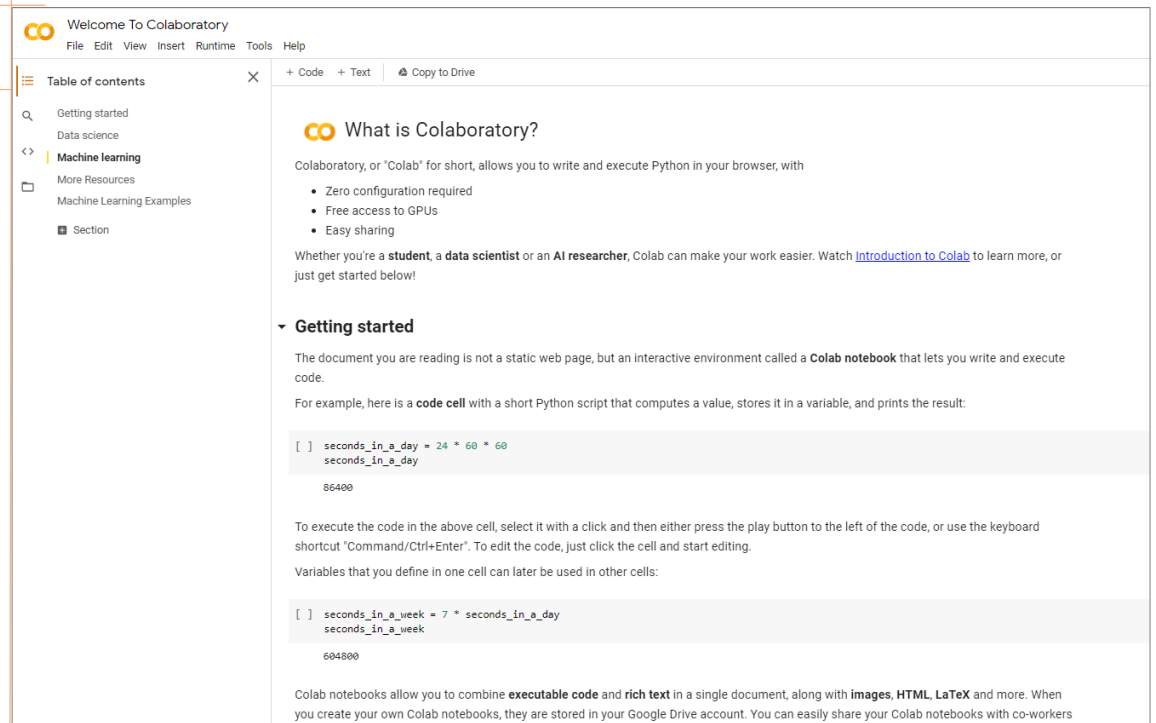
To access Colab you have to have an active Google account and access the following website in the navigator:

https://Colab.research.Google.com/

The navigator will then enter inside the environment and display a Welcome window with examples. You also can create a new notebook or open a previous saved in your Google Drive. The data needs to be uploaded to your Google Drive. Then, in order to run your programs, you need to connect to a remote computer (CPU, GPU or TPU).

The environment is similar to the Jupyter Notebook working on your standard Python, in fact it uses the same .iPynb files.





Int 4. Google Colaboratory links

What is Google Colab?

https://iseprostorglive.blob.core.windows.net/user-assets/projects/AIhDsT0NGhVzHl2q9duLx3hVBPFfe6aW/assets/mc-c41aae8939705703b234e4f8bb389b34.mp4

Some videos about Google Colab

(Spanish) https://www.youtube.com/watch?v=Vhl91Az-rzo
(English) https://www.youtube.com/watch?v=i-HnvsehuSw

A good tutorial in Google Colab

https://towardsdatascience.com/getting-started-with-Google-Colab-f2fff97f594c

Conclusions

- Coding is a critical skill to master data analysis within our actual and future jobs.
- Coding expands your data analysis possibilities as well as allows geoscientists to have different perspectives.
- As starting point, let install the Python from the python.org website.
- Google Colaboratory or Microsoft Azure Notebooks are useful, but only for practicing and learning. Do not use confidential data or codes in these services.

Annexes

Links from individuals or organizations

Geology and Python: http://geologyandpython.com

Agile: https://agilescientific.com/ & https://github.com/agile-geoscience

Software Underground: https://softwareunderground.org

GeoSci: http://geosci.xyz

SEG tutorials: https://github.com/seg/tutorials

Books

https://jakevdp.github.io/WhirlwindTourOfPython/

https://jakevdp.github.io/PythonDataScienceHandbook/