Introduction to Scientific Python

Iñigo Aldazabal Mensa – Centro de Física de Materiales
(CSIC-UPV/EHU)
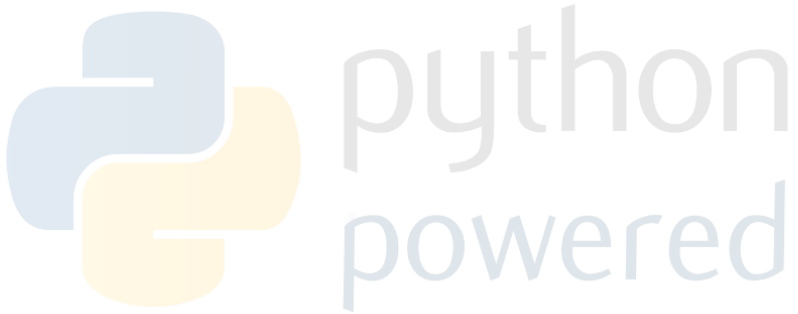
Software Carpentry Workshop

```
2023-01-10-cfm
```

CFM, San Sebastian, 27th January 2023

Why Python?

Why Python?

Python is inherently slow compared to C/C++ or FORTRAN, so why Python for Scientific Computing?

Python is slow, but...

Syntactically, Python code looks like executable pseudo code:

- A Python program can have 5-10 times less lines than its C or FORTRAN counterpart
- Thus, program development using Python is 5-10 times faster than using C/C++...

and...

The number of bugs in a program scales linearly with the number of lines of the program.

Why Python?

We should forget about small efficiencies, say about 97% of the time:
premature optimization is the root of all evil *(in programming).*

– Computer Programming as an Art (1974), Donald Knuth

In software engineering, it is a good approximation that 90% of the execution time of a computer program is spent executing 10% of the code

```
print("Hello, world!")
```

Why Python?

We should forget about small efficiencies, say about 97% of the time:
***premature optimization is the root of all evil** (in programming).*

– Computer Programming as an Art (1974), Donald Knuth

In software engineering, it is a good approximation that 90% of the
execution time of a computer program is spent executing 10% of the
code

*The best approach is often to write only the performance-critical parts
of the application in C++ or Java, and use Python for all higher-level
control and customization.*

– Guido van Rossum

Why Python?

We have (for free)

- A general purpose language with a huge spectrum of freely available libraries for almost anything you can think of.
- A very easy to learn (and read) language that smoothly interfaces with C/C++ and FORTRAN (eg. calculation kernels).
- Lots of wrappers for well stablished, fast and long time tested numerical packages.
- Lots of high level utility libraries for scientific computing: plotting, data analisys, parallelization, ...

Why Python? Batteries Included...

Why Python? ...and PyPi

Find, install and publish Python packages with the Python Package Index

Search projects   🔍

Or browse projects

431,155 projects	4,119,890 releases	7,451,535 files	663,148 users

python™
Package
Index

The Python Package Index (PyPI) is a repository of software for the Python programming language.

PyPI helps you find and install software developed and shared by the Python community. Learn about installing packages ↗.

Package authors use PyPI to distribute their software. Learn how to package your Python code for PyPI ↗.

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

Python Scientific Computing Environment

Quantum Computing

QuTiP
PyQuil
Qiskit
PennyLane

Statistical Computing

Pandas
statsmodels
Xarray
Seaborn

Signal Processing

SciPy
PyWavelets
python-control

Image Processing

Scikit-image
OpenCV
Mahotas

Graphs and Networks

NetworkX
graph-tool
igraph
PyGSP

Astronomy Processes

AstroPy
SunPy
SpacePy

Cognitive Psychology

PsychoPy

Bioinformatics

BioPython
Scikit-Bio
PyEnsembl
ETE

Bayesian Inference

PyStan
PyMC3
ArviZ
emcee

Mathematical Analysis

SciPy
SymPy
cvxpy
FEniCS

Chemistry

Cantera
MDAnalysis
RDKit

Geoscience

Pangeo
Simpeg
ObsPy
Fatiando a Terra

Geographic Processing

Shapely
GeoPandas
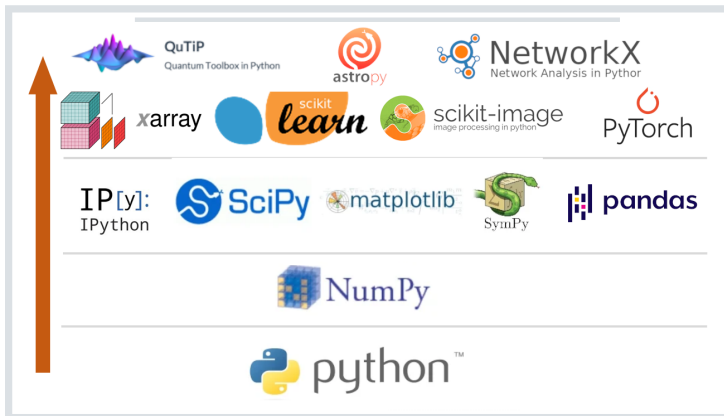Folium

Architecture & Engineering

COMPAS
City Energy Analyst
Sverchok

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

Python Scientific Computing Environment

The SciPy stack consists of Python along with the most commonly used scientific, mathematical, and ML libraries.

These include **NumPy**, **Matplotlib**, the **SciPy** library itself, and **IPython / Jupyter**

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

Python Scientific Computing Environment

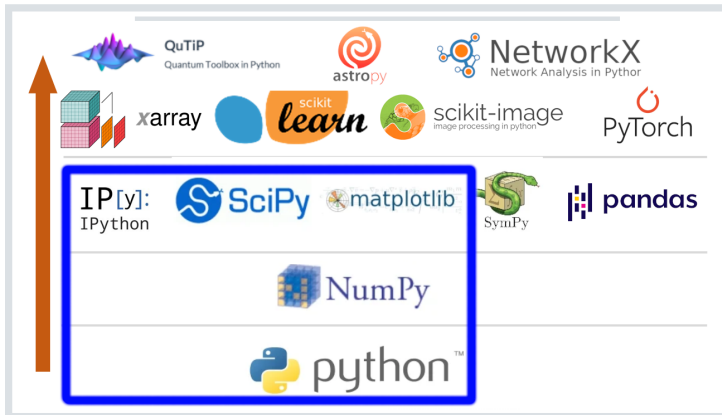The SciPy stack consists of Python along with the most commonly used scientific, mathematical, and ML libraries.

These include **NumPy**, **Matplotlib**, the **SciPy** library itself, and **IPython / Jupyter**

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

NumPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]:
IPython

IPython
Enhanced
Interactive Console

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

NumPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]:
IPython
IPython
Enhanced
Interactive Console

```python
import numpy as np

# Create a numpy array, x
x = np.array( [1.1, 1.3, 1.5] )
y = np.sin(x)

# create a random two dimensional numpy array, A
A = np.random.rand(3,3)

A.transpose()
A.trace()
```

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

SciPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]:
IPython

iPython
Enhanced
Interactive Console

SciPy is a collection of mathematical algorithms and convenience functions built on the Numpy extension of Python.

Much of SciPy is a thin layer of code on top of the C and FORTRAN scientific routines that are freely available at http://www.netlib.org/.

Provides the user with high-level commands and classes for manipulating and visualizing data.

With SciPy an interactive Python session becomes a data-processing and system-prototyping environment rivaling sytems such as MATLAB, IDL, Octave, R-Lab, and SciLab.

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

SciPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]:
IPython

IPython
Enhanced
Interactive Console

SciPy subpackages (some of them)

- constants: Physical and mathematical constants
- fftpack: Fast Fourier Transform routines
- integrate: Integration and ordinary differential equation solvers
- interpolate: Interpolation and smoothing splines
- linalg: Linear algebra
- optimize: Optimization and root-finding
- signal: Signal processing
- special: Special functions
- ...

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

SciPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]: **IPython**
IPython Enhanced
Interactive Console

```python
import numpy as np
from scipy.special import gamma

x = np.array( [1.1, 2., 3.] )
y = gamma(x)

print (y)
```

```
[ 0.95135077,  1. ,  2. ]
```

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

matplotlib

	NumPy Base N-dimensional array package		SciPy library Fundamental library for scientific computing		Matplotlib Comprehensive 2D Plotting		IPython Enhanced Interactive Console

matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms.

matplotlib can be used in Python scripts, the IPython shell, Jupyter Notebooks (ala MATLAB® or Mathematica®), and several graphical user interface toolkits.

For simple plotting the `pyplot` interface provides a MATLAB-like interface, particularly when combined with the `IPython / Jupyter` environment.

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

SciPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
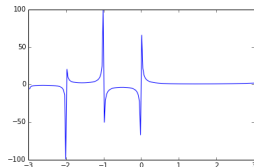library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

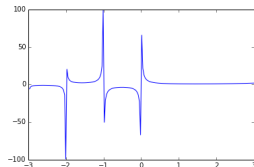IP[y]:
IPython

IPython
Enhanced
Interactive Console

```python
import numpy as np
from scipy.special import gamma
import matplotlib.pyplot as plt

x = np.linspace( start=-3., stop
    =3., num=200 )
y = gamma(x)

plt.plot(x,y)
plt.savefig("gamma.png")
```

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

SciPy

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib

Comprehensive 2D
Plotting

IPython
Enhanced
Interactive Console

```python
import numpy as np
from scipy.special import gamma
import matplotlib.pyplot as plt

x = np.linspace( start=-3., stop
    =3., num=200 )
y = gamma(x)

plt.plot(x,y)
plt.savefig("gamma.png")
```


Just google matplotlib images for examples!

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

IPython / Jupyter

| NumPy Base N-dimensional array package | SciPy library Fundamental library for scientific computing | Matplotlib Comprehensive 2D Plotting | IP[y]: IPython | IPython Enhanced Interactive Console |

The **IPython** / **Jupyter Notebook** / **JupyterLab** is an open-source web-based interactive computing system that enables users to create and share documents that contain live code, LATEXequations, visualizations and explanatory text.

These documents contain a full record of a computation and its results and can be shared on email, Dropbox, version control systems (like git/GitHub) or with the Jupyter online notebook viewer nbviewer.jupyter.org.

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

IPython / Jupyter

NumPy
Base
N-dimensional
array package

SciPy library
Fundamental
library for scientific
computing

Matplotlib
Comprehensive 2D
Plotting

IP[y]:
IPython

IPython
Enhanced
Interactive Console

Ipython / Jupyter notebooks allows us to:

- **Edit code in the browser**, with syntax highlighting, indentation, and tab completion/introspection.
- **Run code in the browser**, with results attached to the code generating them.
- See the results with **rich media representation** as HTML, LaTeX, PDF, PNG, etc.
- Embed **interactive user interface controls** and visualization.
- Author **narrative text** using Markdown markup language.
- Build **hierarchical documents** with headings, sections, etc.
- Use **LaTeX** syntax in Markdown, rendered in the browser.

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
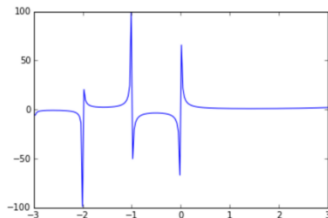IPython / Jupyter

IPython / Jupyter


```
In [45]: %matplotlib inline
         import numpy as np
         from scipy.special import gamma
         import matplotlib.pyplot as plt

         x = np.linspace( start=-3., stop=3., num=200 )
         y = gamma(x)

         plt.plot(x,y)
```
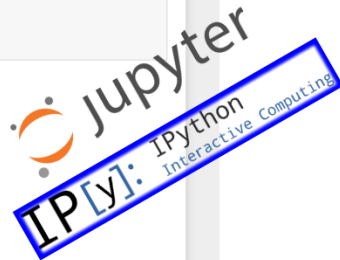Out[45]: [<matplotlib.lines.Line2D at 0x7fc4ac4660b8>]

```
In [ ]: |
```

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

IPython / Jupyter

Python
Python Scientific Computing Environment

NumPy
SciPy
matplotlib
IPython / Jupyter

Python Scientific Computing Environment

Let's play with it!