# Laboratorio di Calcolo

Academic Year 2025 – 2026

Università degli Studi di Roma “Tor Vergata”

# Presentation of the Course

Università di Roma Tor Vergata

- Lecturer
  - Hendrik Speleers
  - speleers@mat.uniroma2.it
- When
  - First semester
  - Thursday 10:00 – 13:00
- Where
  - PC Lab 17

# Presentation of the Course

Università di Roma Tor Vergata

- Content
  - Introduction to the Python scientific ecosystem
- Textbooks
  - “Scientific Python Lectures”, by G. Varoquaux, E. Gouillart, O. Vahtras, et al.
  - “From Python to Numpy”, by N.P. Rougier
- Online material

  https://www.mat.uniroma2.it/~speleers/teaching/labcalc/labcalc.html

# Presentation of the Course

Università di Roma Tor Vergata

- Scientific Computing
  - Solving scientific problems with computers
  - Mathematical models and numerical algorithms
- Exam
  - Project
  - In teams of 2 or 3 students
  - Develop a numerical solution for given differential problem
  - Using the Python scientific ecosystem

# Presentation of the Course

Università di Roma Tor Vergata

- Why the Python language?


# Presentation of the Course

Università di Roma Tor Vergata

- Why the Python language?
  - Batteries included
    - Rich collection of already existing bricks
    - Don‘t reinvent the wheel!
  - Easy to learn + easy communication
    - Python is an interpreted language
    - Scientists don‘t need to be expert programmers!
  - Efficient code
    - Python numerical modules are computationally efficient
    - Python aims for fast development time + fast execution time


# Presentation of the Course

Università di Roma Tor Vergata

- Python scientific ecosystem
  - Python base language
    - The language: flow control, data types, data collections, ...
    - Standard library: string processing, file management, …
    - Development tools: debugging, document generation, ...
  - Libraries for scientific computing
    - **NumPy**: creation and manipulation of numerical arrays
    - **SciPy**: high-level numerical routines
    - **Matplotlib**: 2D visualization and basic 3D visualization
    - **SymPy**: computer algebra system for symbolic computation
    - And many, many more...

# Presentation of the Course

Università di Roma Tor Vergata

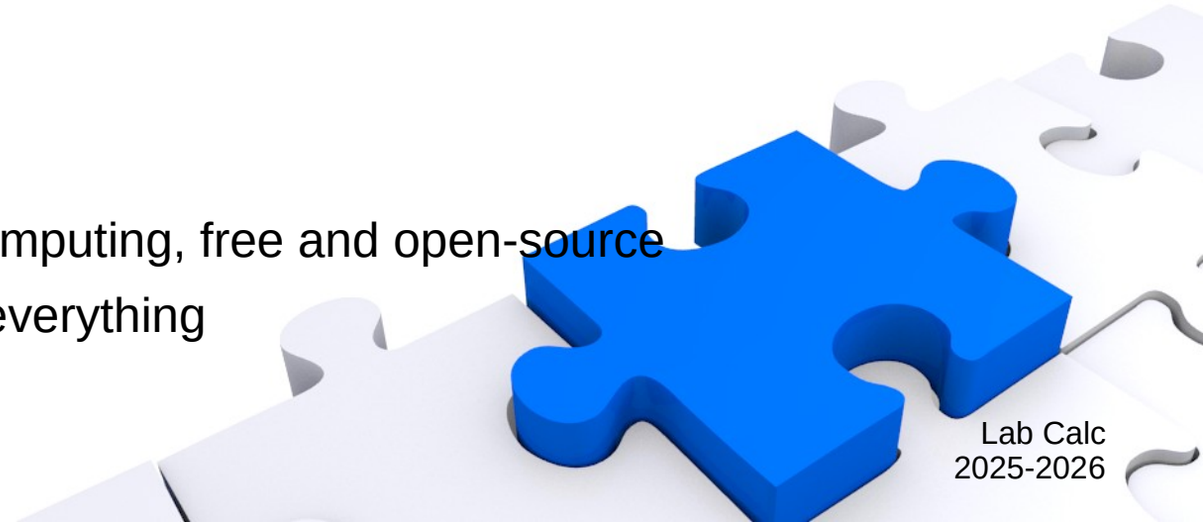
- Python scientific ecosystem
  - Advanced interactive environment
    - **IPython**: advanced Python console
      http://ipython.org/
    - **Jupyter notebook**: browser environment
      https://jupyter.org/
    - **Colaboratory**: Jupyter notebook in the cloud
      https://colab.research.google.com/
  - Fully-featured scientific Python distributions
    - **Anaconda Python**, specialized in data analysis and visualization
      https://www.anaconda.com/
    - Others: **Enthought Canopy**, **WinPython**, ...


ANACONDA®

# Presentation of the Course

Università di Roma Tor Vergata

- Other scientific programming solutions?
  - Compiled languages: C, C++, Fortran, ...
    - Pro: very fast
    - Con: painful use (compilation, verbose syntax, memory management, …)
  - Matlab
    - Pro: rich collection of libraries for scientific computing, pleasant development environment
    - Con: restrictive base languague, not free
  - Python
    - Pro: rich collection of libraries for scientific computing, free and open-source
    - Con: general-purpose but not specialized in everything



# Presentation of the Course

Università di Roma Tor Vergata

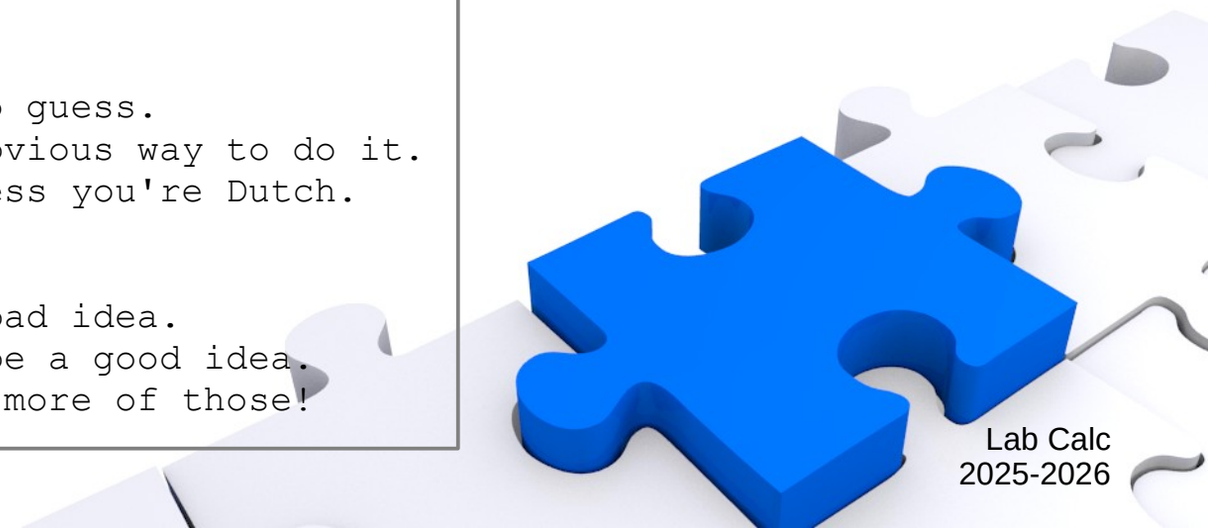
- A little bit of history
  - Python is created by Guido van Rossum
    - Birthplace: CWI, Amsterdam
    - Conceived in the late 1980s
    - First released in 1991
  - Python's design philosophy emphasizes code readability
    - The Zen of Python
  - Multi-paradigm programming language
    - Structured programming, Object-oriented programming, ...
  - Name is derived from “Monty Python”, British comedy group

# Presentation of the Course

Università di Roma Tor Vergata

- The Zen of Python

```
Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
```



# Presentation of the Course

Università di Roma Tor Vergata

- Python2 or Python3 ?
  - Python 2.x is legacy (since 2000)
    - No more new releases (no language improvements)
  - Python 3.x is the present and future of the language (since 2008)
    - Not everything might be fully supported yet (third party software, ...)
    - Not backwards compatibility with 2.x (be careful!)
  - Fully ported
    - NumPy, SciPy, ... for scientific computing
    - IPython/Jupyter for interactive computing
    - ...

# Presentation of the Course

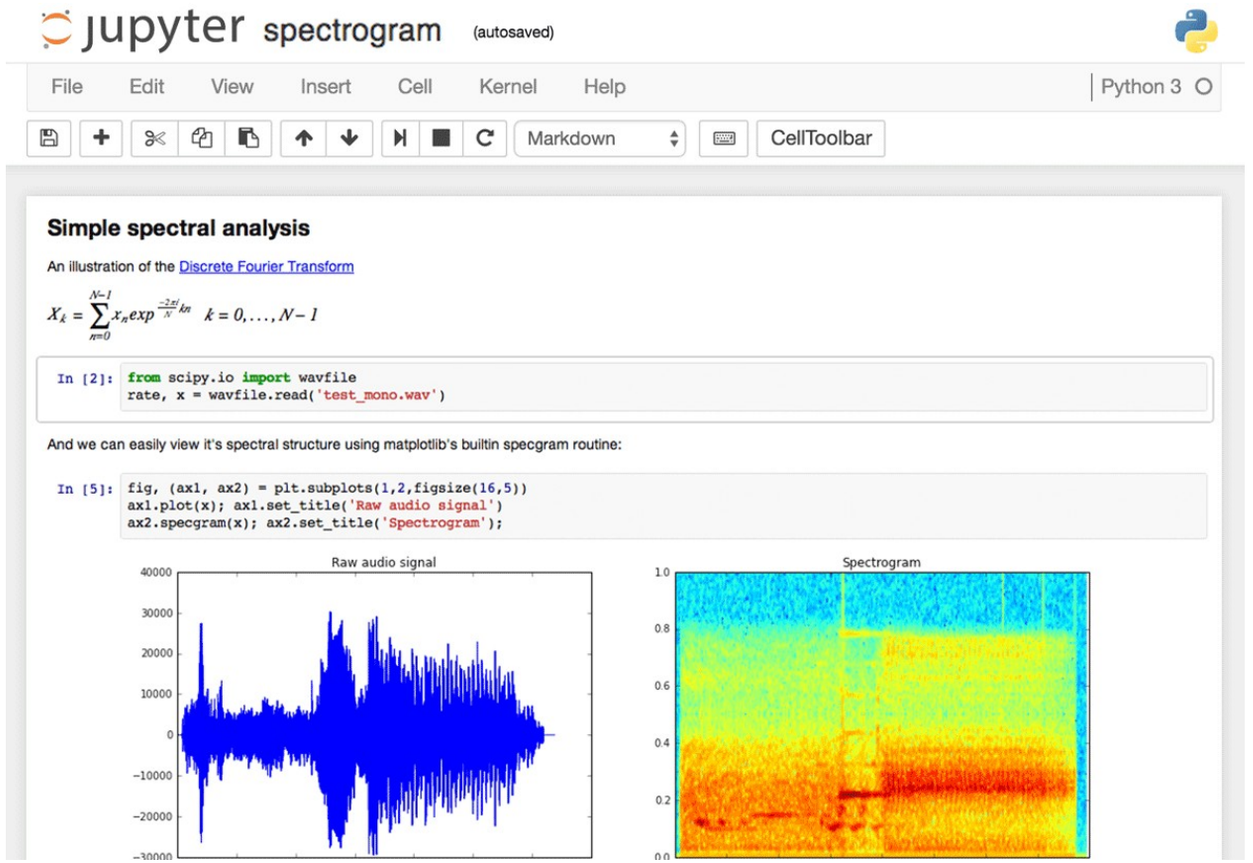
Università di Roma Tor Vergata

- The IPython/Jupyter notebook
  - Jupyter (Julia, Python and R) is a spin-off from IPython
  - Web-based interactive computational environment
  - A notebook document can contain
    - live code (e.g. Python code)
    - visualizations
    - explanatory text (in markup syntax)
    - supports HTML and LaTeX
  - Try it out live!
    - https://jupyter.org/try


# Presentation of the Course

Università di Roma Tor Vergata

- The IPython/Jupyter notebook




# Presentation of the Course

Università di Roma Tor Vergata

- Python example

  Say “Hello, world!” in Python


  - Start IPython or Jupyter
  - Write:

```
In [1]: print('Hello, world!')
Hello, world!
```

  - Note: To execute code in notebook, press “shift enter”