Get started with scientific Python

Potsdam (February 7-9, 2018)

Benoît Bovy - GFZ, section 5.5

Small survey

First, let's have a look at the results of the survey...

The Scientific Python learning curve

The Scientific Python learning curve

The Python language is easy to learn.

The Scientific Python learning curve

The Python language is easy to learn.

But the Python scientific "ecosystem" is very large and rich.

The Scientific Python learning curve

The Python language is easy to learn.

But the Python scientific "ecosystem" is very large and rich.

This is challenging for newcomers.

The Scientific Python learning curve

The Python language is easy to learn.

But the Python scientific "ecosystem" is very large and rich.

This is challenging for newcomers.

The Python scientific ecosystem provides many high-level, easy-to-use tools.

The Scientific Python learning curve

The Python language is easy to learn.

But the Python scientific "ecosystem" is very large and rich.

This is challenging for newcomers.

The Python scientific ecosystem provides many high-level, easy-to-use tools.

But harnessing the potential of these powerful tools often requires good understanding of some abstract concepts.

The Scientific Python learning curve

The Python language is easy to learn.

But the Python scientific "ecosystem" is very large and rich.

This is challenging for newcomers.

The Python scientific ecosystem provides many high-level, easy-to-use tools.

But harnessing the potential of these powerful tools often requires good understanding of some abstract concepts.

This may be also challenging for new users.

Goals of this short course

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Hopefully, you will get answers to these questions

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Hopefully, you will get answers to these questions

- Where to start?

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Hopefully, you will get answers to these questions

- Where to start?
- Where can I find the right tool(s) for my problem?

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Hopefully, you will get answers to these questions

- Where to start?
- Where can I find the right tool(s) for my problem?
- Where can I look for further help and information?

Goals of this short course

Introduce a few "pythonic" concepts and idioms.

Give a broad (though rather shallow) overview of the scientific ecosystem.

Give some tricks, advices and good practices to efficiently write efficient code.

Hopefully, you will get answers to these questions

- Where to start?
- Where can I find the right tool(s) for my problem?
- Where can I look for further help and information?
- Isn't Python too slow?

Outline - Topics

Outline - Topics

Session 1: introduction to Python

- Generalities, history...
- Basic syntax and data structures
- A brief tour of the CPython standard library

Outline - Topics

Session 1: introduction to Python

- Generalities, history...
- Basic syntax and data structures
- A brief tour of the CPython standard library

Session 2: the Python scientific ecosystem

- General overview
- Introduction to core libraries (e.g., Numpy, Matplotlib, Scipy, Pandas)
- How to profile / accelerate / parallelize numerical code in Python?
- A brief tour of the modern scientific/data stack

Outline - Topics

Session 1: introduction to Python

- Generalities, history...
- Basic syntax and data structures
- A brief tour of the CPython standard library

Session 2: the Python scientific ecosystem

- General overview
- Introduction to core libraries (e.g., Numpy, Matplotlib, Scipy, Pandas)
- How to profile / accelerate / parallelize numerical code in Python?
- A brief tour of the modern scientific/data stack

Session 3: mini-projects

Outline - Topics

Session 4: writing scripts with a command-line interface

Outline - Topics

Session 4: writing scripts with a command-line interface

Session 5: some elements of advanced programming in Python

- Classes, decorators, etc.

Outline - Topics

Session 4: writing scripts with a command-line interface

Session 5: some elements of advanced programming in Python

- Classes, decorators, etc.

Session 6: Maintain, share and publish your code

- Good practices of programming and software development

A brief presentation of

The origin of Python

source: wikipedia

Python is not new at all, it was created more than 25 years ago by Guido van Rossum.

Python (CPython) is a community-based, open-source project managed by the Python Software Foundation.

The language vs. its implementation

The name "Python" refers to the language, which is independent from its implementation (software).

The language vs. its implementation

The name "Python" refers to the language, which is independent from its implementation (software).

Python is actually implemented in different languages.

name	language
CPython	C
Jython	Java
IronPython	C#
PyPy	just-in-time compilation

The language vs. its implementation

The name "Python" refers to the language, which is independent from its implementation (software).

Python is actually implemented in different languages.

name	language
CPython	C
Jython	Java
IronPython	C#
PyPy	just-in-time compilation

CPython is the reference and most used implementation.

The language vs. its implementation

The name "Python" refers to the language, which is independent from its implementation (software).

Python is actually implemented in different languages.

name	language
CPython	C
Jython	Java
IronPython	C#
PyPy	just-in-time compilation

CPython is the reference and most used implementation.

The Python scientific libraries are built almost exclusively on top of CPython.

The language vs. its implementation

The name "Python" refers to the language, which is independent from its implementation (software).

Python is actually implemented in different languages.

name	language
CPython	C
Jython	Java
IronPython	C#
PyPy	just-in-time compilation

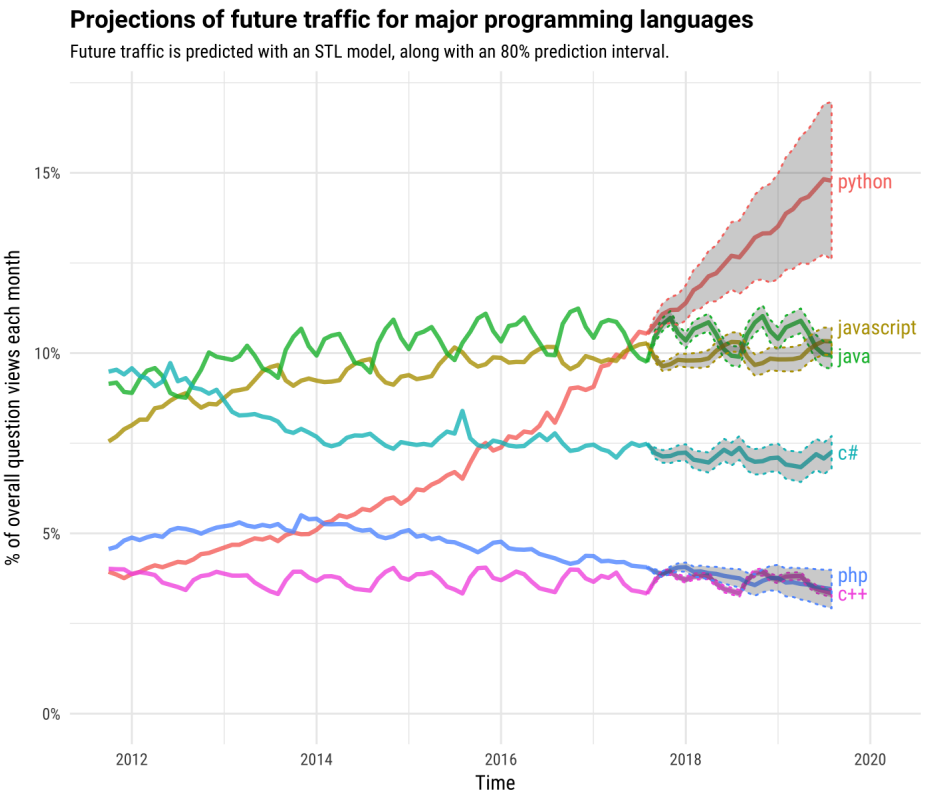
CPython is the reference and most used implementation.

The Python scientific libraries are built almost exclusively on top of CPython.

Therefore, hereafter we use "Python" to refer to both the language and the CPython implementation.

The rise of Python

Python has slowly gained popularity over the years with a recent high increase, especially within the scientific community.



source: stackoverflow

Why is Python popular?

Scientific work involves a whole range of different computational tasks.

Numerics

Data management

Text processing

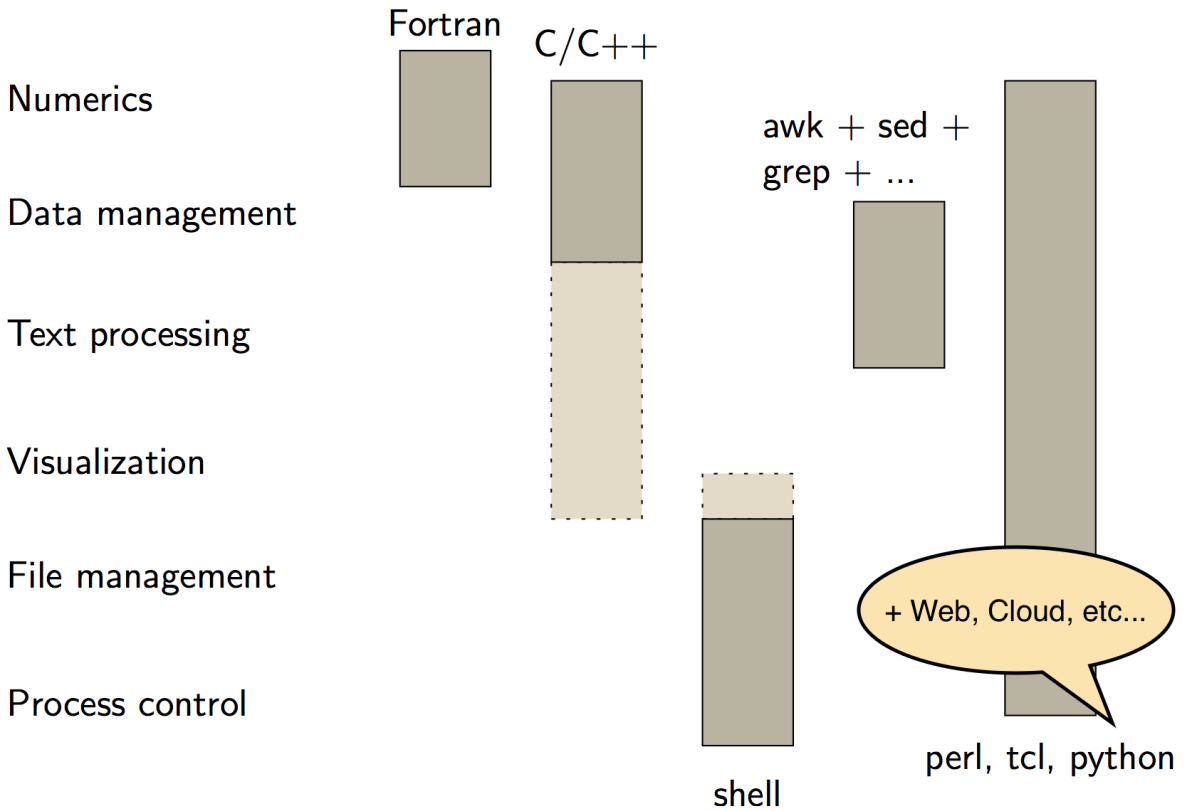
Visualization

File management

Process control

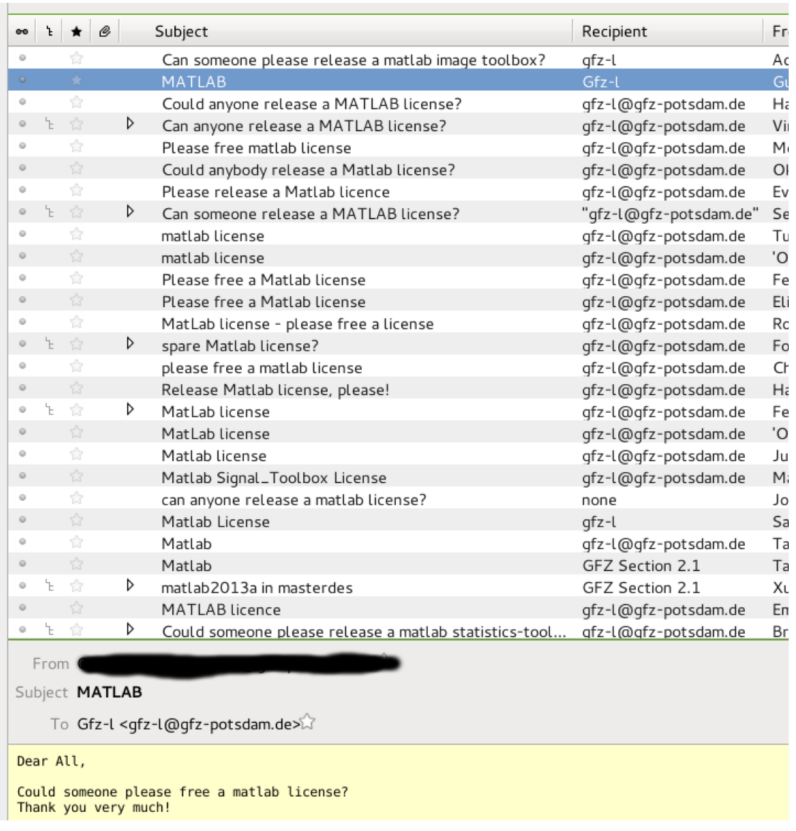
Why is Python popular?

Python is a general purpose language.



source: kinherd.org (Sebastian Heimann)

Why is Python popular?



Subject	Recipient	Fr
Can someone please release a matlab image toolbox?	gfz-l	Ac
MATLAB	Gfz-l	Gu
Could anyone release a MATLAB license?	gfz-l@gfz-potsdam.de	Ha
Can anyone release a MATLAB license?	gfz-l@gfz-potsdam.de	Vi
Please free matlab license	gfz-l@gfz-potsdam.de	M
Could anybody release a Matlab license?	gfz-l@gfz-potsdam.de	Ol
Please release a Matlab licence	gfz-l@gfz-potsdam.de	Ev
Can someone release a MATLAB license?	"gfz-l@gfz-potsdam.de"	Se
matlab license	gfz-l@gfz-potsdam.de	Tu
matlab license	gfz-l@gfz-potsdam.de	'O
Please free a Matlab license	gfz-l@gfz-potsdam.de	Fe
Please free a Matlab license	gfz-l@gfz-potsdam.de	Eli
MatLab license - please free a license	gfz-l@gfz-potsdam.de	Rc
spare Matlab license?	gfz-l@gfz-potsdam.de	Fo
please free a matlab license	gfz-l@gfz-potsdam.de	Ch
Release Matlab license, please!	gfz-l@gfz-potsdam.de	Ha
MatLab license	gfz-l@gfz-potsdam.de	Fe
MatLab license	gfz-l@gfz-potsdam.de	'O
Matlab license	gfz-l@gfz-potsdam.de	Ju
Matlab Signal_Toolbox License	gfz-l@gfz-potsdam.de	Ma
can anyone release a matlab license?	none	Jo
Matlab License	gfz-l	Sa
Matlab	gfz-l@gfz-potsdam.de	Ta
Matlab	GFZ Section 2.1	Ta
matlab2013a in masterdes	GFZ Section 2.1	Xu
MATLAB licence	gfz-l@gfz-potsdam.de	En
Could someone please release a matlab statistics-tool...	gfz-l@gfz-potsdam.de	Br

From [redacted]
Subject **MATLAB**
To Gfz-l <gfz-l@gfz-potsdam.de>

```
Dear All,

Could someone please free a matlab license?
Thank you very much!
```

This doesn't happen with Python.

Why is Python popular?

Clean syntax, readable code

- **Accessible to non computer scientists**

Why is Python popular?

Clean syntax, readable code

- **Accessible to non computer scientists**

High level, interpreted language

- **Faster development (less to maintain, no compilation)**

Why is Python popular?

Clean syntax, readable code

- **Accessible to non computer scientists**

High level, interpreted language

- **Faster development (less to maintain, no compilation)**

Platform independent code, batteries included, easily extensible, robust core scientific librairies + great number of librairies around

- **Reuse instead of reinvent**

Why is Python popular?

Clean syntax, readable code

- **Accessible to non computer scientists**

High level, interpreted language

- **Faster development (less to maintain, no compilation)**

Platform independent code, batteries included, easily extensible, robust core scientific librairies + great number of librairies around

- **Reuse instead of reinvent**

Multi-paradigm (imperative, object-oriented, functional)

- **Suited for writing small scripts as well as large applications**

Python weaknesses

Python weaknesses

Python is slow

- Workaround: call compiled code from within Python
- This does not always matter! "Premature optimization is the root of all evil"

Python weaknesses

Python is slow

- Workaround: call compiled code from within Python
- This does not always matter! "Premature optimization is the root of all evil"

Python 2 vs. Python 3

- No forward nor backward compatibility
- Python 3 has been released a while ago, but then has been slowly adopted
- Unless you have no choice, use Python 3!

Python weaknesses

Python is slow

- Workaround: call compiled code from within Python
- This does not always matter! "Premature optimization is the root of all evil"

Python 2 vs. Python 3

- No forward nor backward compatibility
- Python 3 has been released a while ago, but then has been slowly adopted
- Unless you have no choice, use Python 3!

The ecosystem is rapidly and constantly evolving

- Continuous maintenance work

How to install Python?

How to install Python?

Python is already installed on many platforms (linux distros, macos...) 😀

How to install Python?

Python is already installed on many platforms (linux distros, macos...) 😀

But we'll also need to install Python scientific libraries 😮

- Possibly including non-Python dependencies
- May not be compatible with the system version

How to install Python?

Python is already installed on many platforms (linux distros, macos...) 😀

But we'll also need to install Python scientific libraries 😮

- Possibly including non-Python dependencies
- May not be compatible with the system version

There are many ways to install Python (CPython) + libraries 🤔

- Plateform-specific package managers (apt, homebrew)
- Cross-platform package managers (conda, pip)
- Python distributions (Enthought, PythonXY)
- Compiling the sources (...really?)

How to install Python?

Python is already installed on many platforms (linux distros, macos...) 😀

But we'll also need to install Python scientific libraries 😮

- Possibly including non-Python dependencies
- May not be compatible with the system version

There are many ways to install Python (CPython) + libraries 🤔

- Plateform-specific package managers (apt, homebrew)
- Cross-platform package managers (conda, pip)
- Python distributions (Enthought, PythonXY)
- Compiling the sources (...really?)

My advice 😉

- use conda!
- (and pip when a package cannot be installed using conda)

Conda

How to install conda?

- Download and install Anaconda (Full distribution)
- Or download and install Miniconda (Minimal installation)

Conda

How to install conda?

- Download and install Anaconda (Full distribution)
- Or download and install Miniconda (Minimal installation)

What are the advantages of using conda?

- Cross-platform
- A lot of packages available for scientific development
- Not only for Python
- Isolated envrionments!
  - Different versions of Python/libraries may coexist on the same system
- Works with pip

Conda

How to install conda?

- Download and install Anaconda (Full distribution)
- Or download and install Miniconda (Minimal installation)

What are the advantages of using conda?

- Cross-platform
- A lot of packages available for scientific development
- Not only for Python
- Isolated envrionments!
  - Different versions of Python/libraries may coexist on the same system
- Works with pip

Useful ressources:

- Conda documentation (including tutorials)
- For geeks: read this blog post

How to execute Python code?

How to execute Python code?

- Python interactive console

```
$ python
Python 3.5.2 |Continuum Analytics, Inc.| (default, Jul  2 2016, 17:52:12)
[GCC 4.2.1 Compatible Apple LLVM 4.2 (clang-425.0.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

How to execute Python code?

- Python interactive console

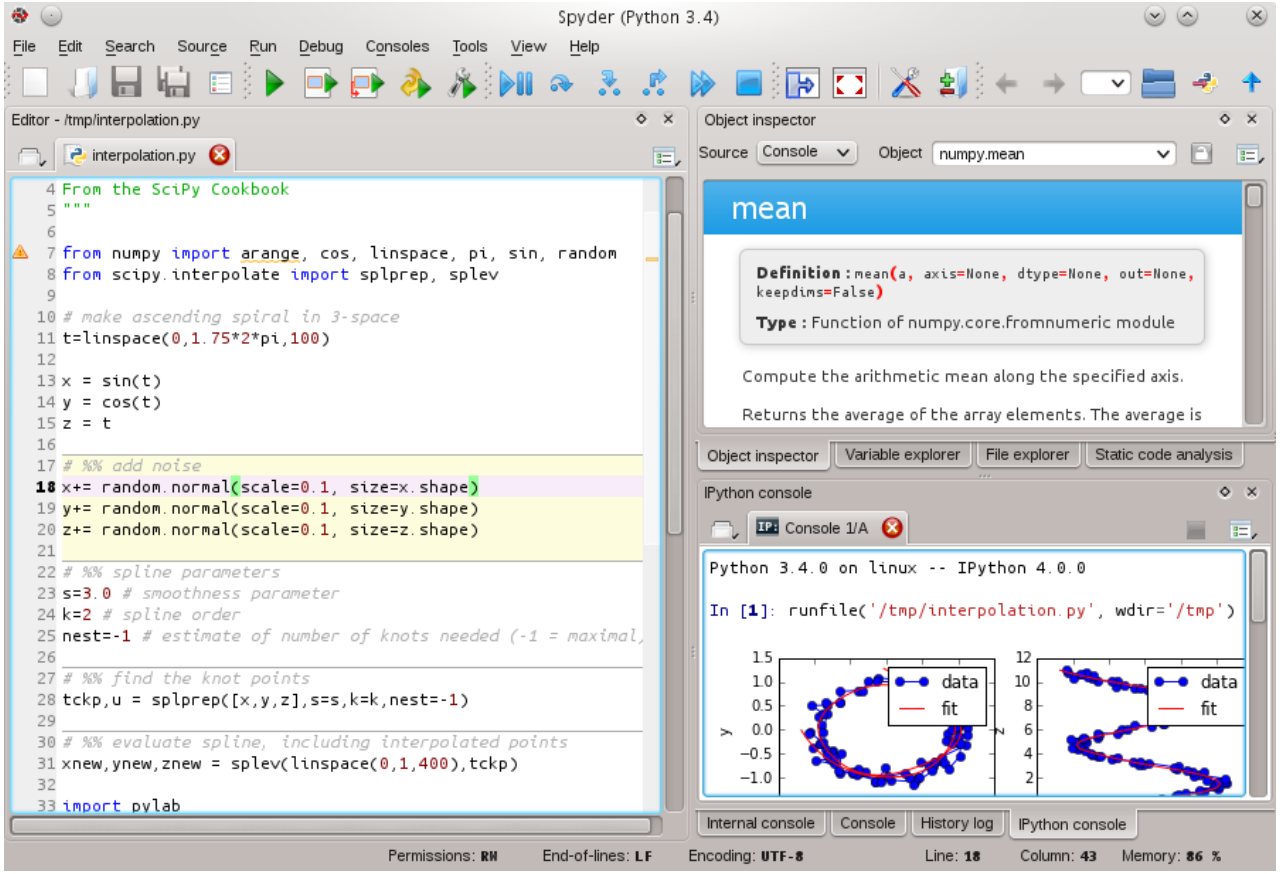
```
$ python
Python 3.5.2 |Continuum Analytics, Inc.| (default, Jul  2 2016, 17:52:12)
[GCC 4.2.1 Compatible Apple LLVM 4.2 (clang-425.0.28)] on darwin
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

- Execute Python scripts (files with *.py* extension)

```
$ python myscript.py
```

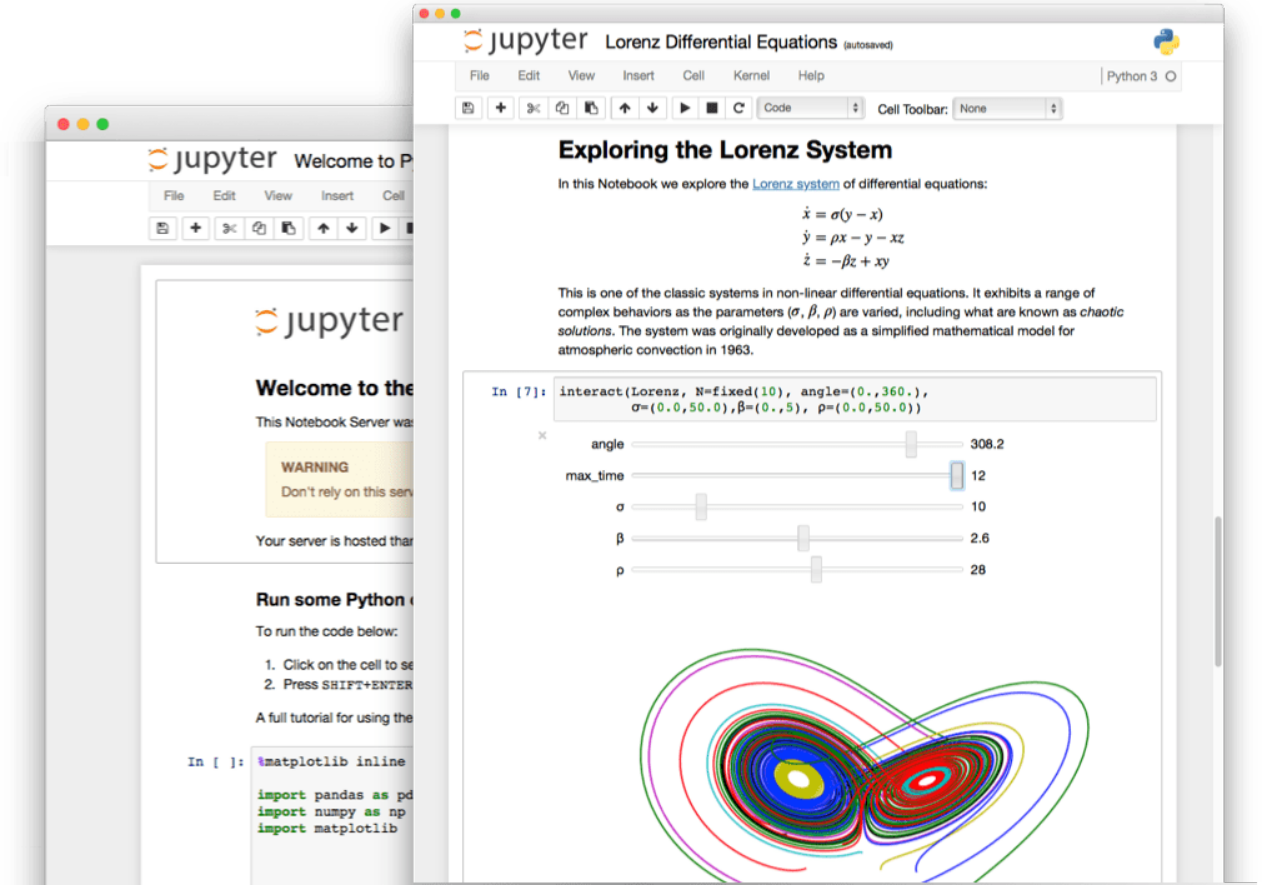
How to execute Python code?

- Spyder (similar to Matlab)



How to execute Python code?

- Jupyter Notebook

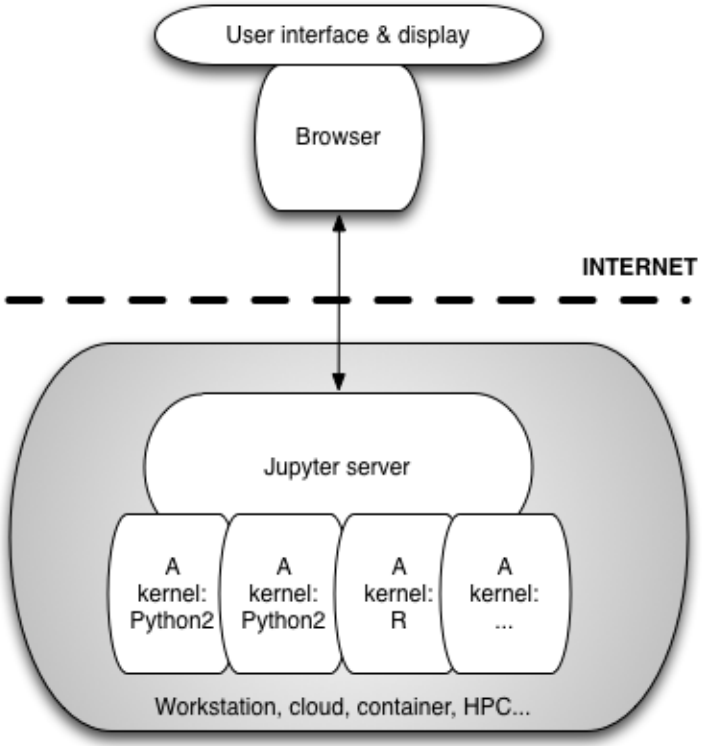


How Jupyter works?

It decouple the interface and the code exectution

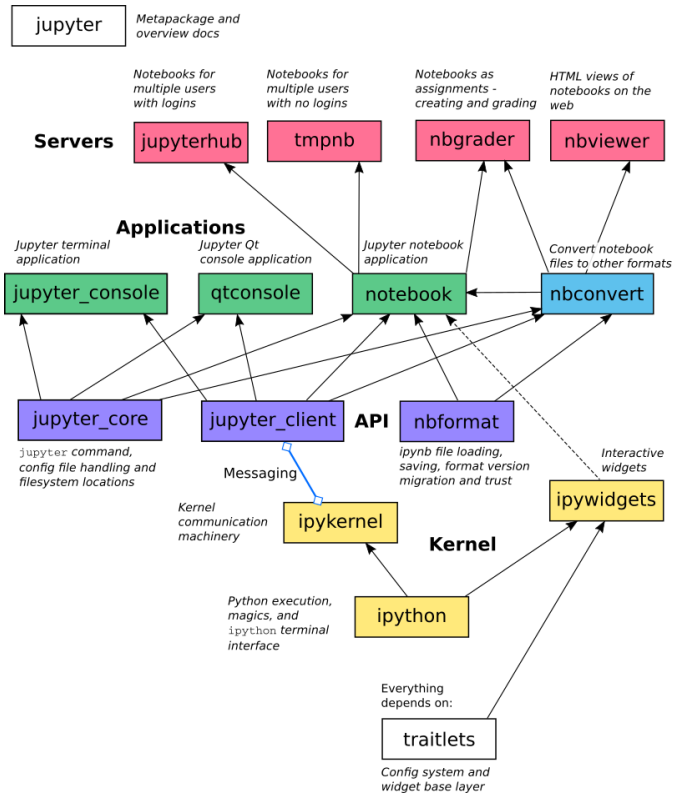
Code can be executed remotely!

Initially for Python (IPython), but language agnostic (R, Julia, Octave... even C++ or Fortran)



source: Oslo repeatability

The Jupyter project: beyond the notebook



source: Jupyter documentation