

# FAIR\_bioinfo for bioinformaticians

## Introduction to the tools of reproducibility in bioinformatics

C. Hernandez<sup>1</sup>   T. Denecker<sup>1</sup>   J.Sellier<sup>2</sup>   C. Toffano-Nioche<sup>1</sup>

<sup>1</sup>Institute for Integrative Biology of the Cell (I2BC)  
UMR 9198, Université Paris-Sud, CNRS, CEA  
91190 - Gif-sur-Yvette, France

<sup>2</sup>Institut de Génétique et de Biologie Moléculaire et Cellulaire (IGBMC)  
CNRS UMR 7104 - Inserm U 1258  
67404 - Illkirch cedex, France

Sept. 2020



# Literate programming

# Introduction

What is literate programming ?

Let us change our traditional attitude to the construction of programs:  
Instead of imagining that our main task is to instruct a computer what to do, let us concentrate rather on explaining to humans what we want the computer to do.

— Donald E. Knuth, Literate Programming, 1984

# Introduction

What is literate programming ?

## Definition

"Literate programming is a programming paradigm introduced by Donald Knuth in which a computer program is given an explanation of its logic in a natural language, such as English, interspersed with snippets of macros and traditional source code, from which compilable source code can be generated." Donald Knuth, 1984.

Wikipedia, 18/08/2020

[https://en.wikipedia.org/wiki/Literate\\_programming#Workflow](https://en.wikipedia.org/wiki/Literate_programming#Workflow)

# Introduction

What does it look like ?

The image displays three overlapping Jupyter Notebook windows. The top window, titled "Lorenz Differential Equations (autosaved)", shows a notebook with the following content:

- Exploring the Lorenz System**
- In this Notebook we explore the [Lorenz system](#) of differential equations:
- $$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= \rho x - y - xz \\ \dot{z} &= -\beta z + xy\end{aligned}$$
- This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters  $(\sigma, \beta, \rho)$  are varied, including what are known as chaotic solutions. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.
- A code cell with the following Python code:

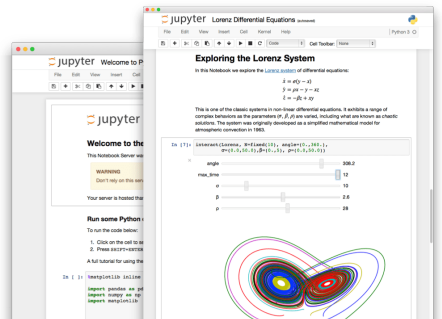
```
In [7]: interact(Lorenz, N=Fixed(10), angle=(0.,360.),
               sigma=(0.0,50.0), beta=(0.,5), rho=(0.0,50.0))
```
- Below the code cell are five interactive sliders for the parameters: angle (0 to 308.2), max\_time (0 to 12),  $\sigma$  (0 to 10),  $\beta$  (0 to 2.6), and  $\rho$  (0 to 28).
- At the bottom is a plot showing the Lorenz attractor, a complex, chaotic trajectory in 3D space, rendered with multiple colored lines.

The middle window shows the "Welcome to the Jupyter Notebook Server" page with a warning: "WARNING: Don't rely on this server. Your server is hosted that..."

The bottom window shows the "Run some Python" page with instructions on how to run code and a code cell with the following Python code:

```
In [ ]: %matplotlib inline
import pandas as pd
import numpy as np
import matplotlib
```

# Introduction



Interactive programming interface allowing to combine both natural and computer languages.

In one file:

- Explanations
- Code
- Results
- Graphs and plots

# Introduction

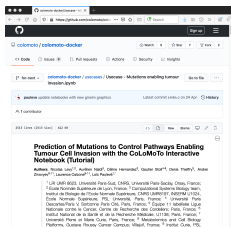
Why using literate programming frameworks ?

Use cases:

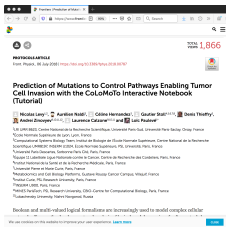
- Day to day analyses
- Analysis reports
- Writing scientific articles

# Example of an article entirely written using a notebook

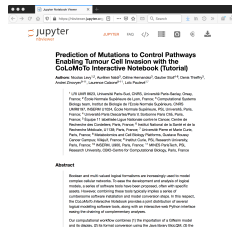
File (on a repository)



Published article



Executable file





# Literate programming

This session :

- Markdown
- Rmarkdown / RStudio
- Jupyter

# Markup and markdown

## Definition

A markup language uses tags to define elements within a document.

Three different types and usage :

- Presentational (used by traditional word-processing systems)
  - ▶ Markup is invisible
- Procedural, provides instructions to process the text (e.g. TeX, PostScript)
  - ▶ Markup is visible and can be directly manipulated by the author.
- Descriptive, to label documents parts (e.g. LaTeX, HTML, XML...)
  - ▶ Emphasizes the document structure.

# Markdown language

Markdown is a Lightweight markup language.

Designed to be :

- easy to write using any generic text editor (plain-text-formatting syntax)
- easy to read in its raw form

# Markdown language

You've probably see it already on GitHub (README), Wikipedia...

```
# Heading
```

```
## Sub-heading
```

```
### Another deeper heading
```

```
A [link](http://example.com).
```

```
Text attributes _italic_, *italic*, **bold**, `monospace`.
```

```
Bullet list:
```

- \* apples
- \* oranges
- \* pears

Github guide :

[urlhttps://guides.github.com/features/mastering-markdown/](https://guides.github.com/features/mastering-markdown/)



# Literate programming

But how is this useful for literate programming?

When you want to weave both code (to be interpreted) and formatting information, you precisely need a lightweight language for the formatting part.

# The challengers

No need to hide, there are currently two main frameworks used in bioinformatics:

RMarkdown and Jupyter

# RMarkdown

# RMarkdown

At the beginning, there was nothing.



At the beginning, there was nothing.

Then came Sweave.

Leisch, Friedrich (2002). "Sweave, Part I: Mixing R and LaTeX: A short introduction to the Sweave file format and corresponding R functions"

At the beginning, there was nothing.

Then came Sweave.

Leisch, Friedrich (2002). "Sweave, Part I: Mixing R and LaTeX: A short introduction to the Sweave file format and corresponding R functions"

And people saw that the path would be long...

knitr (2011)



"The knitr package was designed to be a transparent engine for dynamic report generation with R, solve some long-standing problems in Sweave, and combine features in other add-on packages into one package"

<https://yihui.org/knitr/>

# RMarkdown

## RMarkdown



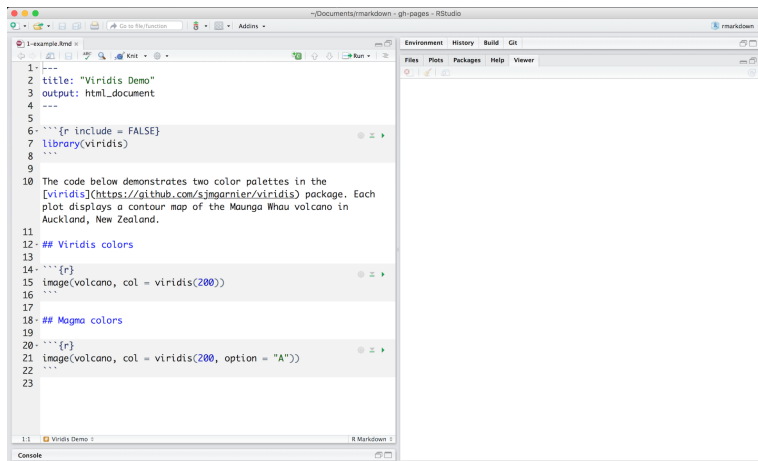
"When you run render, R Markdown feeds the .Rmd file to knitr, which executes all of the code chunks and creates a new markdown (.md) document which includes the code and its output.

The markdown file generated by knitr is then processed by pandoc which is responsible for creating the finished format."

<https://rmarkdown.rstudio.com>

# RMarkdown

## RMarkdown



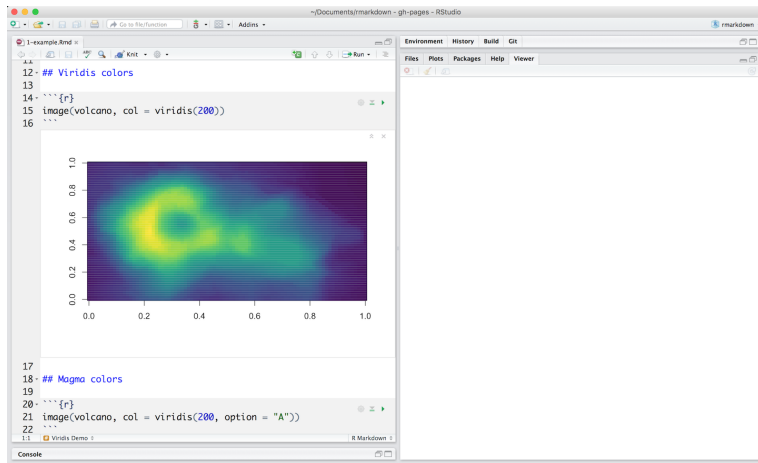
```
1 ---
2 title: "Viridis Demo"
3 output: html_document
4 ---
5
6 ```{r include = FALSE}
7 library(viridis)
8 ```
9
10 The code below demonstrates two color palettes in the
11 [viridis](https://github.com/sjmgarnier/viridis) package. Each
12 plot displays a contour map of the Maunga Whau volcano in
13 Auckland, New Zealand.
14
15 ## Viridis colors
16
17 ```{r}
18 image(volcano, col = viridis(200))
19 ```
20
21 ## Magma colors
22
23 ```{r}
24 image(volcano, col = viridis(200, option = "A"))
25 ```
```

Integrated into RStudio, IDE for R.



# RMarkdown

## R Notebooks



## R Notebooks and more...

Markdown Basics

Output Formats

Notebooks

Slide Presentations

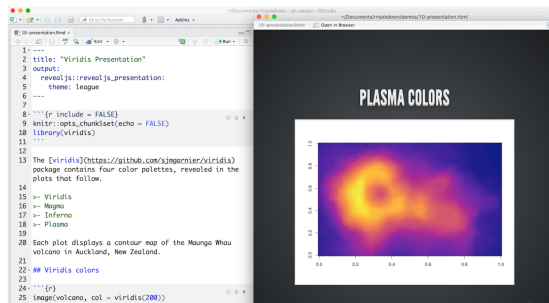
Dashboards

Websites

Interactive Documents

Cheatsheets

file below, which is available [here](#) on RStudio Cloud.



# Jupyter



## A bit of history...

- 2011 : IPython (interactive Python shell) with notebook functionalities
- 2014 : Spin-off project called Project Jupyter
- a non-profit, open-source project maintained by a strong Community
- " Jupyter will always be 100% open-source software, free for all to use and released under the liberal terms of the modified BSD license"
- A reference to the three core programming languages supported by Jupyter (Julia, Python and R)

<https://jupyter.org/>

What can it do?

What can it do?  
Everything (excepted coffee)

But what is it exactly ?

But what is it exactly ?

Web-based interactive computational environment.

But what is it exactly ?

Web-based interactive computational environment.

- Web-based : client/server

But what is it exactly ?

Web-based interactive computational environment.

- Web-based : client/server
- Interactive : notebook system

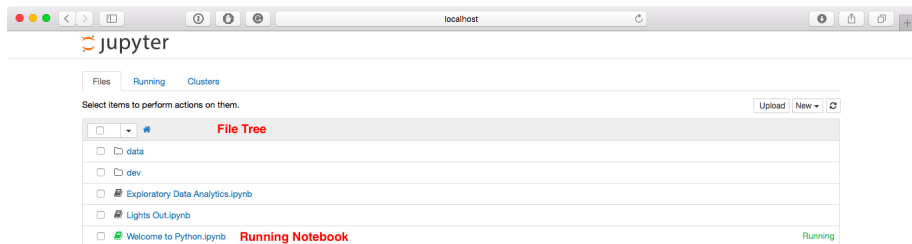
But what is it exactly ?

Web-based interactive computational environment.

- Web-based : client/server
- Interactive : notebook system
- Computational environment : console, many kernels available...



## Dashboard



The screenshot shows the Jupyter Dashboard in a web browser window. The browser's address bar displays 'localhost'. The Jupyter logo is visible in the top left of the dashboard. Below the logo, there are three tabs: 'Files', 'Running', and 'Clusters'. The 'Files' tab is currently selected. Below the tabs, there is a message: 'Select items to perform actions on them.' To the right of this message are buttons for 'Upload', 'New', and a refresh icon. Below this is a 'File Tree' section with a list of files and folders. The list includes 'data', 'dev', 'Exploratory Data Analytics.ipynb', 'Lights Out.ipynb', and 'Welcome to Python.ipynb'. The 'Welcome to Python.ipynb' file is highlighted in green and labeled 'Running Notebook' in red text, with a green 'Running' status indicator to its right.

Files Running Clusters

Select items to perform actions on them. Upload New ↻

**File Tree**

- ☐ data
- ☐ dev
- ☐ Exploratory Data Analytics.ipynb
- ☐ Lights Out.ipynb
- ☒ Welcome to Python.ipynb **Running Notebook** Running

## Notebook editor

jupyter Welcome to Python (unsaved changes) Python 3

File Edit View Insert Cell Kernel Help Menubar

CellToolbar Toolbar Cell Mode Indicator Kernel Indicator

**Welcome to the Temporary Notebook (tmpnb) service!**

This Notebook Server was **launched just for you**. It's a temporary way for you to try out a recent development version of the IPython/Jupyter notebook.

**WARNING**  
Don't rely on this server for anything you want to last - your server will be *deleted after 10 minutes of inactivity*.

Your server is hosted thanks to [Rackspace](#), on their on-demand bare metal servers, [OnMetal](#).

**Run some Python code!**

To run the code below:

1. Click on the cell to select it.
2. Press **SHIFT+ENTER** on your keyboard or press the play button (▶) in the toolbar above.

A full tutorial for using the notebook interface is available [here](#).

```
In [ ]: %matplotlib inline
import pandas as pd
import numpy as np
import matplotlib
```

## Project Jupyter

- A non-profit, open-source project maintained by a strong Community
- Adopted by the biggest in the Cloud industry (Google, Microsoft, Amazon...)
- And financed by the biggest (Google, Microsoft, EU Horizon 2020 program, Alfred P. Sloan Foundation...)

Inside the Python community (snakemake, conda...)

Integration with GitHub since 2015 (renderer)

Nbviewer : a static renderer for Jupyter notebooks



[JUPYTER](#) [FAQ](#)

# nbviewer

**A simple way to share Jupyter Notebooks**

Enter the location of a Jupyter Notebook to have it rendered here:

Go!

`https://nbviewer.jupyter.org/`



## Turn a Git repo into a collection of interactive notebooks

Have a repository full of Jupyter notebooks? With Binder, open those notebooks in an executable environment, making your code immediately reproducible by anyone, anywhere.

New to Binder? Get started with a Zero-to-Binder tutorial in [Julia](#), [Python](#) or [R](#).

Build and launch a repository

GitHub repository name or URL

GitHub ▾

Git branch, tag, or commit

Path to a notebook file (optional)

File ▾

launch

Copy the URL below and share your Binder with others:

🏠

<https://mybinder.org/>

## More to come : Jupyter Lab 1.0

**In Depth: Linear Regression**

Just as naive Bayes (discussed earlier in [Depth: Naive Bayes Classification](#)) is a good starting point for classification tasks, linear regression models are a good starting point for regression tasks. Such models are popular because they can be fit very quickly, and are very interpretable. You are probably familiar with the simplest form of a linear regression model (i.e., fitting a straight line to data) but such models can be extended to model more complicated data behavior.

In this section we will start with a quick intuitive walk-through of the mathematics behind this well-known problem, before seeing how before moving on to see how linear models can be generalized to account for more complicated patterns in data.

We begin by

File Edit View Run Kernel Tabs Settings Help

Simple

We will use

Consider a

Code Mirrors

Code Mirrors

```

{
  "kernel_name": "python3",
  "language": "python",
  "name": "python3",
  "version": 3
},
{
  "file_extension": ".py",
  "mime_type": "text/x-python",
  "name": "python",
  "extension": "python",
  "extension": "python"
},
{
  "name": "python",
  "extension": "python",
  "extension": "python"
},
{
  "name": "python",
  "extension": "python",
  "extension": "python"
}

```

Console

Python 3

Julia 1.0

python notebook

```

[1]: using Plots, GeoM
[2]: plot(1:10, rand(10,1), "scatter", w=50)
[3]:
[4]:
[5]:
[6]:
[7]:
[8]:
[9]:
[10]:
[11]:
[12]:
[13]:
[14]:
[15]:
[16]:
[17]:
[18]:
[19]:
[20]:
[21]:
[22]:
[23]:
[24]:
[25]:
[26]:
[27]:
[28]:
[29]:
[30]:
[31]:
[32]:
[33]:
[34]:
[35]:
[36]:
[37]:
[38]:
[39]:
[40]:
[41]:
[42]:
[43]:
[44]:
[45]:
[46]:
[47]:
[48]:
[49]:
[50]:
[51]:
[52]:
[53]:
[54]:
[55]:
[56]:
[57]:
[58]:
[59]:
[60]:
[61]:
[62]:
[63]:
[64]:
[65]:
[66]:
[67]:
[68]:
[69]:
[70]:
[71]:
[72]:
[73]:
[74]:
[75]:
[76]:
[77]:
[78]:
[79]:
[80]:
[81]:
[82]:
[83]:
[84]:
[85]:
[86]:
[87]:
[88]:
[89]:
[90]:
[91]:
[92]:
[93]:
[94]:
[95]:
[96]:
[97]:
[98]:
[99]:
[100]:

```

R

```

[1]: ggplot(data=iris, aes(Sepal.Length,

```

# Conclusion ?

Who's the best?

# Conclusion ?

Who's the best?

It depends...



# Conclusion ?

Who's the best?

It depends...

- R analyses? Go for RMarkdown/RStudio
- R analyses for a publication ? Consider Jupyter with an R kernel

# Conclusion ?

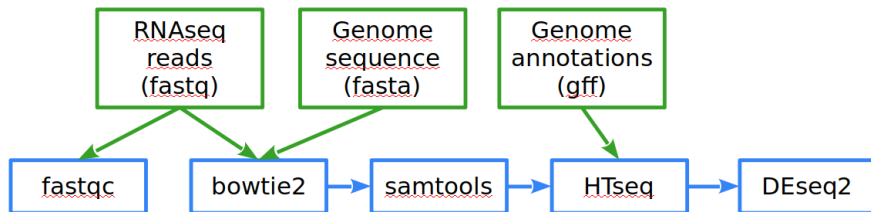
Who's the best?

It depends...

- R analyses? Go for RMarkdown/RStudio
- R analyses for a publication ? Consider Jupyter with an R kernel
- Python analyses ? Why do you even ask...

# Practical session

## Analysis workflow



green=input, blue=tool

**fastqc** control quality of the input reads

**bowtie2** reads mapping on the genome sequence

**samtools** mapped reads selection & formatting

**HTseq** count table of mapped reads on genes (annotations)

**DESeq2** statistical analysis: genes list having differential expression

# Practical session

## Savoir FAIRe

- Markdown
- Learn the structure of an Rmd file
- Turn a script into a notebook
- Extend the notebook with new functionalities
- Bonus: introduction to Jupyter