Reproducibility Crisis, Open Science,... and Computer Science

Arnaud Legrand

MVA, ENS Paris-Saclay
March 2023

What is Science?

Question 1: In less than 5 lines give a definition of "Science"

What is Science?

Question 1: In less than 5 lines give a definition of "Science"

Dictionary of science and technology

1. the study of the physical and natural world and phenomena, especially by using systematic observation and experiment
2. a particular area of study or knowledge of the physical world
3. a systematically organized body of knowledge about a particular subject

New Oxford Dictionary the intellectual and practical activity encompassing the systematic study of the structure and behavior of the physical and natural world through observation and experiment : the world of science and technology.

1. a particular area of this : veterinary science | the agricultural sciences.
2. a systematically organized body of knowledge on a particular subject : the science of criminology.
3. archaic knowledge of any kind.

What is Science?

Question 1: In less than 5 lines give a definition of "Science"

Dictionary of science and technology

1. the study of the physical and natural world and phenomena, especially by using systematic observation and experiment
2. a particular area of study or knowledge of the physical world
3. a systematically organized body of knowledge about a particular subject

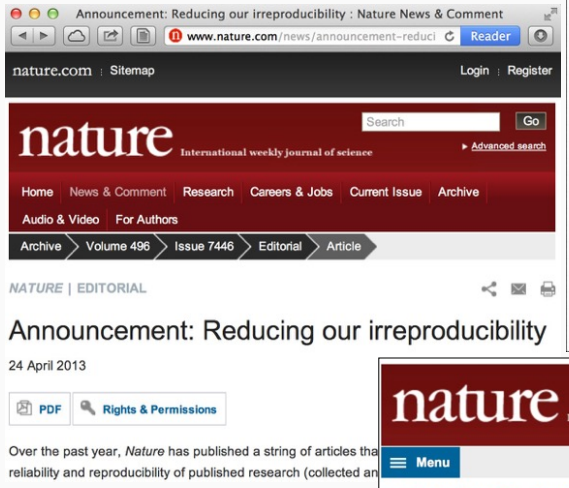
New Oxford Dictionary the intellectual and practical activity encompassing the systematic study of the structure and behavior of the physical and natural world through observation and experiment : the world of science and technology.

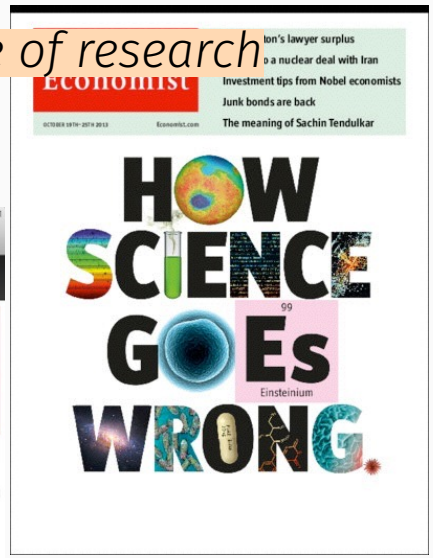
1. a particular area of this : veterinary science | the agricultural sciences.
2. a systematically organized body of knowledge on a particular subject : the science of criminology.
3. archaic knowledge of any kind.

Building Reliable Knowledge

Public Evidence for a Lack of Reproducibility

- J.P. Ioannidis. *Why Most Published Research Findings Are False* PLoS Med. 2005.
- *Lies, Damned Lies, and Medical Science*, The Atlantic. Nov, 2010
- *Reproducibility: A tragedy of errors*, Nature, Feb 2016.
- Steen RG, *Retractions in the scientific literature: is the incidence of research fraud increasing?*, J. Med. Ethics 37, 2011





Courtesy V. Stodden, SC, 2015

Newsworthy Stories about Scientific Misconduct

Dong-Pyou Han Assistant professor, Biomedical sciences, Iowa State University, 2013

Falsified blood results to make it appear as though a vaccine exhibited anti-HIV activity

- Han and his team received ≈ $19 million from NIH
- 1 retracted publication and resignation of university. Sentenced in 2015 to 57 months imprisonment for fabricating and falsifying data in HIV vaccine trials. $7.2 million!

Newsworthy Stories about Scientific Misconduct

Dong-Pyou Han Assistant professor, Biomedical sciences, Iowa State University, 2013

Falsified blood results to make it appear as though a vaccine exhibited anti-HIV activity

- Han and his team received ≈ $19 million from NIH
- 1 retracted publication and resignation of university. Sentenced in 2015 to 57 months imprisonment for fabricating and falsifying data in HIV vaccine trials. $7.2 million!

Diederik Stapel Professor, Social Psychology, Univ. Tilburg, 2011

I failed as a scientist. I adapted research data and fabricated research. Not once, but several times, not for a short period, but over a longer period of time. [..] I am aware of the suffering and sorrow that I caused to my colleagues... I did not withstand the pressure to score, to publish, the pressure to get better in time. I wanted too much, too fast. In a system where there are few checks and balances, where people work alone, I took the wrong turn.

58 retracted publications

Newsworthy Stories about Scientific Misconduct

Dong-Pyou Han Assistant professor, Biomedical sciences, Iowa State University, 2013

Falsified blood results to make it appear as though a vaccine exhibited anti-HIV activity

- Han and his team received ≈ $19 million from NIH
- 1 retracted publication and resignation of university. Sentenced in 2015 to 57 months imprisonment for fabricating and falsifying data in HIV vaccine trials. $7.2 million!

Diederik Stapel Professor, Social Psychology, Univ. Tilburg, 2011

I failed as a scientist. I adapted research data and fabricated research. Not once, but several times, not for a short period, but over a longer period of time. [..] I am aware of the suffering and sorrow that I caused to my colleagues... I did not withstand the pressure to score, to publish, the pressure to get better in time. I wanted too much, too fast. In a system where there are few checks and balances, where people work alone, I took the wrong turn.

58 retracted publications

Brian Wansink Professor, Psychological Nutrition, Cornell, 2016

I gave her a data set of a self-funded, failed study which had null results. I said "This cost us a lot of time and our own money to collect. There's got to be something here we can salvage because it's a cool (rich & unique) data set." I told her what the analyses should be. [..] Every day she came back with puzzling new results, and every day we would scratch our heads, ask "Why," and come up with another way to reanalyze the data with yet another set of plausible hypotheses

17 retracted publications

Scientific Misconduct? What are the consequences ?

Reinhart and Rogoff Professors of Economics at Harvard

> *gross debt [..] exceeding 90 percent of the economy has a significant negative effect on economic growth* – *Growth in a Time of Debt (2010)*
>
> *While using RR's working spreadsheet, we identified coding errors, selective exclusion of available data, and unconventional weighting of summary statistics.* – *2013: Herndon, Ash and Pollin*
>
> *For 3 years, austerity was not presented as an option but as a necessity.* – *2013: Paul_Krugman*

At least, a scientific debate has been possible.

Scientific Misconduct? What are the consequences ?

Reinhart and Rogoff Professors of Economics at Harvard

> *gross debt [..] exceeding 90 percent of the economy has a significant negative effect on economic growth* – *Growth in a Time of Debt (2010)*
>
> *While using RR's working spreadsheet, we identified coding errors, selective exclusion of available data, and unconventional weighting of summary statistics.* – *2013: Herndon, Ash and Pollin*
>
> *For 3 years, austerity was not presented as an option but as a necessity.* – *2013: Paul_Krugman*

At least, a scientific debate has been possible.

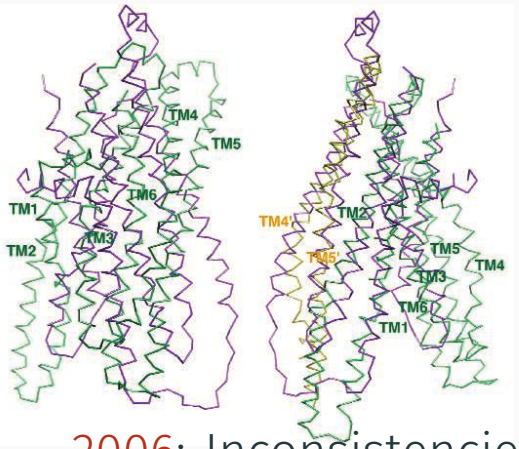
Bad science is **deleterious**

- It is used to backup stupid politics, it affects people's life, ...
- It blurs the frontier between scientists and crooks

Media attention inflates **conspiracy** opinions 

- *Scientific result are worthless.*
- *Scientists can't even agree with each others on economy/climate/vaccine/5G/...*
- *Stop the scientific dictatorship/lobby!*

How computers broke science



Geoffrey Chang (Scripps, UCSD) works on crystalography and studies the structure of cell membrane proteins.

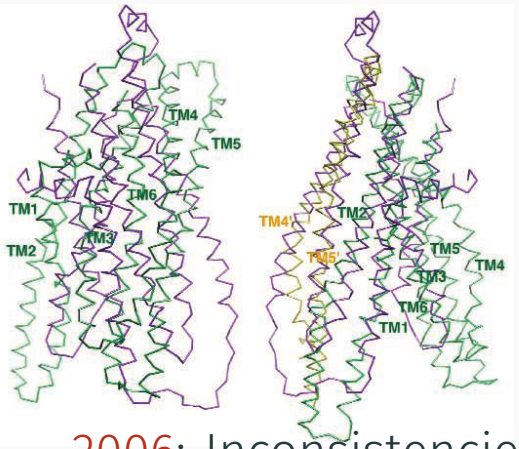
He specialized in structures of multidrug resistant transporter proteins in bacteria: MsbA de Escheria Choli (Science, 2001), Vibrio cholera (Mol. Biology, 2003), Salmonella typhimurium (Science, 2005)

2006: Inconsistencies reveal a programming mistake

> *A homemade data-analysis program had flipped two columns of data, inverting the electron-density map from which his team had derived the protein structure.*

5 retractations that motivate improved software engineering practices in comp. biology

How computers broke science



Geoffrey Chang (Scripps, UCSD) works on crystalography and studies the structure of cell membrane proteins.

He specialized in structures of multidrug resistant transporter proteins in bacteria: MsbA de Escheria Choli (Science, 2001), Vibrio cholera (Mol. Biology, 2003), Salmonella typhimurium (Science, 2005)

2006: Inconsistencies reveal a programming mistake

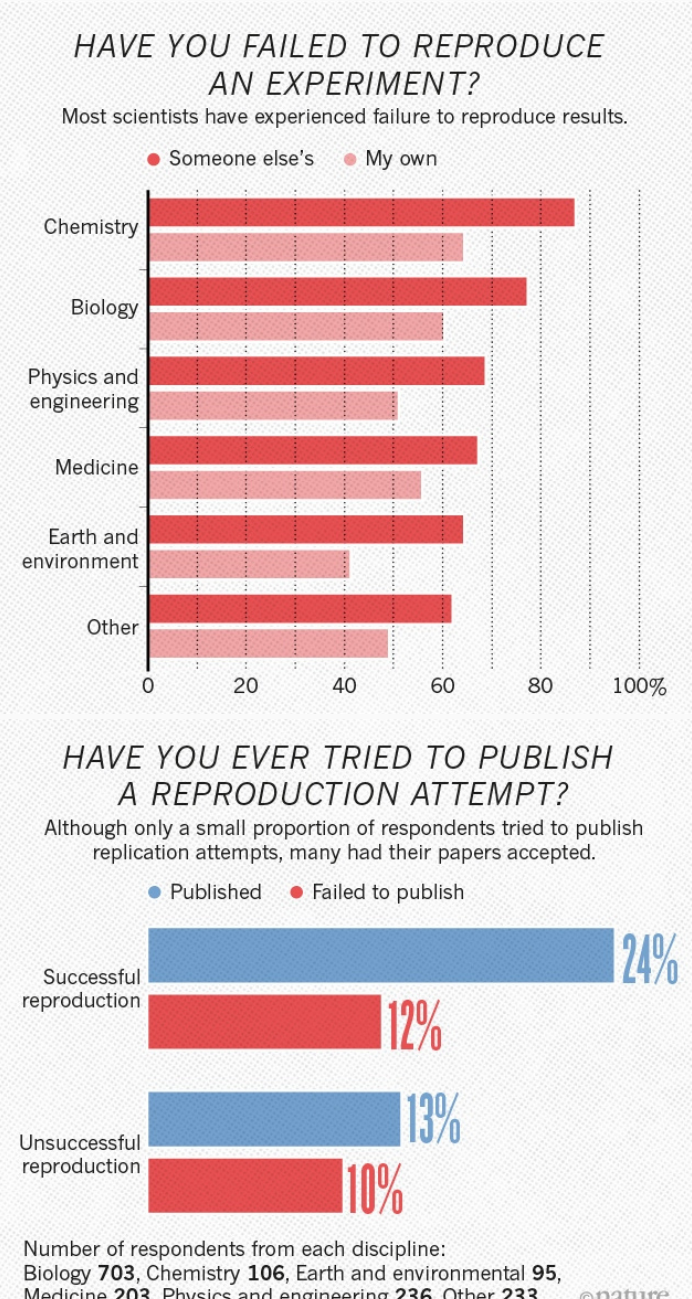
> *A homemade data-analysis program had flipped two columns of data, inverting the electron-density map from which his team had derived the protein structure.*

5 retractations that motivate improved software engineering practices in comp. biology

There is worse!

- The generalized and intensive use of spreadsheets (COVID tracing)
- Relying on black box statistical methods is infinitely easier than understanding them (Learning and Data Analytics frameworks = nuke)
- Numerical errors and software environment unawareness

A Reproducibility Crisis?



1,500 scientists lift the lid on reproducibility, Nature, May 2016

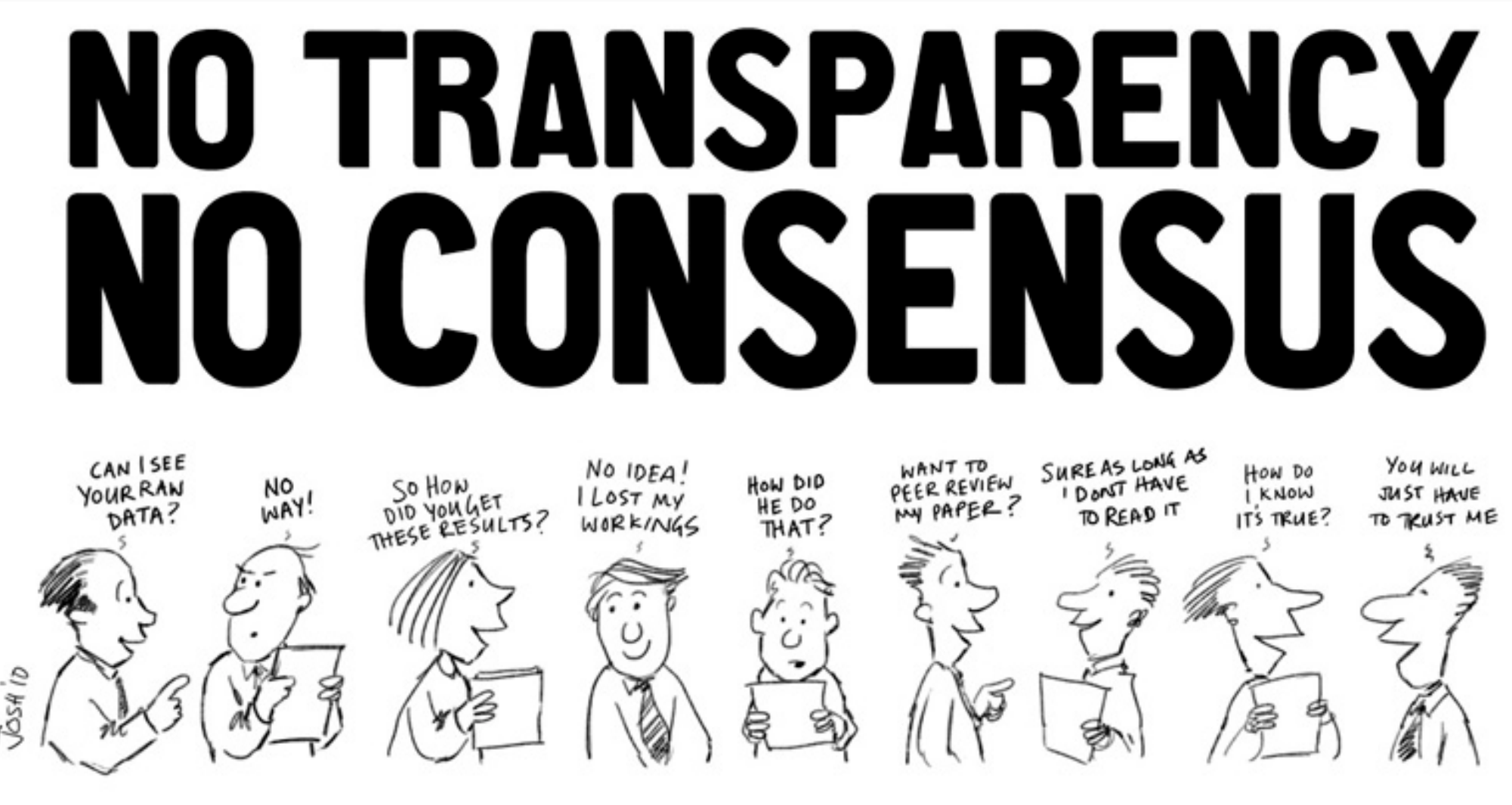
Social causes

- Fraud, conflict of interest (pharmaceutic, ...)
- No incentive to reproduce/check our own work (afap), nor the work of others (big results!), nor to allow others to check (competition)
- Peer review does not scale: 1+ million articles per year!

Methodological or technical causes

- The many biases (apophenia, confirmation, hindsight, experimenter, ...): bad designs
- Selective reporting, weak analysis (statistics, data manipulation mistakes, computational errors)
- Lack of information, code/raw data unavailable

Scientific Consensus



Different Reproducibility Concerns in Modern Science

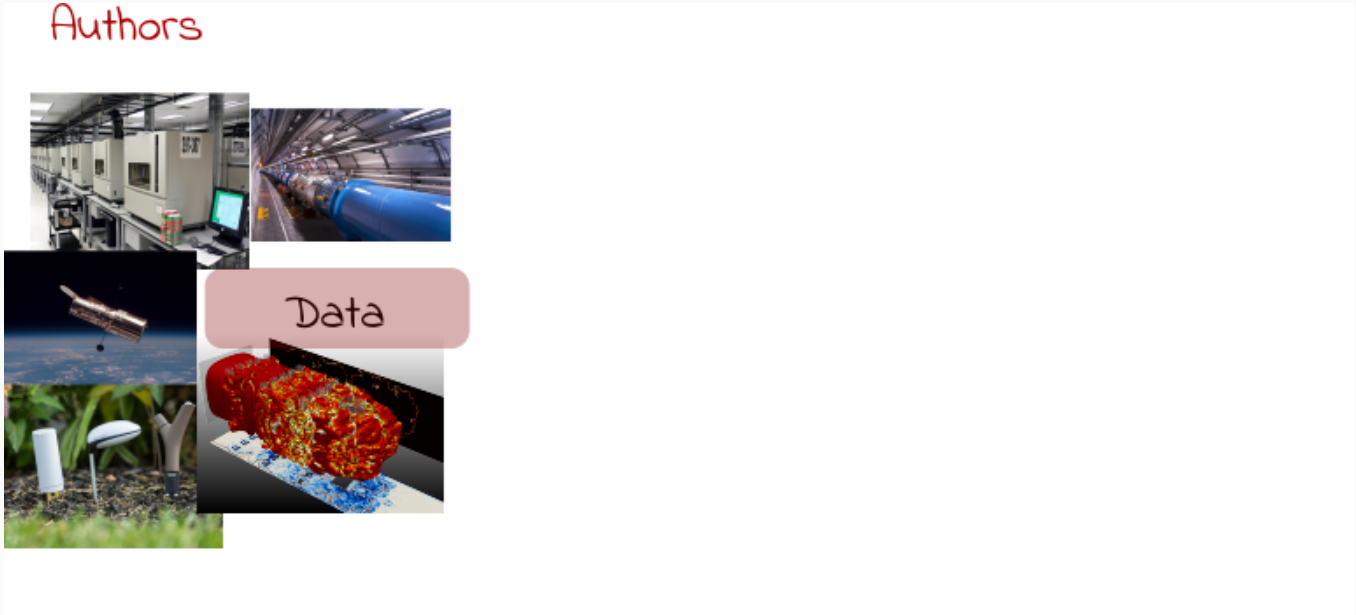
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Different Reproducibility Concerns in Modern Science

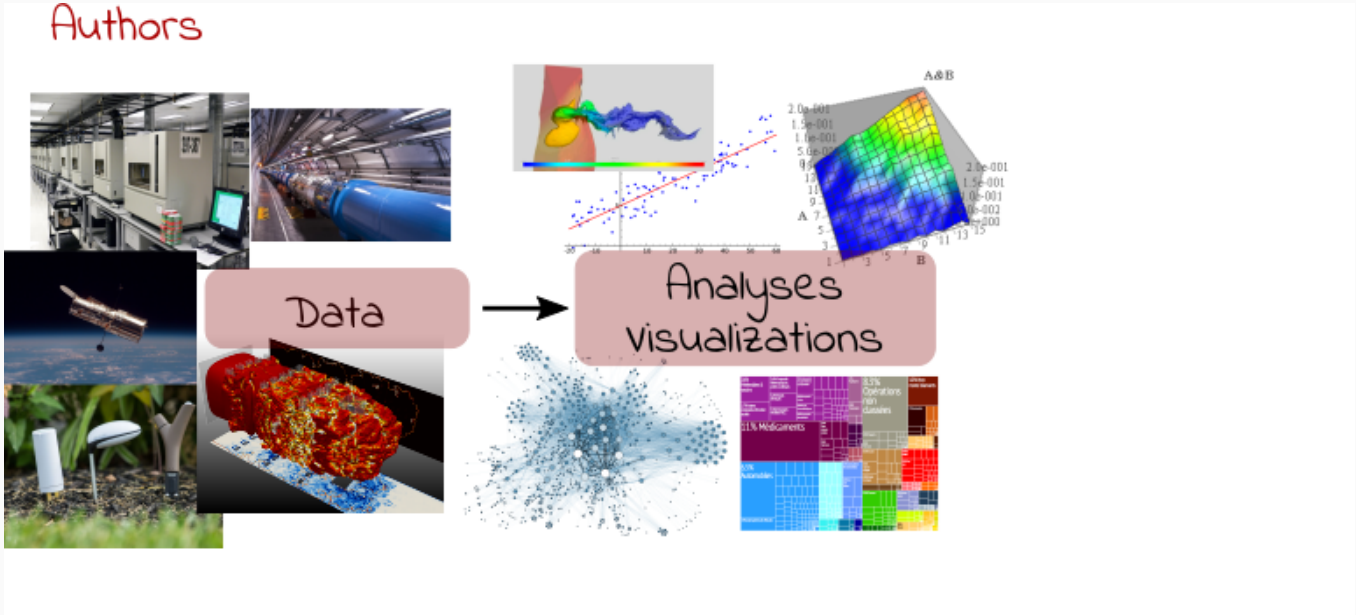
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Different Reproducibility Concerns in Modern Science

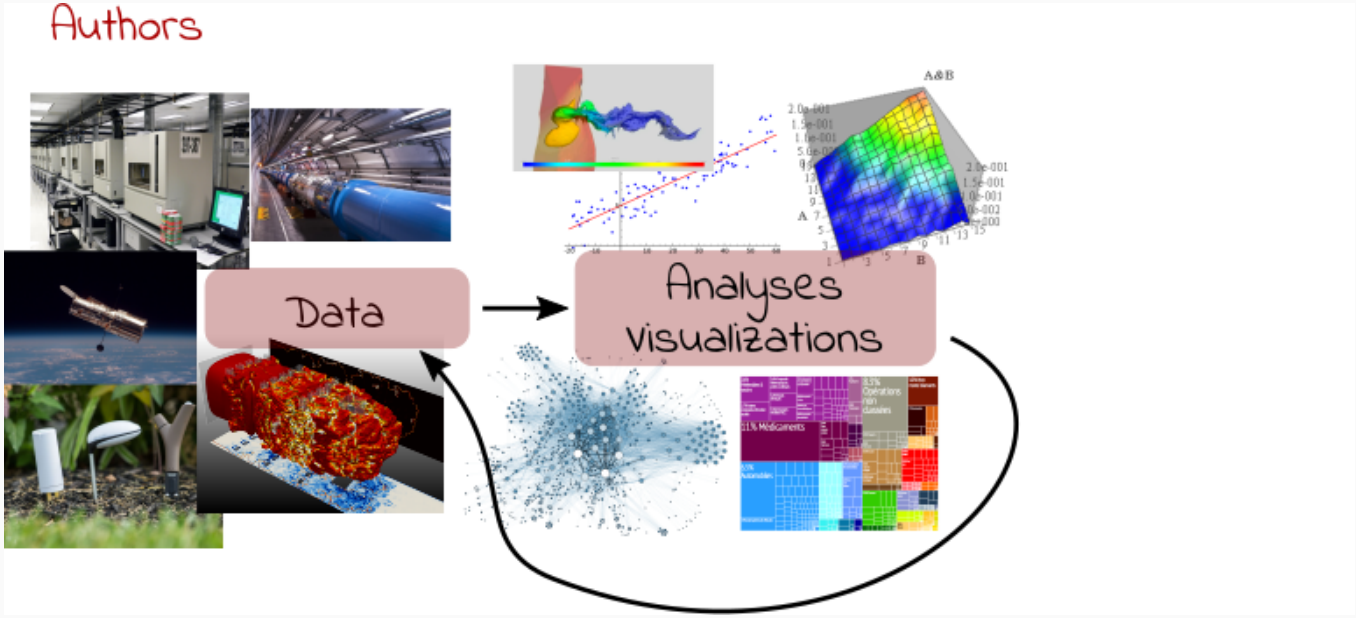
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Different Reproducibility Concerns in Modern Science

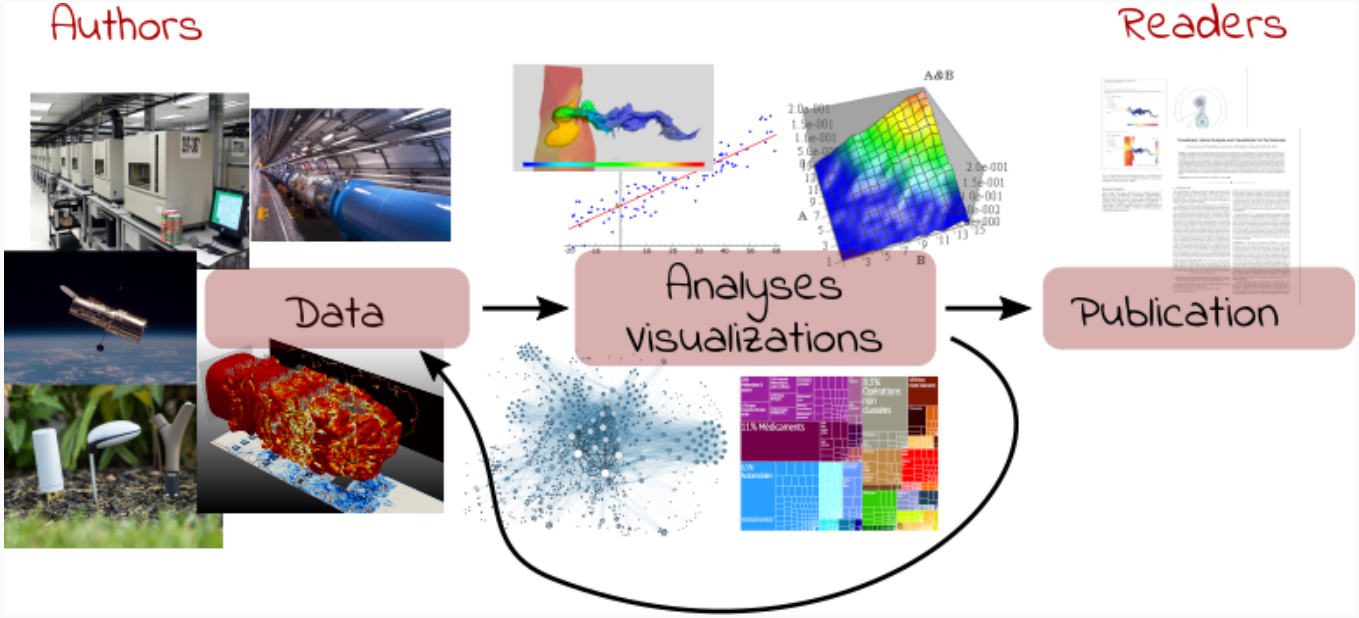
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Different Reproducibility Concerns in Modern Science

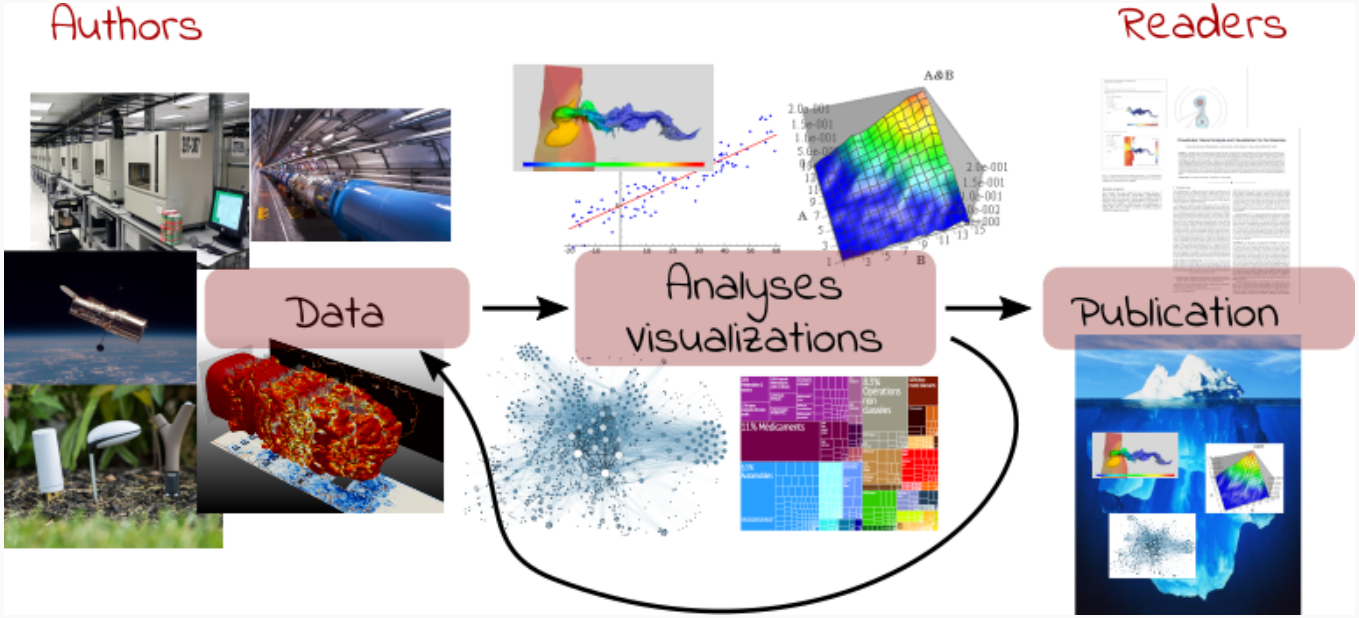
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Different Reproducibility Concerns in Modern Science

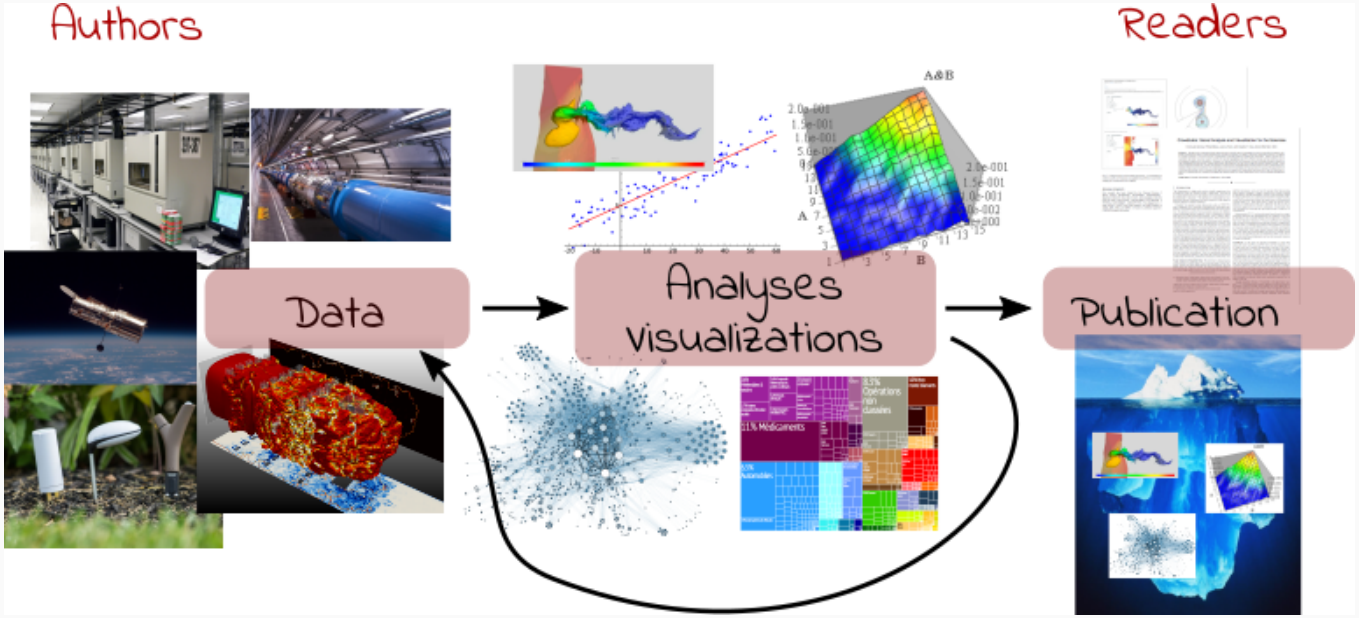
Social Sciences, Oncology, ... methodology, statistics, pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Artificial Intelligence most of the above

The processing steps between raw observations and findings have gotten increasingly numerous and complex



Reproducible Research = Bridging the Gap by working Transparently

Reproducible Research Practices

Reproducibility (glossary may vary)

Many definitions (*replicability, repeatability, reproducibility*), sometimes conflicting (*new data, same person, independant researcher*)

experimental reproducibility	similar input (data) + similar experimental protocol	→	**similar results**[1]
statistical reproducibility	different input (data) + same analysis	→	**same conclusions**[2]
computational reproducibility	similar input (data) + same code/software + same software environment	→	**exact same results**[3]

Reproducible Research = A way of doing science so that scientific experiments, discoveries, results, etc. can be easily reproduced (done again), to be confirmed, or to be built on for the next study.

– Courtesy G. Durrif, 2021

[1] Up-to measurement variability and precision
[2] Independently from (random) sampling variability (fight bias)
[3] Bitwise

"Reproducible Research": First Appearance

Claerbout & Karrenbach, meeting of the Society of Exploration Geophysics, 1992

Electronic Documents Give Reproducible Research a New Meaning **RE1.3**

Jon F. Claerbout and Martin Karrenbach, Stanford Univ.

SUMMARY

A revolution in education and technology transfer follows from the marriage of word processing and software command scripts. In this marriage an author attaches to every figure caption a pushbutton or a name tag usable to recalculate the figure from all its data, parameters, and programs. This provid[illegible] of reproducibility in compute[illegible]

In 1990, we set this sequence of goals:

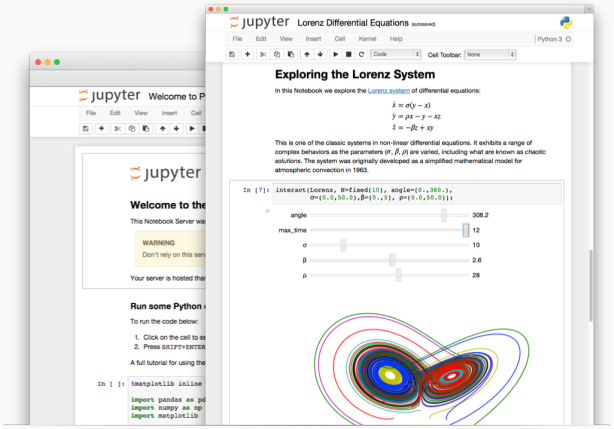
- Learn how to merge a publication with its underlying computational analysis.
- Teach researchers how to prepare a document in a form where they themselves can reproduce their own research results a year or more later by "pressing a single button".
- Learn how to leave finished work in a condition where coworkers can reproduce the calculation including the final illustration by pressing a button in its caption.
- Prepare a complete copy of our local software environment so that graduating students can take their work away with them to other sites, press a button, and reproduce their Stanford work.
- Merge electronic documents written by multiple authors (SEP reports).
- make incremental improvements in electronic-document software
- seek partners for broadening standards (and making incremental improvements).

Our basic goal is reproducible research. The electronic document is our means to this end. In principle, reproducibility in research can be achieved without electronic documents and that is how we started. Our first nonelectronic reproducible document was a textbook in which the paper document contained the name of a program script in every figure caption. The program scripts were organized by book chapter and section so they could be correlated to an accompanying magnetic tape dump of the file system. The magnetic tape also contained all the necessary data to feed the program script.

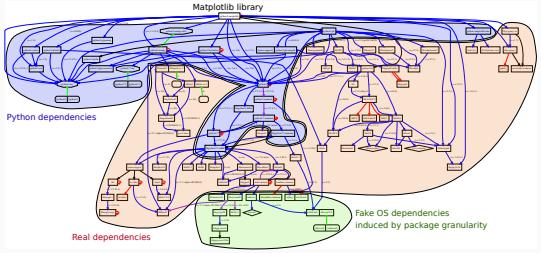
Now that we have begun using CD-ROM publication, we can go much further. Every figure caption contains a pushbutton that jumps to the appropriate science directory (folder) and initiates a figure rebuild command and then displays the figure, possibly as a movie or interactive program. We normally display seismic images of the earth's interior. but to reach wider audiences, Figure 1 shows a satellite weather picture which the pushbutton will animate as seen on commerical television. We include all our plot software as well as freely available software from many sources, including compilers and the LaTeX word processing system. Naturally

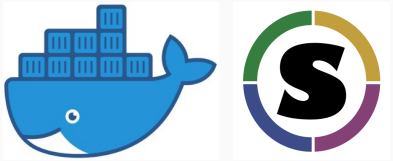
Existing Tools, Emerging Standards

Notebooks and workflows

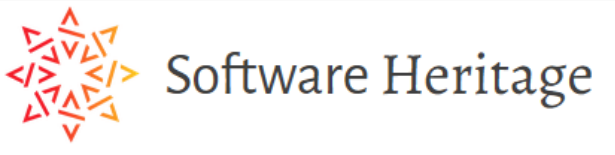


Software environments



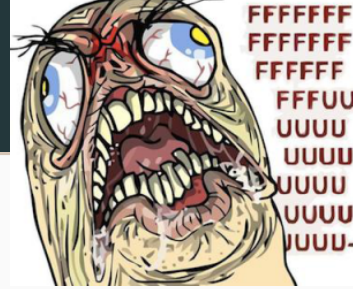


Sharing platforms



Good Practice #1
Taking Notes and Documenting

Frustration as an Author/Reviewer



Author

- I thought I used the same parameters but I'm getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can't remember
- The damned fourth reviewer asked for a major revision and wants me to change Figure 3. Which code and which data set did I use?
- It worked yesterday! 6 months later: Why did I do that?

Reviewer

- As usual, there is no confidence interval, I wonder about the variability and whether the difference is significant or not
- That can't be true, I'm sure they removed some points
- Why is this graph in logscale? How would it look like otherwise? I'm not even sure of what this value means. If only I could access the generation script

Tool 1: Computational Notebooks/Litterate Programming

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

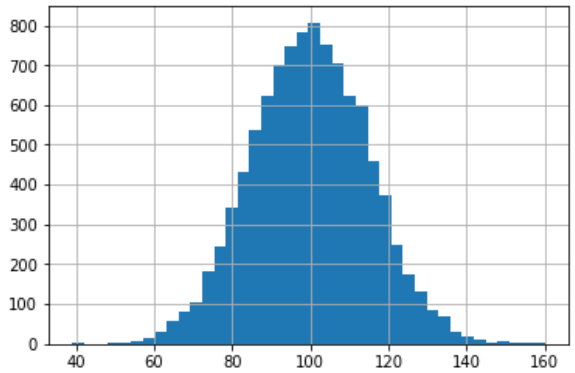
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement

jupyter example_pi (autosaved)

File Edit View Insert Cell Kernel Widgets Help Hide Code Python 3

Code CellToolbar

Hide Prompts Hide Code Hide Outputs

```
# Un document computationnel
```

Hide Prompts Hide Code Hide Outputs

```
Mon ordinateur m'indique que $\pi$ vaut *approximativement*
```

In [1]:

Hide Prompts Hide Code Hide Outputs

```
from math import *
print(pi)
```

```
3.141592653589793
```

Hide Prompts Hide Code Hide Outputs

```
Mais calculé avec la __méthode__ des [aiguilles de Buffon]
(https://fr.wikipedia.org/wiki/Aiguille_de_Buffon), on obtiendrait comme
__approximation__ :
```

In [2]:

Hide Prompts Hide Code Hide Outputs

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

Out[2]: 3.1437198694098765

Hide Prompts Hide Code Hide Outputs

```
On peut inclure des formules mathématiques comme $\frac{1}{\sigma
\sqrt{2\pi}} \exp \left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des
_dessins qui n'ont rien à voir_ avec $\pi$ (si ce n'est une constante de
normalisation... ☺).
```

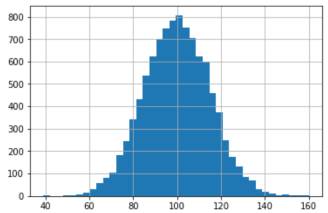
In [3]:

Hide Prompts Hide Code Hide Outputs

```
%matplotlib inline
import matplotlib.pyplot as plt

mu, sigma = 100, 15
x = mu + sigma*np.random.randn(10000)

plt.hist(x,40)
plt.grid(True)
plt.show()
```



Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

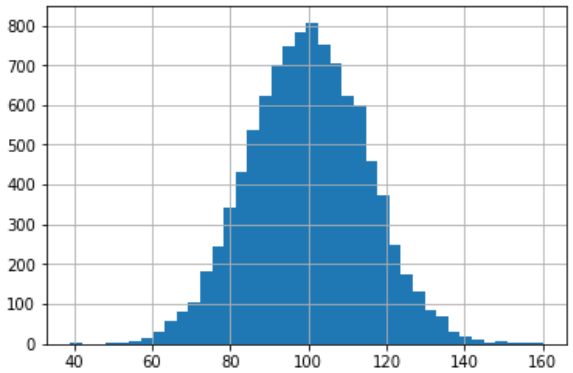
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement

Document final

jupyter example_pi (autosaved)

File Edit View Insert Cell Kernel Widgets Help Hide Code Python 3

Code CellToolbar

Hide Prompts Hide Code Hide Outputs

```
# Un document computationnel
```

Hide Prompts Hide Code Hide Outputs

```
Mon ordinateur m'indique que $\pi$ vaut *approximativement*
```

In [1]:

Hide Prompts Hide Code Hide Outputs

```
from math import *
print(pi)
```

```
3.141592653589793
```

Hide Prompts Hide Code Hide Outputs

```
Mais calculé avec la __méthode__ des [aiguilles de Buffon]
(https://fr.wikipedia.org/wiki/Aiguille_de_Buffon), on obtiendrait comme
__approximation__ :
```

In [2]:

Hide Prompts Hide Code Hide Outputs

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

Out[2]: 3.1437198694098765

Hide Prompts Hide Code Hide Outputs

```
On peut inclure des formules mathématiques comme $\frac{1}{\sigma
\sqrt{2\pi}} \exp \left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des
_dessins qui n'ont rien à voir_ avec $\pi$ (si ce n'est une constante de
normalisation... ☺).
```

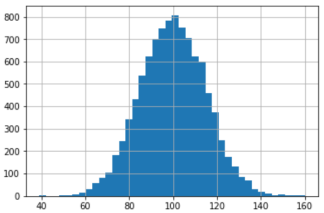
In [3]:

Hide Prompts Hide Code Hide Outputs

```
%matplotlib inline
import matplotlib.pyplot as plt

mu, sigma = 100, 15
x = mu + sigma*np.random.randn(10000)

plt.hist(x,40)
plt.grid(True)
plt.show()
```



Mark Down

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

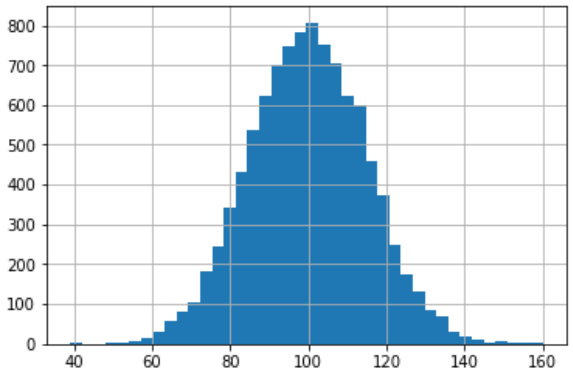
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

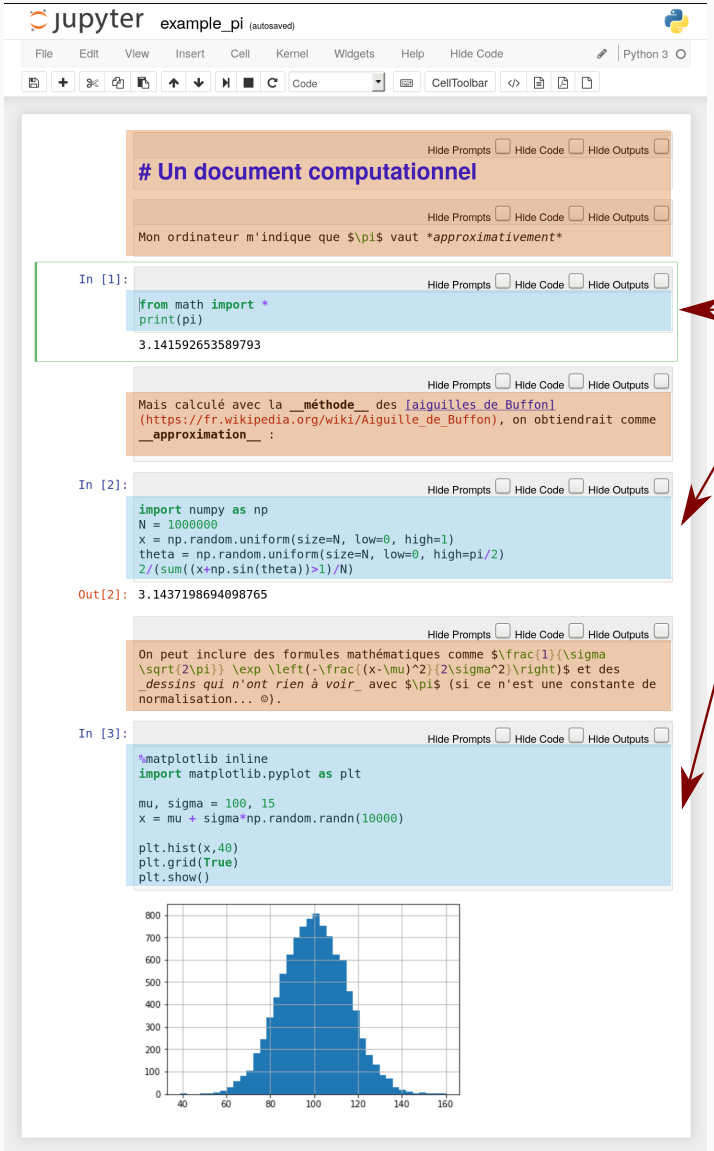
```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement



Code

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

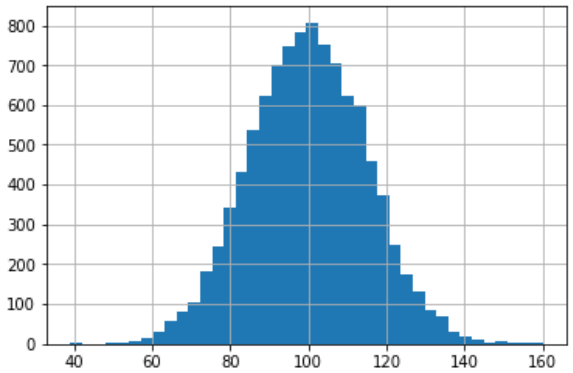
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

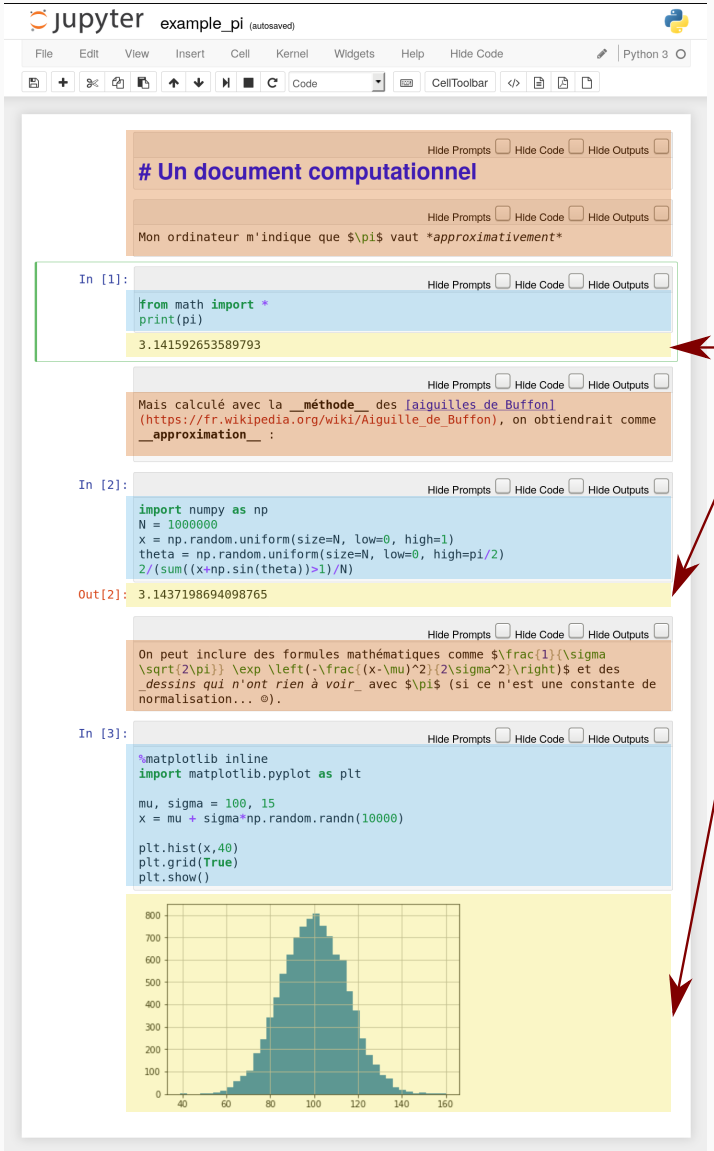
```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement



Résultats

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

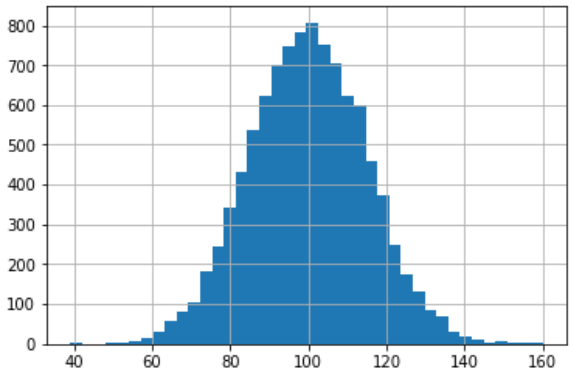
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

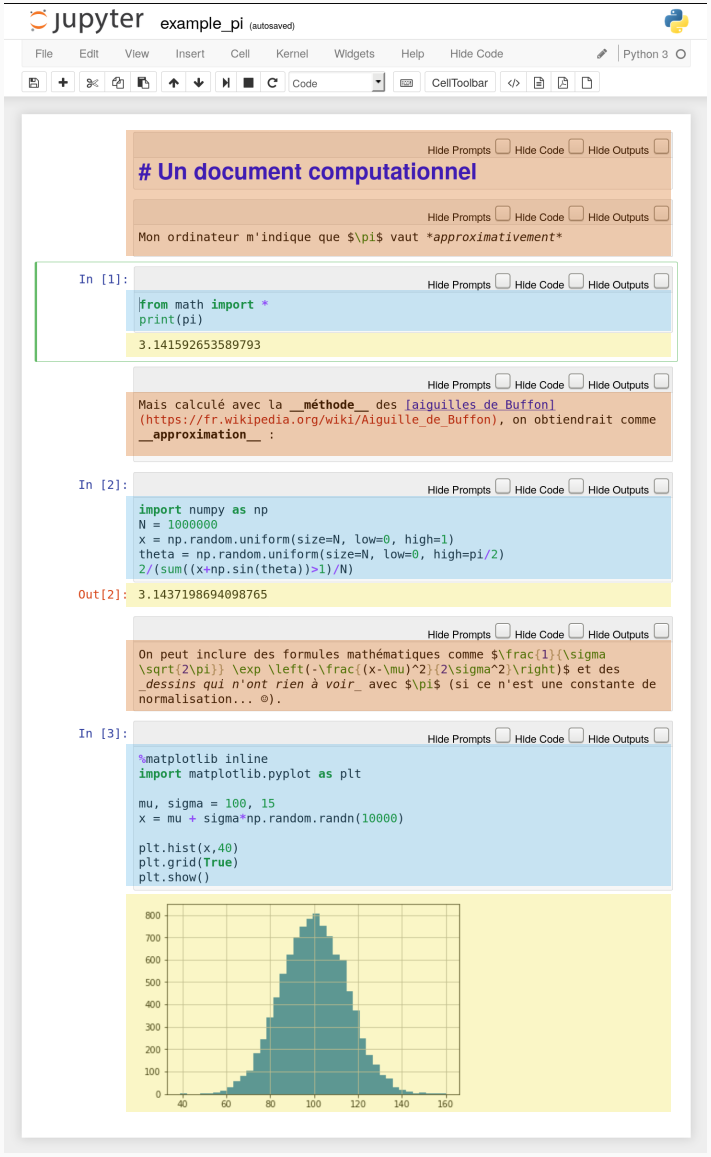
```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement



Export

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

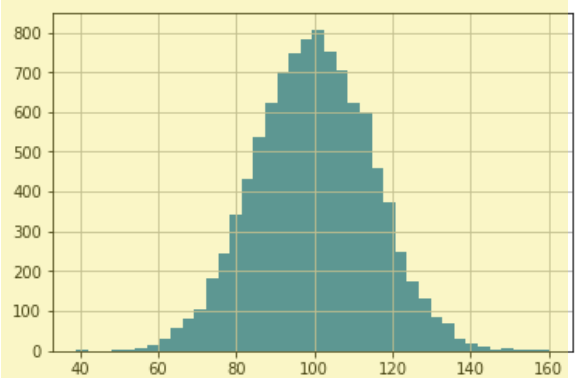
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).



Tool 1: Computational Notebooks/Litterate Programming

Document initial dans son environnement

jupyter example_pi (autosaved)

File Edit View Insert Cell Kernel Widgets Help Hide Code Python 3

Code CellToolbar

Hide Prompts Hide Code Hide Outputs

```
# Un document computationnel
```

Hide Prompts Hide Code Hide Outputs

```
Mon ordinateur m'indique que $\pi$ vaut *approximativement*
```

In [1]:

Hide Prompts Hide Code Hide Outputs

```
from math import *
print(pi)
```

```
3.141592653589793
```

Hide Prompts Hide Code Hide Outputs

```
Mais calculé avec la __méthode__ des [aiguilles de Buffon]
(https://fr.wikipedia.org/wiki/Aiguille_de_Buffon), on obtiendrait comme
__approximation__ :
```

In [2]:

Hide Prompts Hide Code Hide Outputs

```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

Out[2]: 3.1437198694098765

Hide Prompts Hide Code Hide Outputs

```
On peut inclure des formules mathématiques comme $\frac{1}{\sigma
\sqrt{2\pi}} \exp \left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des
_dessins qui n'ont rien à voir_ avec $\pi$ (si ce n'est une constante de
normalisation... ☺).
```

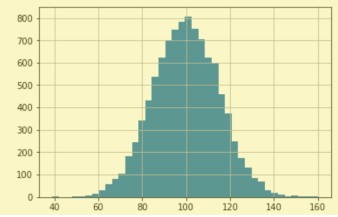
In [3]:

Hide Prompts Hide Code Hide Outputs

```
%matplotlib inline
import matplotlib.pyplot as plt

mu, sigma = 100, 15
x = mu + sigma*np.random.randn(10000)

plt.hist(x,40)
plt.grid(True)
plt.show()
```



Export

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

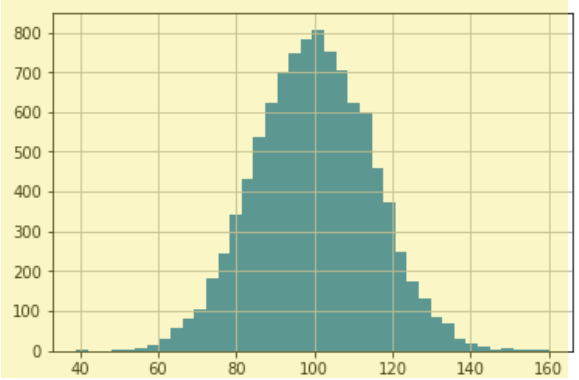
```
3.141592653589793
```

Mais calculé avec la **méthode** des aiguilles de Buffon, on obtiendrait comme **approximation** :

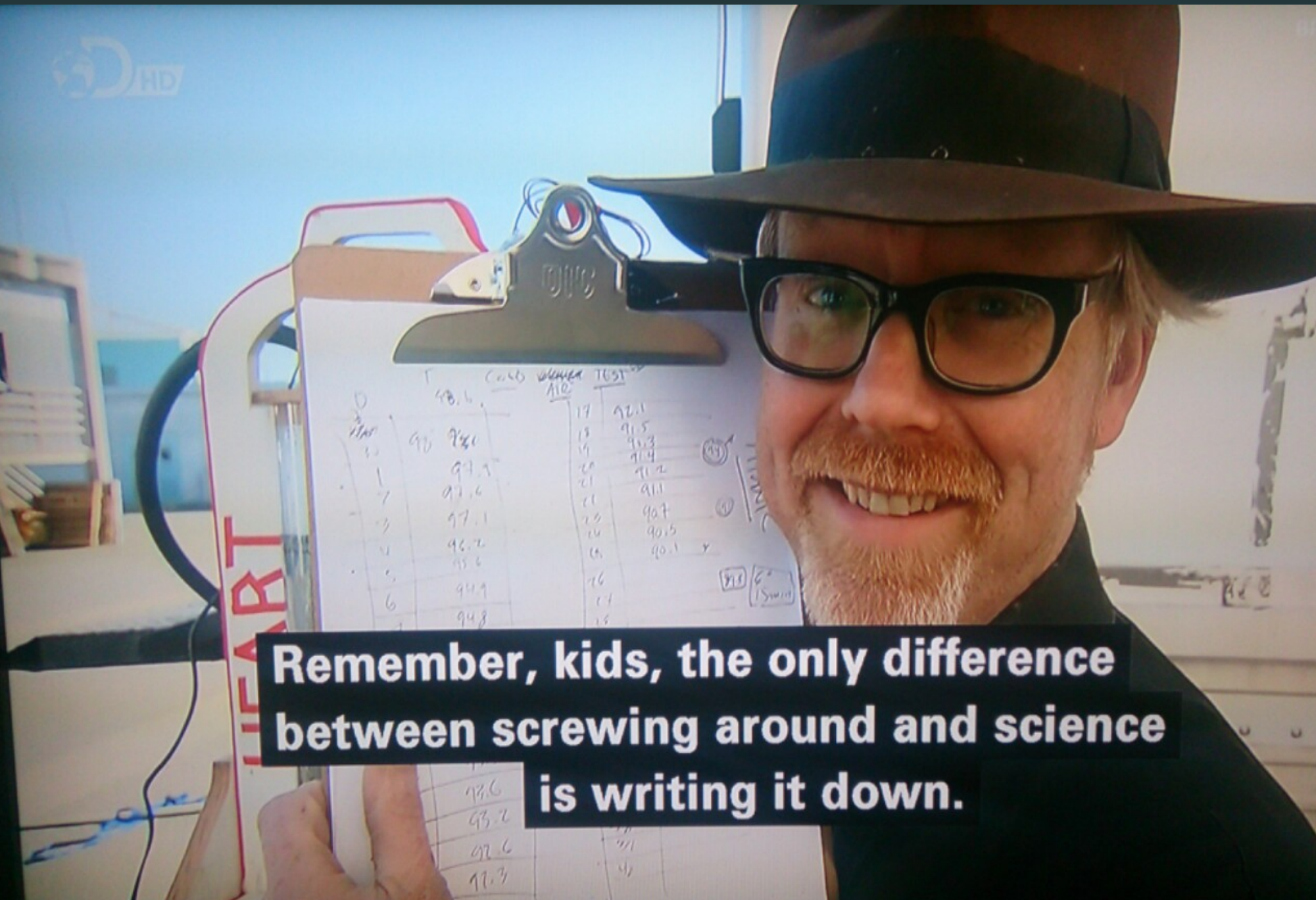
```
import numpy as np
N = 1000000
x = np.random.uniform(size=N, low=0, high=1)
theta = np.random.uniform(size=N, low=0, high=pi/2)
2/(sum((x+np.sin(theta))>1)/N)
```

```
3.1437198694098765
```

On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et des *dessins qui n'ont rien à voir* avec π (si ce n'est une constante de normalisation... ☺).

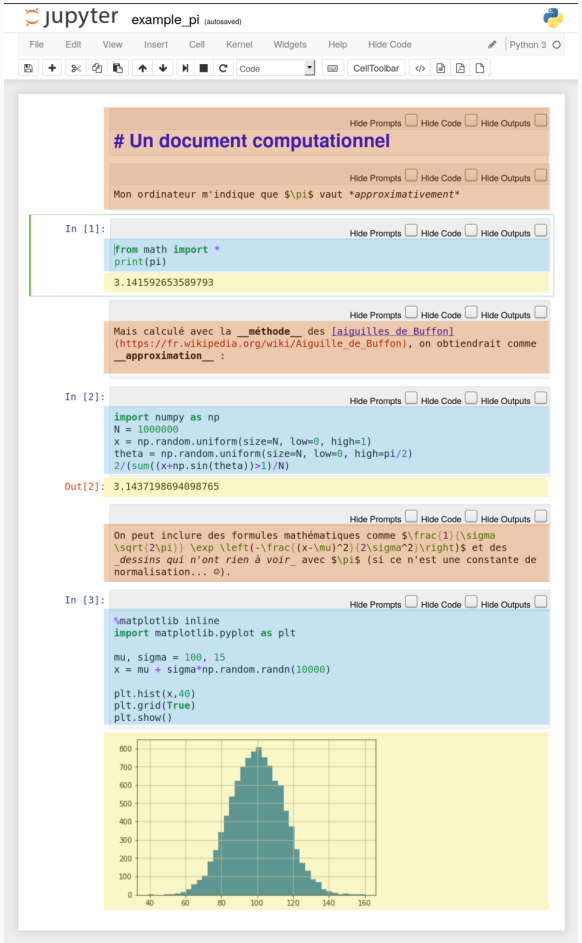


Tool 1 bis: Laboratory Notebooks, Computational Documents



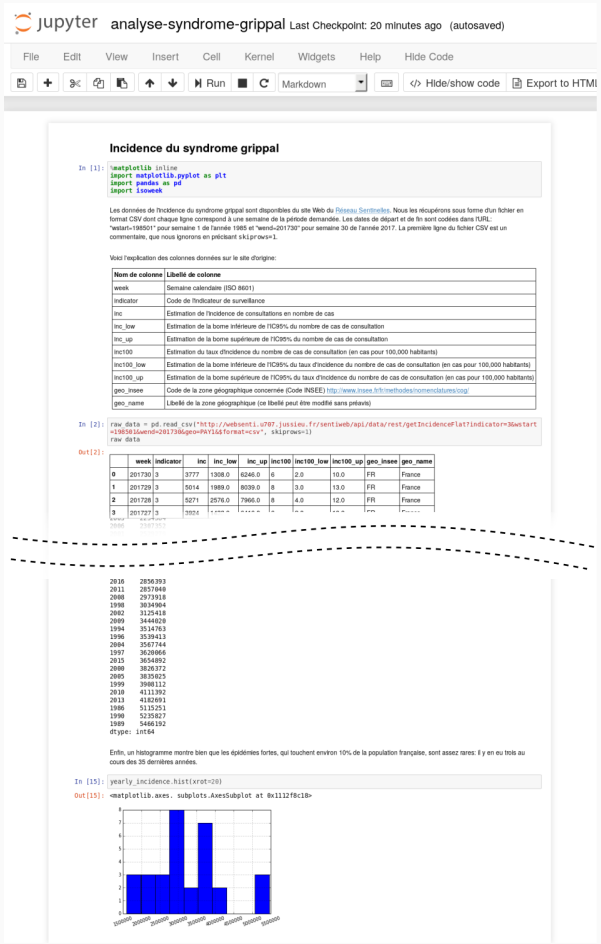
Tool 1 ter: Workflows

Notebooks are no panacea and do not help developing clean code



Tool 1 ter: Workflows

Notebooks are no panacea and do not help developing clean code



jupyter analyse-syndrome-grippal Last Checkpoint: 20 minutes ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help Hide Code

Run Markdown Hide/show code Export to HTML

Incidence du syndrome grippal

```
In [1]: %matplotlib inline
        import matplotlib.pyplot as plt
        import pandas as pd
        import isoweek
```

Les données de l'incidence du syndrome grippal sont disponibles du site Web du Réseau Sentinelles. Nous les récupérons sous forme d'un fichier en format CSV dont chaque ligne correspond à une semaine de la période demandée. Les dates de départ et de fin sont codées dans l'URL : "sstart=198501" pour semaine 1 de l'année 1985 et "send=201730" pour semaine 30 de l'année 2017. La première ligne du fichier CSV est un commentaire, que nous ignorons en précisant `skiprows=1`.

Voici l'explication des colonnes données sur le site d'origine:

Nom de colonne	Libellé de colonne
week	Semaine calendaire (ISO 8601)
indicator	Code de l'indicateur de surveillance
inc	Estimation de l'incidence de consultations en nombre de cas
inc_low	Estimation de la borne inférieure de l'IC95% du nombre de cas de consultation
inc_up	Estimation de la borne supérieure de l'IC95% du nombre de cas de consultation
inc100	Estimation du taux d'incidence du nombre de cas de consultation (en cas pour 100,000 habitants)
inc100_low	Estimation de la borne inférieure de l'IC95% du taux d'incidence du nombre de cas de consultation (en cas pour 100,000 habitants)
inc100_up	Estimation de la borne supérieure de l'IC95% du taux d'incidence du nombre de cas de consultation (en cas pour 100,000 habitants)
geo_insee	Code de la zone géographique concernée (Code INSEE) http://www.insee.fr/fr/methodes/nomenclatures/cog/
geo_name	Libellé de la zone géographique (ce libellé peut être modifié sans préavis)

```
In [2]: raw_data = pd.read_csv("http://websenti.u707.jussieu.fr/sentiweb/api/data/rest/getIncidenceFlat?indicator=3&sstart
        =198501&sweend=201730&geo=PAYS1&format=csv", skiprows=1)
        raw_data
```

Out[2]:

	week	indicator	inc	inc_low	inc_up	inc100	inc100_low	inc100_up	geo_insee	geo_name
0	201730	3	3777	1308.0	6246.0	6	2.0	10.0	FR	France
1	201729	3	5014	1989.0	8039.0	8	3.0	13.0	FR	France
2	201728	3	5271	2576.0	7966.0	8	4.0	12.0	FR	France
3	201727	3	3924							

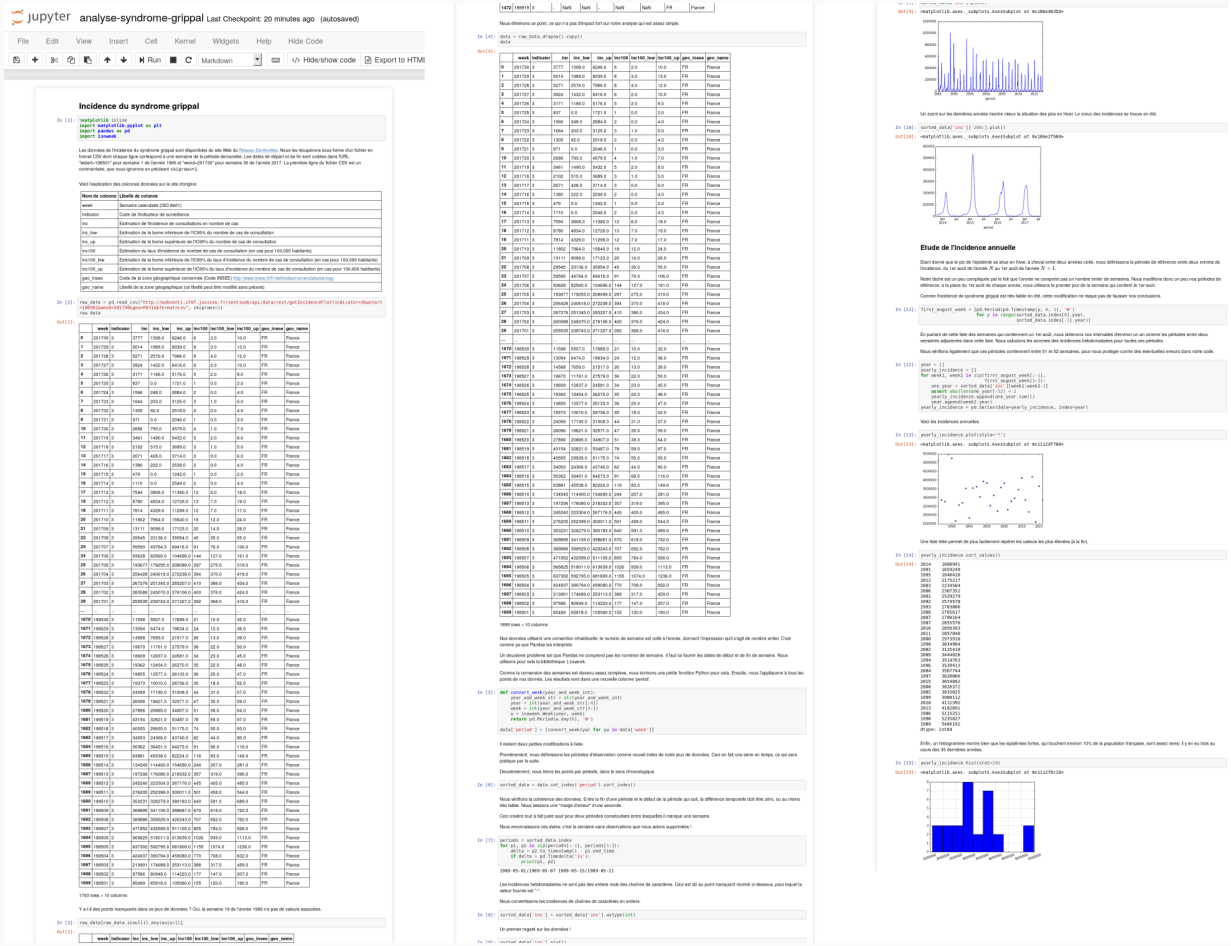
```
2016    2856393
2011    2857040
2008    2973918
1998    3034904
2002    3125418
2009    3444028
1994    3514763
1996    3539413
2004    3567744
1997    3620066
2015    3654892
2000    3826372
2005    3835025
1999    3908112
2010    4111392
2013    4182691
1986    5115251
1990    5235827
1989    5466192
dtype: int64
```

Enfin, un histogramme montre bien que les épidémies fortes, qui touchent environ 10% de la population française, sont assez rares: il y en eu trois au cours des 35 dernières années.

```
In [15]: yearly_incidence.hist(xrot=20)
Out[15]: <matplotlib.axes._subplots.AxesSubplot at 0x1112f8c18>
```

Tool 1 ter: Workflows

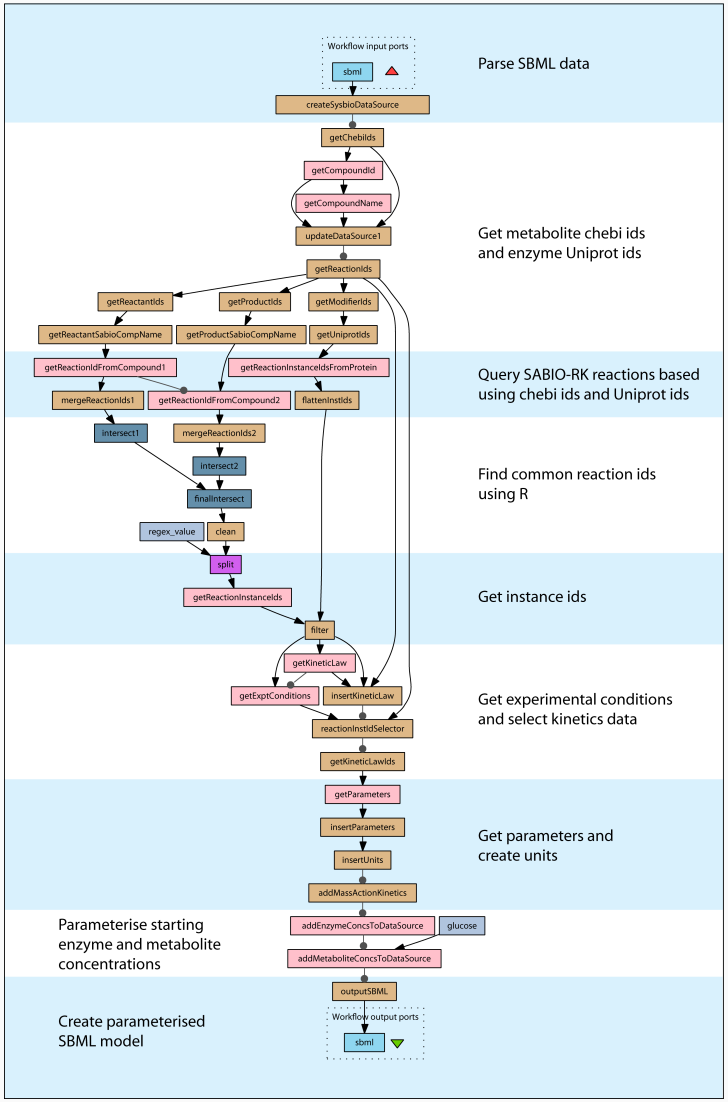
Notebooks are no panacea and do not help developing clean code

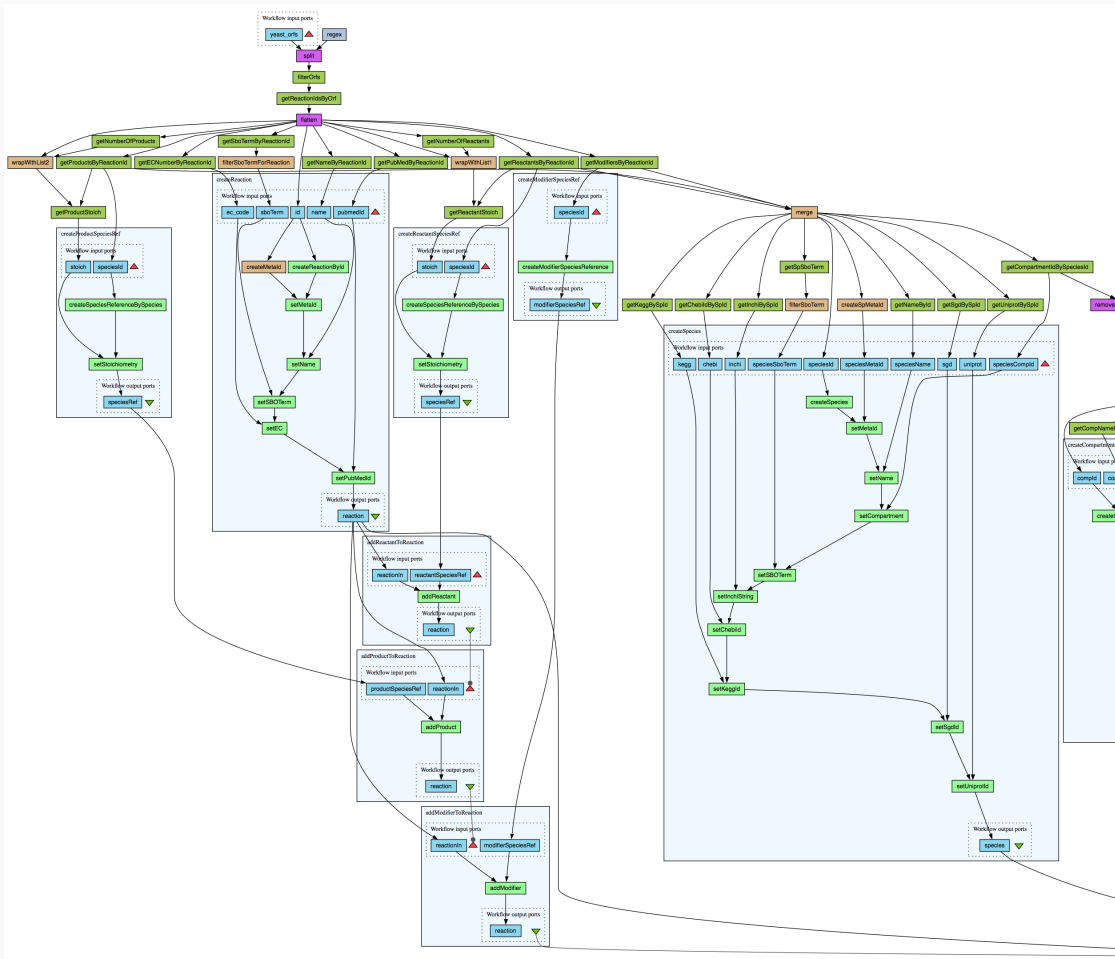


Tool 1 ter: Workflows

Notebooks are no panacea and do not help developing clean code

Tool 1 ter: Workflows





Tool 1 ter: Workflows

Workflows:

- Clearer high-level view
- Explicit composition of codes and data movement
- Safer sharing, reusing, and execution
- Notebooks are a variant that is both impoverished and richer
  - No simple/mature path from a notebook to a workflow

Examples:

- Galaxy, Kepler, Taverna, Pegasus, Collective Knowledge, VisTrails
- Light-weight: `make`, dask, drake, swift, `snakemake`, ...
- Hybrids: SOS-notebook, ...

Good Practice #2
Controling Software Environment

Argh... damned computers

- **Alice**: I got 3.123123 **Bob**: I got segfault
- Damned! It used to work!!! Whenever I upgrade my computer, things break so I try to stay away from this ☹

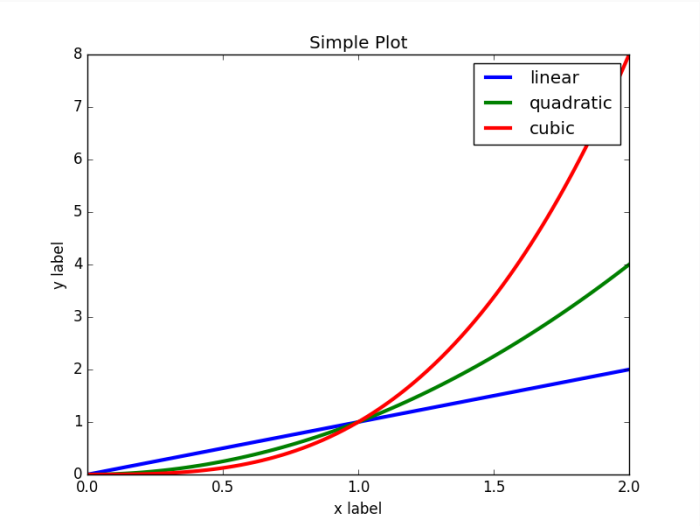
- Whenever trying the code of my colleague, I had to install `libFoo-1.5c` but I broke everything and now neither his code nor mine works! ☹

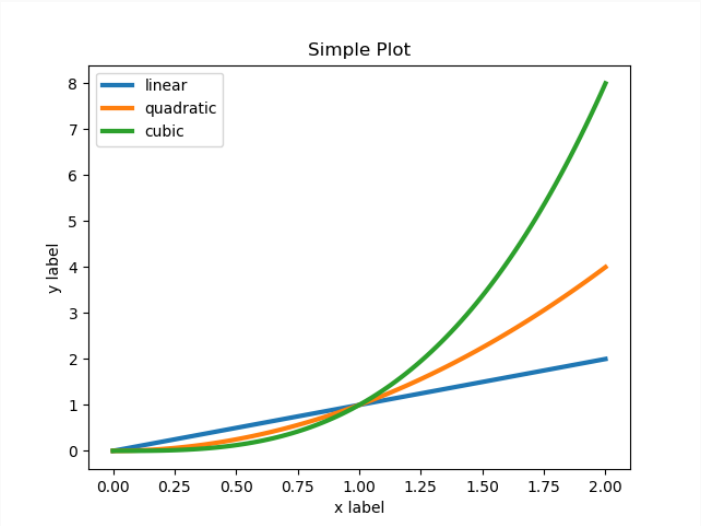
- But hey! Here is my code. It's on GitHub so feel free to play with it! I'm doing open science 😉
  1. No one will ever run/use your code if it isn't easy to install
  2. No one will ever manage to run your code if you don't document how to run it
  3. Others (even you) are unlikely to get the same results unless you control and share your software environment

Software dependencies: horror stories

Software dependencies: horror stories

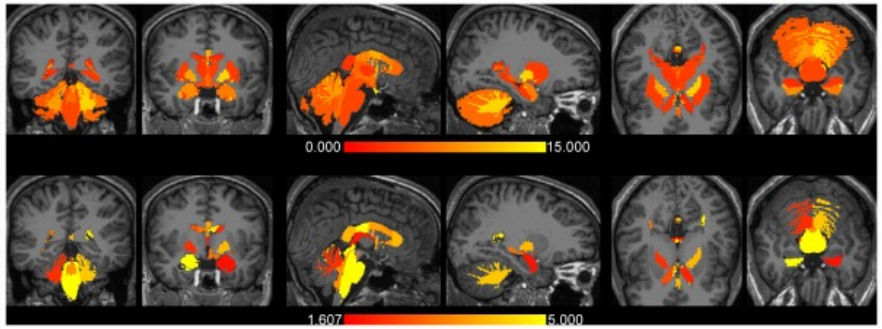
- Software environment evolution





Software dependencies: horror stories

- Software environment evolution
- OS heterogeneity

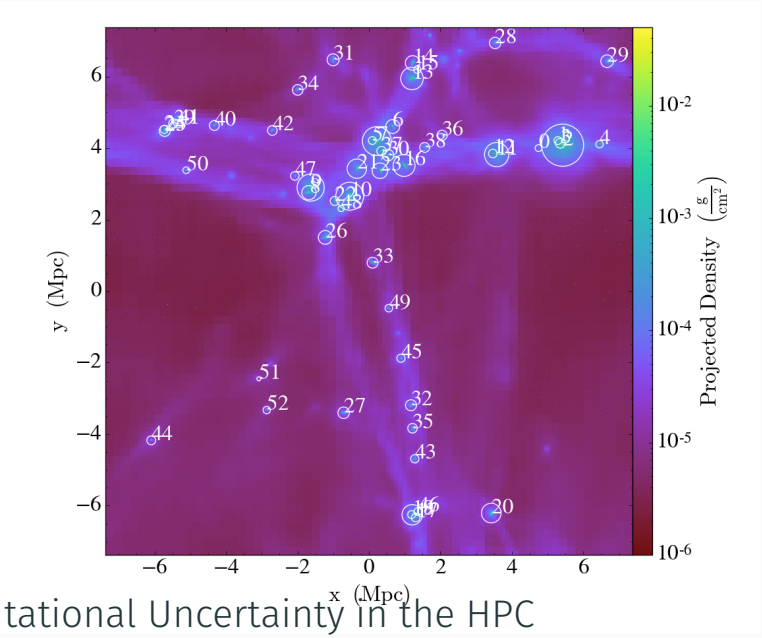


The Effects of FreeSurfer Version, Workstation Type, and Macintosh Operating System Version on Anatomical Volume and Cortical Thickness Measurements (PLOS ONE, 2012)

> *Significant differences in volume and cortical thickness were revealed across FreeSurfer versions:*
>
> - *volume:* *8.8 ± 6.6% (range 1.3-64.0%)*
> - *cortical thickness:* *2.8 ± 1.3% (range 1.1-7.7%)*
>
> *About a factor two smaller differences were found between the Mac and HP workstations and between Mac OSX 10.5 and OSX 10.6.*
>
> *In the context of an ongoing study, users are discouraged to update to a new major release of either FreeSurfer or operating system.*
>
> *Formal assessment of the accuracy of FreeSurfer is desirable.*

Software dependencies: horror stories

- Software environment evolution
- OS heterogeneity
- Impact of the compiler

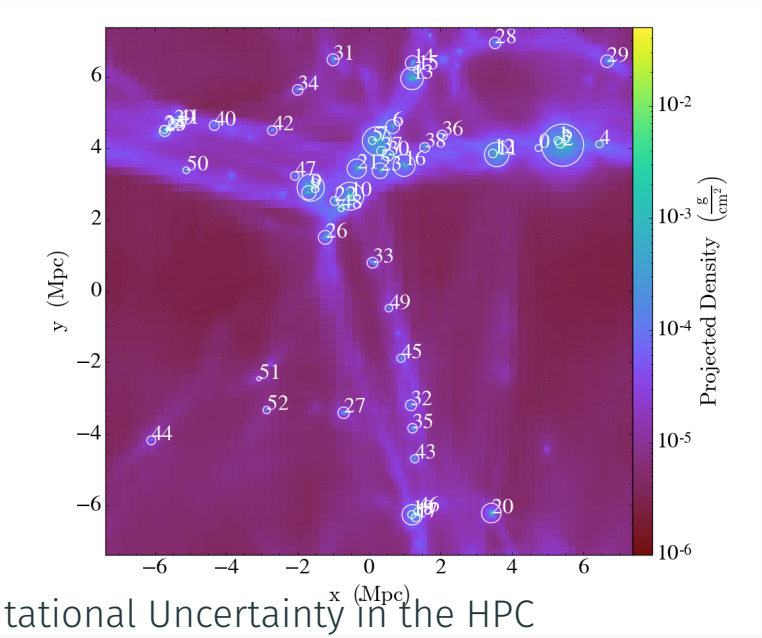


Assessing Reproducibility: An Astrophysical Example of Computational Uncertainty in the HPC Context (ResCuE-HPC, 2018)

Compiler	Optim.	Largest Halo Avg Mass.	Std. Err	Walltime
gcc@6.2.0	None	2.273E 46	1.069E 44	22h

Software dependencies: horror stories

- Software environment evolution
- OS heterogeneity
- Impact of the compiler

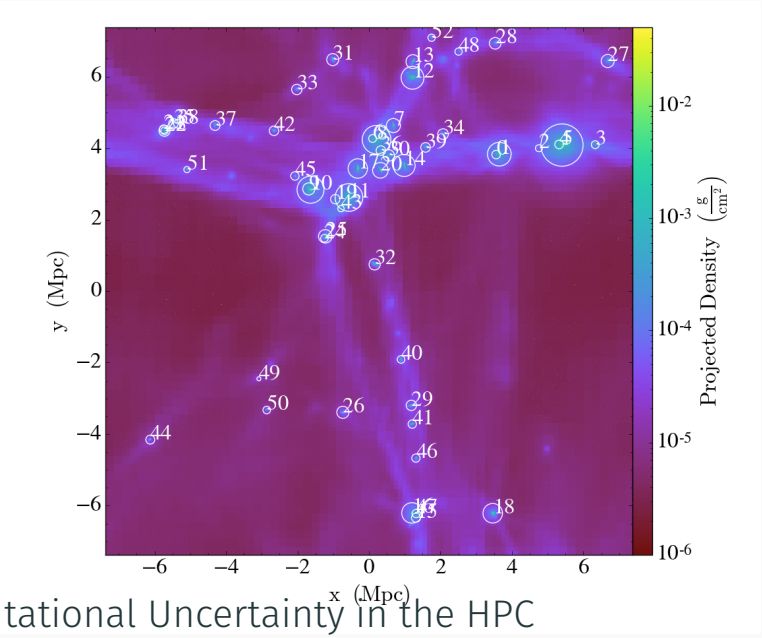


Assessing Reproducibility: An Astrophysical Example of Computational Uncertainty in the HPC Context (ResCuE-HPC, 2018)

Compiler	Optim.	Largest Halo Avg Mass.	Std. Err	Walltime
gcc@6.2.0	None	2.273E 46	1.069E 44	22h

Software dependencies: horror stories

- Software environment evolution
- OS heterogeneity
- Impact of the compiler

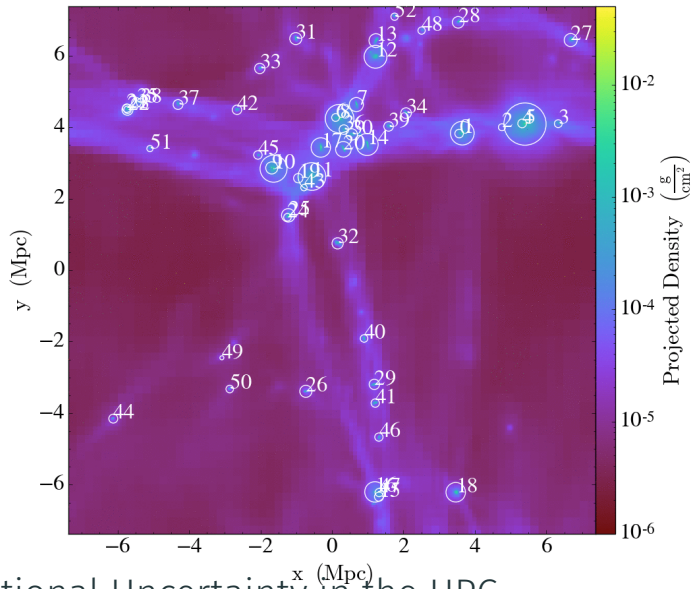


Assessing Reproducibility: An Astrophysical Example of Computational Uncertainty in the HPC Context (ResCuE-HPC, 2018)

Compiler	Optim.	Largest Halo Avg Mass.	Std. Err	Walltime
gcc@6.2.0	None	2.273E 46	1.069E 44	22h
gcc@6.2.0	Normal	2.266E 46	1.218E 44	10h
gcc@6.2.0	High	2.275E 46	1.199E 44	9h

Software dependencies: horror stories

- Software environment evolution
- OS heterogeneity
- Impact of the compiler



Assessing Reproducibility: An Astrophysical Example of Computational Uncertainty in the HPC Context (ResCuE-HPC, 2018)

Compiler	Optim.	Largest Halo Avg Mass.	Std. Err	Walltime
gcc@6.2.0	None	2.273E 46	1.069E 44	22h
gcc@6.2.0	Normal	2.266E 46	1.218E 44	10h
gcc@6.2.0	High	2.275E 46	1.199E 44	9h
intel@16.0.3	None	22.71 E 46	1.587E 44	39h
intel@16.0.3	Normal	43.30 E 46	1.248E 44	7h
intel@16.0.3	High	2.268E 46	1.414E 44	6h
cce@8.5.5	Low	43.11 E 46	1.353E 44	16h
cce@8.5.5	Normal	2.271E 46	1.261E 44	6h
cce@8.5.5	High	2.272E 46	1.341E 44	5h

Complex ecosystems

```
import matplotlib
print(matplotlib.__version__)
```

```
3.5.1
```

Complex ecosystems

```
import matplotlib
print(matplotlib.__version__)
```

```
3.5.1
```

```
apt show python3-matplotlib
```

```
Package: python3-matplotlib
Version: 3.5.1-2+b1
Source: matplotlib (3.5.1-2)
Maintainer: Sandro Tosi <morph@debian.org>
Installed-Size: 27.6 MB
Depends: libjs-jquery, libjs-jquery-ui, python-matplotlib-data (>= 3.5.1),
         python3-dateutil, python3-pil.imagetk, python3-pyparsing (>= 1.5.6),
         python3-six (>= 1.4), python3-numpy (>= 1:1.20.0), python3-numpy-
abi9,
         python3 (<< 3.11), python3 (>= 3.9~), python3-cycler (>= 0.10.0),
         python3-fonttools, python3-kiwisolver, python3-packaging, python3-
pil,
         python3:any, libc6 (>= 2.29), libfreetype6 (>= 2.2.1),
         libgcc-s1 (>= 3.3.1), libqhull-r8.0 (>= 2020.1), libstdc++6 (>= 11)
Recommends: python3-tk
Suggests: dvipng, ffmpeg, fonts-staypuft, ghostscript, gir1.2-gtk-3.0, inkscape,
```

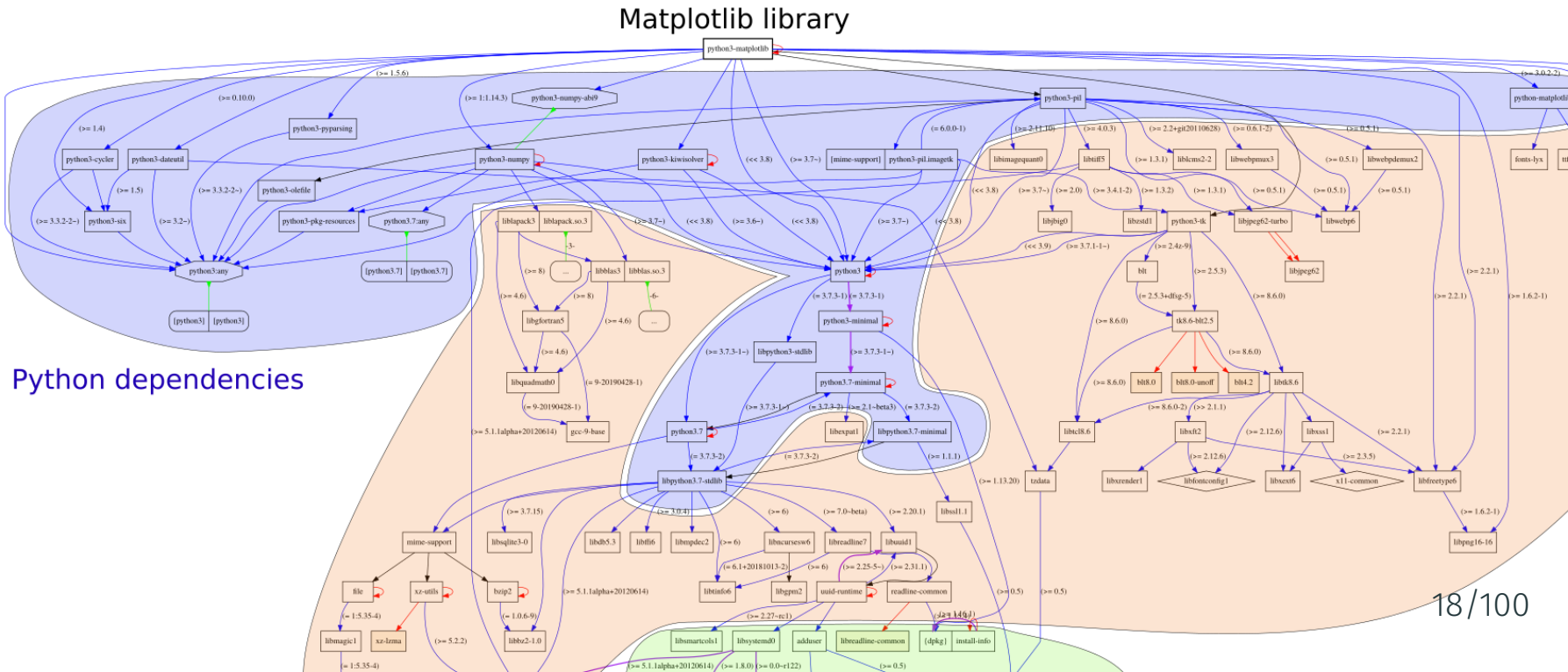
Complex ecosystems

```
import matplotlib
print(matplotlib.__version__)
```

3.5.1

```
apt show python3-matplotlib
```

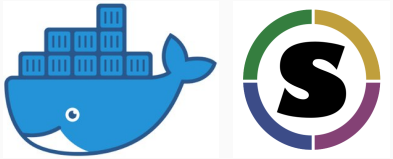
Package: python3-matplotlib



Tool 2: Containers and Package Managers

The good

The bad



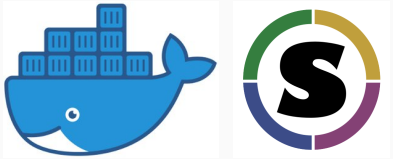
The uggly

Automatic tracking

Tool 2: Containers and Package Managers

The good

The bad



The uggly

Automatic tracking

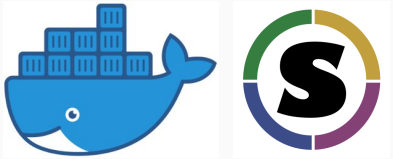
Containers

- Pros: Lightweight, Good isolation, Easy to use
  - Running as easy as `docker run <img> <cmd>`
  - Building images: `docker build -f <Dockerfile>`
  - Sharing through the Docker Hub: `docker pull/push <img>`

Tool 2: Containers and Package Managers

The good

The bad



The uggly

Automatic tracking

Containers

- Pros: Lightweight, Good isolation, Easy to use
- Cons: Opaque, Container build is generally not reproducible
  - Recipes rarely follow *reproducible good practices*

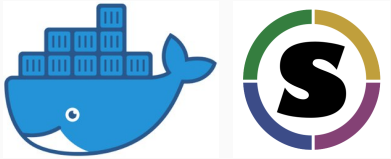
```
1  FROM ubuntu:20.04
2  RUN apt-get update
3       && apt-get upgrade -y
4       && apt-get install -y ...
```

- Choose a stable image (and the smallest possible)
- Include only the necessary libraries (e.g. no graphics libs)
- Avoid system updates (instead freeze sources)

Tool 2: Containers and Package Managers

The good

The bad



The uggly

Automatic tracking

Containers

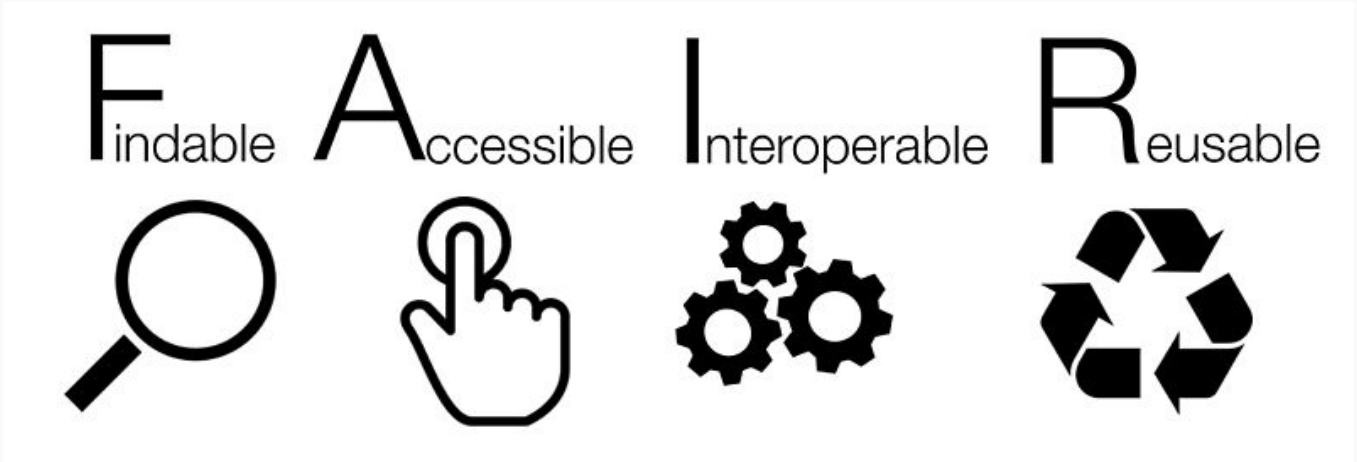
- Pros: Lightweight, Good isolation, Easy to use
- Cons: Opaque, Container build is generally not reproducible

Package managers (the uggly and the good)

- Language specific: `pip/pipenv/virtualenv`, `conda`, `CRAN/Bioconductor`
  - Limits: version management, durability, permeable, language centric
- GUIX/NiX = Full-fledged functional package manager
  - Native support for environment (*à la git*)
  - Isolation through `--pure`
  - Recompile from source (cache recommended)

Good Practice #3
Version Control and Archiving

FAIR principles



https://www.go-fair.org/fair-principles/

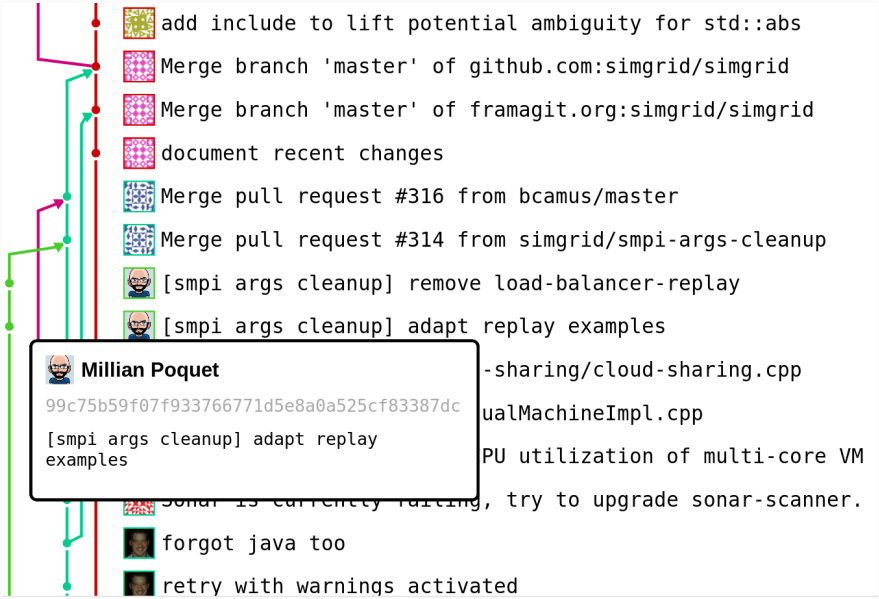
- *"Open as much as possible and close as much as necessary"*
- Management, publication, annotation (metadata), archiving
- Source code = specific data with specific consideration

Let's go beyond general principles!

Tool 3: Version Control and Forge

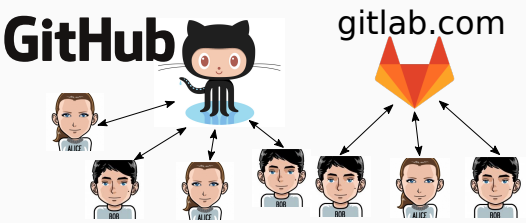
Git = version control

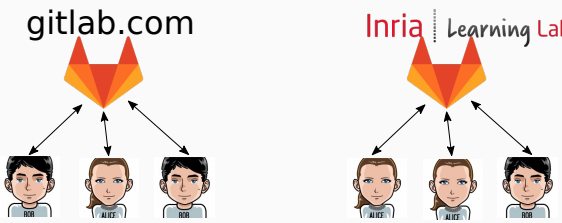
- Developed in 2005 by Linus Torvalds for the kernel development
- Local and efficient rollbacks
- Distributed: everyone has a full copy of the history

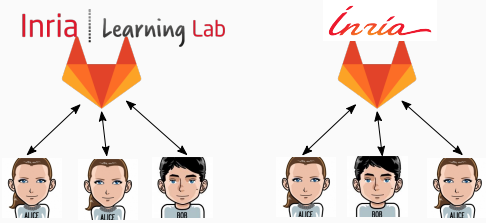


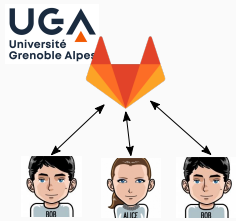
GitHub, GitLab, and Co

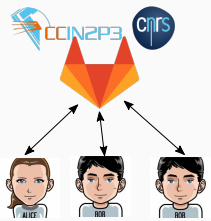
- Free hosting of public projects, social network











Limitation

- Managing large data: ~~Git LFS~~ `Git Annex` (or DataLad)

Tool 3bis: Fighting Information Loss with Archives

 or  = awesome collaborations ($\neq$ archive)

- D. Spinellis. *The Decay and Failures of URL References*. CACM, 46(1), 2003
  The half-life of a referenced URL is approximately 4 years from its publication date.
- P. Habibzadeh. *Decay of References to Web sites in Articles Published in General Medical Journals: Mainstream vs Small Journals*. Applied Clinical Informatics. 4 (4), 2013
  half life ranged from 2.2 years in EMHJ to 5.3 years in BMJ
- Discontinuated forges: Code Space, Gitorious, Google code, Inria Gforge

Tool 3bis: Fighting Information Loss with Archives

or

= awesome collaborations (≠ archive)

- D. Spinellis. *The Decay and Failures of URL References*. CACM, 46(1), 2003
  The half-life of a referenced URL is approximately 4 years from its publication date.
- P. Habibzadeh. *Decay of References to Web sites in Articles Published in General Medical Journals: Mainstream vs Small Journals*. Applied Clinical Informatics. 4 (4), 2013
  half life ranged from 2.2 years in EMHJ to 5.3 years in BMJ
- Discontinuated forges: Code Space, Gitorious, Google code, Inria Gforge

Article archives arXiv.org HAL archives-ouvertes.fr

Data archives figshare zenodo

Software Archive Software Heritage

INTERNET ARCHIVE

Collect/Preserve/Share

What Will it Take ?

Soft. Engineering, Statistics, and Reproducible Research in the curricula

Manifesto: *"I solemnly pledge"* (WSSSPE, Lorena Barba, FAIR)

1. I will teach my graduate students about reproducibility
2. All our research code (and writing) is under version control
3. We will always carry out verification and validation
4. We will share data, plotting script & figure under CC-BY
5. We will upload the preprint to arXiv at the time of submission of a paper
6. We will release code at the time of submission of a paper
7. We will add a "Reproducibility" declaration at the end of each paper
8. I will keep an up-to-date web presence

Learn and Teach using online resources like

- Software Carpentry, The Turing Way, ...

Changing Publishing Practices

Artifact evaluation and ACM badges

Major conferences

- Supercomputing: Artifact Description (AD) mandatory, Artifact Evaluation (AE) still optional, Double blind vs. RR
- NeurIPS, ICLR: open reviews, reproducibility challenge

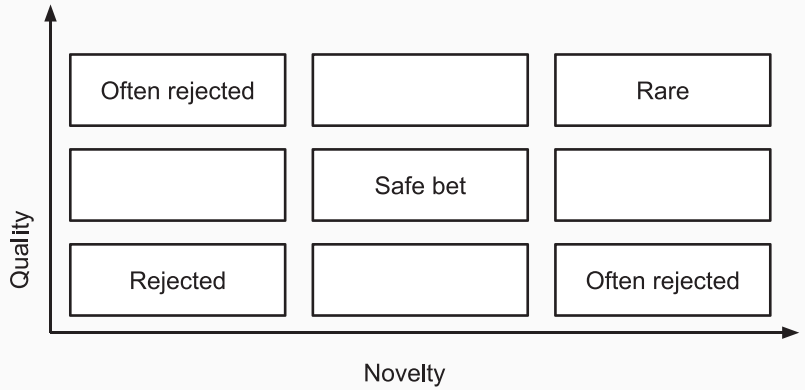
Joelle Pineau @ NeurIPS'18

- ACM SIGMOD 2015-2019, Most Reproducible Paper Award...

Mentalitie are evolving people care, make stuff available, errors are found and fixed

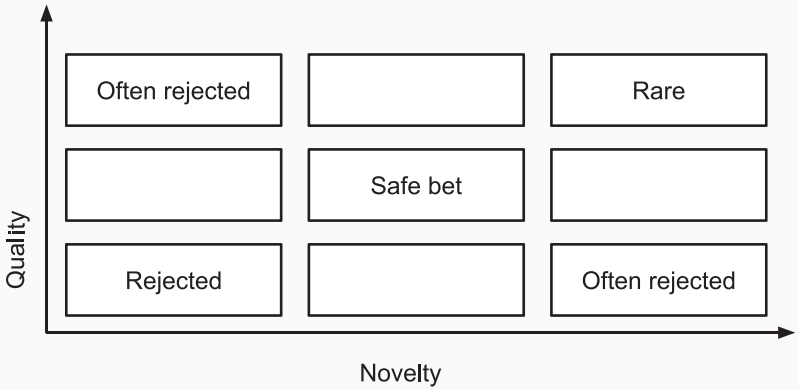
Changing Academic Practices (~~Publish or Perish~~)

- Goodhart's Law: Are Academic Metrics Being Gamed?, M. Fire 2019
  - AI: over 1,000 ranked journals ($\times$10 in 15 years)
  - Shorter papers with increasing self references
  - More and more papers without any citation
  - Sharp increase in the number of new authors publishing at a much faster rate given their career age
- The Truth, The Whole Truth, and Nothing But the Truth: A Pragmatic Guide to Assessing Empirical Evaluations, *TOPLAS* 2016



Changing Academic Practices (~~Publish or Perish~~)

- Goodhart's Law: Are Academic Metrics Being Gamed?, M. Fire 2019
  - AI: over 1,000 ranked journals (×10 in 15 years)
  - Shorter papers with increasing self references
  - More and more papers without any citation
  - Sharp increase in the number of new authors publishing at a much faster rate given their career age
- The Truth, The Whole Truth, and Nothing But the Truth: A Pragmatic, Guide to Assessing Empirical Evaluations, *TOPLAS* 2016



- Impact factor abandoned by Dutch university in hiring and promotion, decisions. Nature, June 2021. *Faculty and staff members at Utrecht University will be evaluated by their commitment to open science*

What about Open Science ?

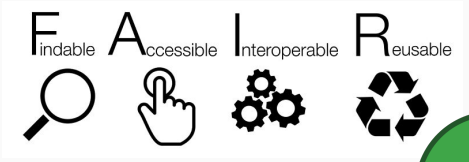
Plan National pour la Science Ouverte (BSN ⇝ CoSO)

- CNRS, Inria, INRAE, ...
- Many flavors: *Citizen Science*

Main pillars:

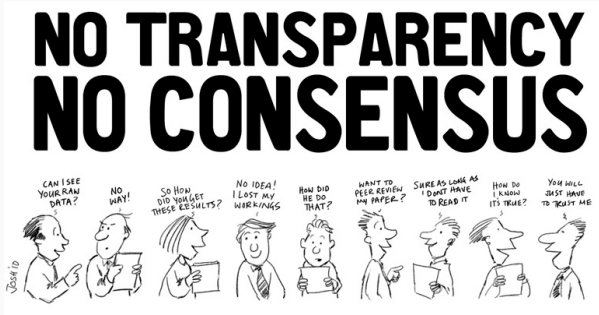
1. Open access

2. Open data



3. Open source
   - *Open hardware*

4. **Open methodology** (Reproducible Research)
   - *Open-notebook science*
   - *Open science infrastructures*



5. Open peer review (avoid collusion)

6. Open educational resources

That's all Folks!

Resources and Acknowledgments

A non-technical introduction to reproducibility issues (in French)

- Loïc Desquilbet, Sabrina Granger, Boris Hejblum, Pascal Pernot, Nicolas Rougier

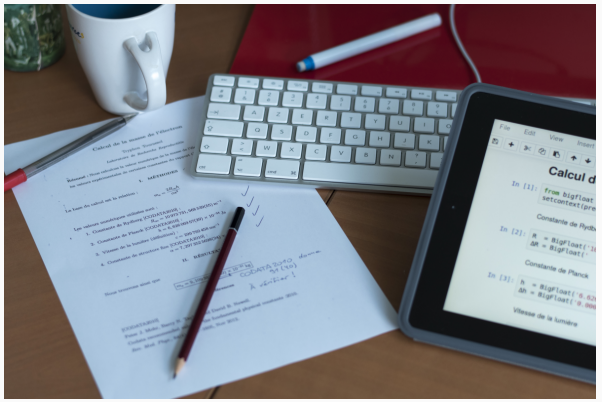
Resources and Acknowledgments

A non-technical introduction to reproducibility issues (in French)

- Loïc Desquilbet, Sabrina Granger, Boris Hejblum, Pascal Pernot, Nicolas Rougier

MOOC Reproducible Research: Methodological principles for a transparent science, Learning Lab Inria

- Konrad Hinsen, Christophe Pouzat
- 3rd Edition: March 2020 – March 2023 (15,000+)



Stay tuned for the MOOC "Advanced RR" planned for ~~2021~~ ~~2022~~ 2023

- Managing data (`FITS/HDF5`, `git annex`)
- Software environment control (`docker`, `singularity`, `guix`)
- Scientific workflow (`make`, `snakemake`)