

REPRODUCIBILITY CRISIS AND OPEN SCIENCE

Arnaud Legrand

Univ. Grenoble Alpes, CNRS, Inria

Inria Alumni: Science Ouverte, November 2019



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

Los Angeles Times | BUSINESS

LOCAL U.S. WORLD BUSINESS SPORTS ENTERTAINMENT HEALTH STYLE TRAVEL

Science has lost its way, at a big cost to humanity

Researchers are rewarded for splashy findings, not for double-checking accuracy. So many scientists looking for cures to diseases have been building on ideas that aren't even true.

Science AAAS-ORG FEEDBACK HELP LIBRARAINS All Science Journals ▾ Search This Site

AAAS NEWS SCIENCE JOURNALS CAREERS MULTIMEDIA COLLECTIONS

Science The World's Leading Journal of Original Scientific Research, Global News, and Commentary.

Science Magazine > 22 JUNE 2013 > MONDAY, 24 JUNE (1660): 229

Article Views ▾ Save to My Folders ▾ Full Text ▾ Full Text (PDF)

Editorial

Reproducibility

Marcia McNutt

► Marcia McNutt is editor-in-chief of *Science*.

Science advances on a foundation of trusted data. But a recent study found that the scientific community was shaken by reports that a trend of irreproducibility was widespread. For example, new research from the National Institutes of Health found that only 41 percent of studies fit one of the top three recommendations of the U.S. National Institute of General Medical Sciences' "Guidelines for Increasing Transparency." Authors will indicate handling (such as how to deal with outliers), whether they used a sufficient signal-to-noise ratio, whether the experimenter was blind to the conduct of the experiments, and whether the results were statistically significant.

TheScientist EXPLORING LIFE. INSPIRING INNOVATION

NIH Tackles Irreproducibility

The federal agency speaks out about how to improve the quality of scientific research.

By Jef Akst | January 28, 2014

Courtesy V. Stodden, SC, 2015

Announcement: Reducing our irreproducibility - Nature News & Comment

nature.com Sitemap Log in Register

nature International weekly journal of science

Home News & Comment Research Careers & Jobs Current Issue Archive

Archive > Volume 490 > Issue 7446 > Editorial > Article

NATURE | EDITORIAL

Announcement: Reducing our irreproducibility

24 April 2013

PDF Rights & Permissions

Over the past year, *Nature* has published a string of articles that highlight the lack of reproducibility of published research (collected on the right).

nature International weekly journal of science

Menu Advanced search Search

archive > volume 483 - issue 7591 - editorials - article

NATURE | EDITORIAL

Must try harder

Nature 483, 509 (29 March 2012) doi:10.1038/483509a

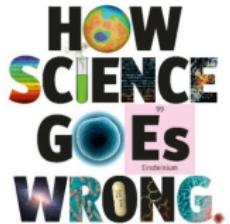
Published online: 28 March 2012

PDF Cite this Reprints Rights & permissions Article metrics

Too many sloppy mistakes are creeping into scientific papers. Lab heads must look more rigorously at the data — and at themselves.

The Economist

Washington's lawyer scruples
How to do a nuclear deal with Iran
Investment tips from Nobel economists
Junk bonds are back
The meaning of Sezin Tendilur



SCIENTIFIC MISCONDUCT ? WHAT ARE THE CONSEQUENCES ?

The Duke University scandal with scientific misconduct on lung cancer

- *Nature Medicine* - 12, (2006) Genomic signatures to guide the use of **chemotherapeutics**, by Anil Potti and 16 other researchers from Duke and USF
- Major commercial labs licensed it and were about to start using it before two statisticians discovered and publicized its faults

Dr. Baggerly and Dr. Coombes found errors almost immediately. Some seemed careless — moving a row or a column over by one in a giant spreadsheet — while others seemed inexplicable. The Duke team shrugged them off as “clerical errors.”

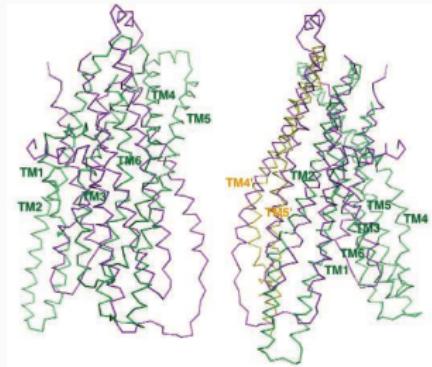
The Duke researchers continued to publish papers on their genomic signatures in prestigious journals. Meanwhile, they started three trials using the work to decide which drugs to give patients.

- Retractions: January 2011. **Ten papers that Potti coauthored in prestigious journals were retracted for varying reasons**

Bad science is deleterious

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

UNFORTUNATE MISTAKES



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

He specialized in structures of **multidrug resistant transporter proteins in bacteria**: MsbA de Escherichia 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 retractions that motivate improved software engineering practices in computational biology

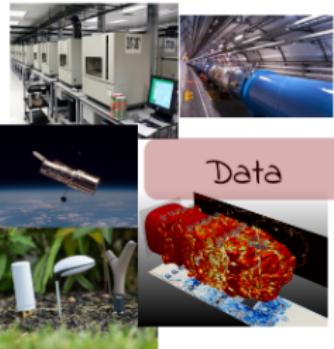
DIFFERENT REPRODUCIBILITY CONCERNS

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

Authors

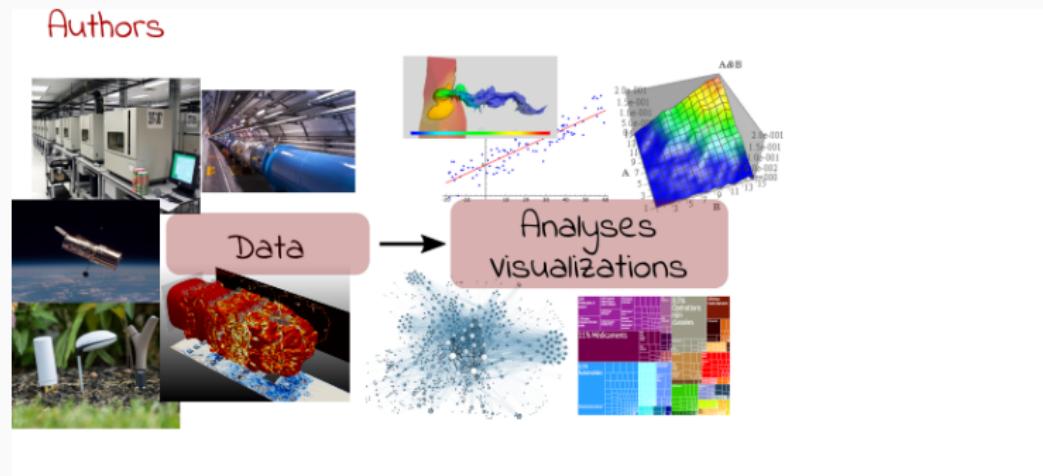


DIFFERENT REPRODUCIBILITY CONCERN

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

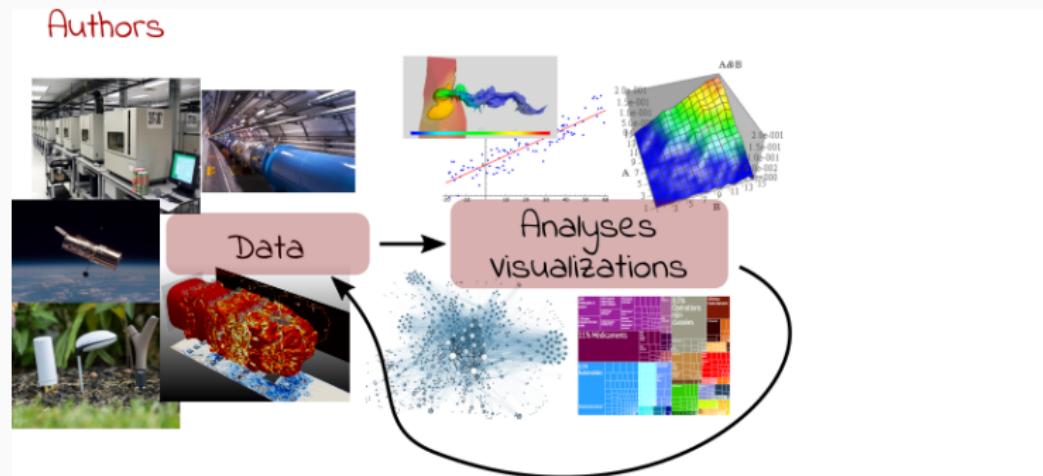


DIFFERENT REPRODUCIBILITY CONCERN

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

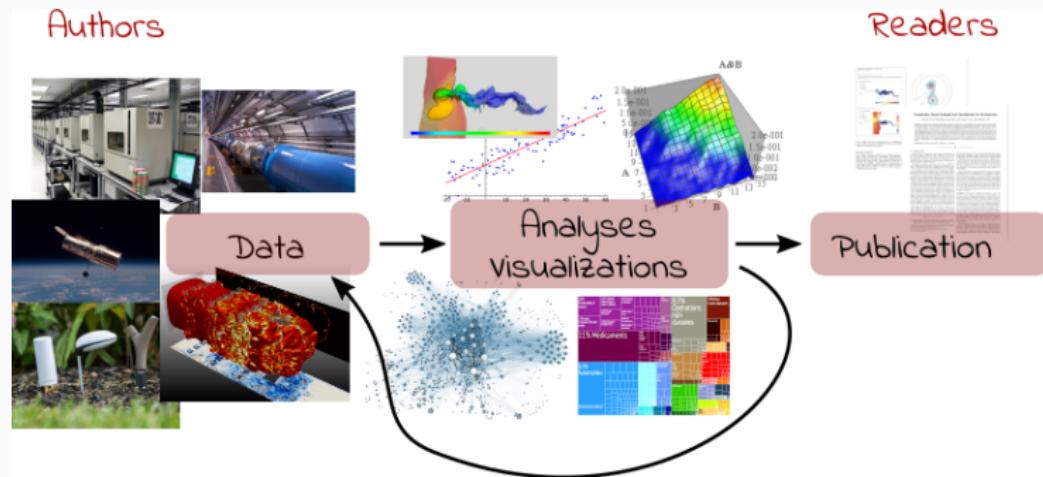


DIFFERENT REPRODUCIBILITY CONCERN

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

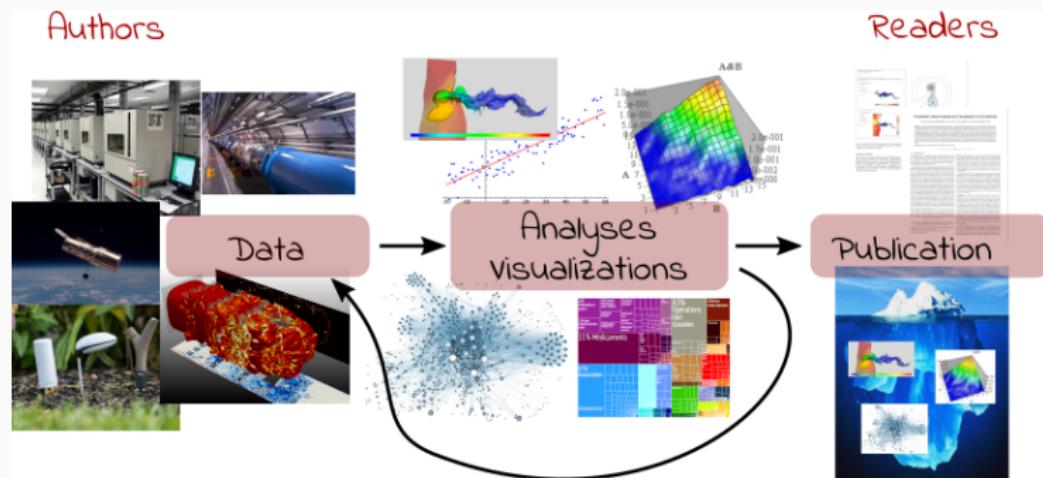


DIFFERENT REPRODUCIBILITY CONCERN

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical issues

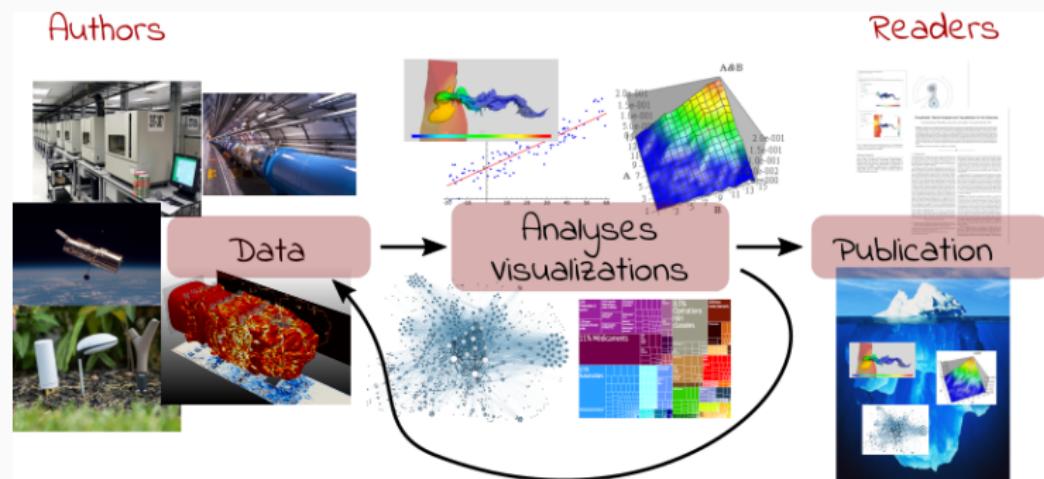


DIFFERENT REPRODUCIBILITY CONCERN

Social Sciences, Oncology methodology, statistics (p-hacking, HARKing)

Genomics software engineering, computational reproducibility, provenance

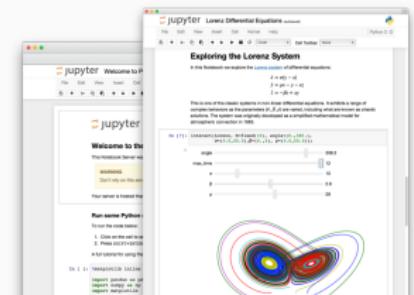
Computational fluid dynamics numerical issues



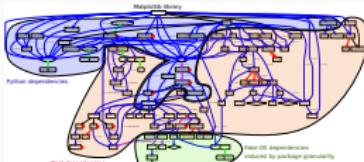
Reproducible Research = Bridging the Gap by working Transparently

EXISTING TOOLS, EMERGING STANDARDS

Notebooks and workflows



Software environments



Sharing platforms



TOOL 1: COMPUTATIONAL NOTEBOOKS/LITTERATE PROGRAMMING

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

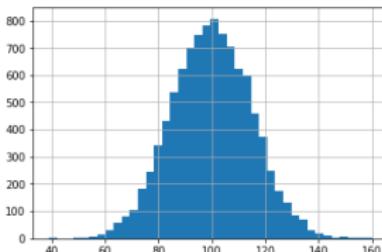
3.141592653589793

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

The screenshot shows a Jupyter Notebook interface with the following content:

```
# Un document computationnel
Mon ordinateur m'indique que $pi$ vaut "approximativement"

In [1]:
from math import *
print(pi)
3.141592653589793

Mais calculé avec la méthode des aiguilles de Buffon (https://fr.wikipedia.org/wiki/Aiguille\_de\_Buffon), on obtientrait comme approximation :
Out[1]: 3.1437198694998765

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 $\pi$, avec $y=1$ (si ce n'est une constante de normalisation...).

In [3]:
%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()
```

A histogram is displayed at the bottom, showing a bell-shaped curve centered around 100, with x-axis values from 40 to 160 and y-axis values from 0 to 800.

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

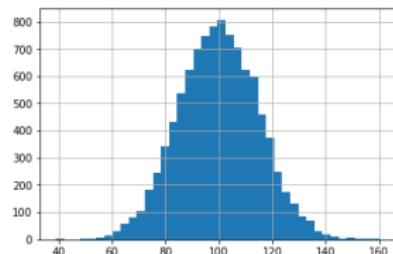
3.141592653589793

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
```

3.1437198694998765

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

```
# Un document computationnel
Mon ordinateur m'indique que $\pi$ vaut "approximativement"
3.141592653589793

Mais calculé avec la méthode des aiguilles de Buffon (https://fr.wikipedia.org/wiki/Aiguille\_de\_Buffon), on obtient aussi comme approximation :
3.14371986944998765

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

3.14371986944998765
```

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

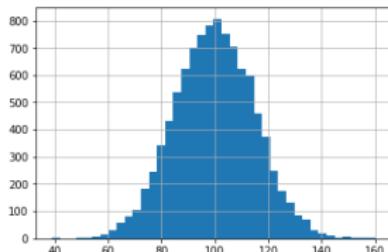
3.141592653589793

Mais calculé avec la [méthode des aiguilles de Buffon](#), on obtient 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=np.pi/2)
2/(sum((x+np.sin(theta))>1))/N
```

3.14371986944998765

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

The screenshot shows a Jupyter Notebook interface with three code cells:

- In [1]:** A single-line print statement that outputs the value of pi.
- In [2]:** A more complex script that includes imports for numpy and random, generates uniform random numbers for x and theta, and calculates an approximation of pi using theBuffon's needle method. It also includes a note about the formula used.
- In [3]:** A histogram plot of 100,000 random numbers, showing a bell-shaped distribution centered around 100.

Annotations with red arrows point from the text "Code" and "Un document computationnel" to the code cells in the notebook.

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

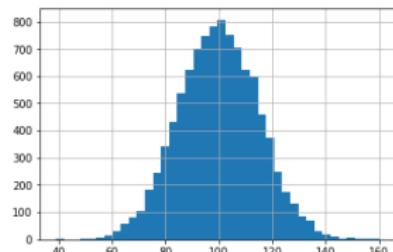
3.141592653589793

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
```

3.14371986949098765

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

Un document computationnel

```
In [1]:  
from math import *  
print(pi)  
3,141592653589793
```

Mais calculé avec la `_methodes_ des éimpulles de Buffon` (https://fr.wikipedia.org/wiki/Algille_de_Buffon), on obtiendrait comme `approximation` :

```
In [2]:  
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

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`, avec `matplotlib` (si ce n'est une constante de normalisation... ☺).

```
In [3]:  
%matplotlib inline  
import matplotlib.pyplot as plt  
  
mu, sigma = 100, 15  
x = mu + sigma*np.random.randn(10000)  
  
plt.hist(x, 99)  
plt.grid(True)  
plt.show()
```

Résultats

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

3.141592653589793

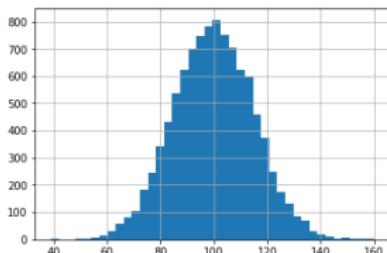
Mais calculé avec la **méthode des aiguilles de Buffon**, on obtiendrait comme approximation :

```
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

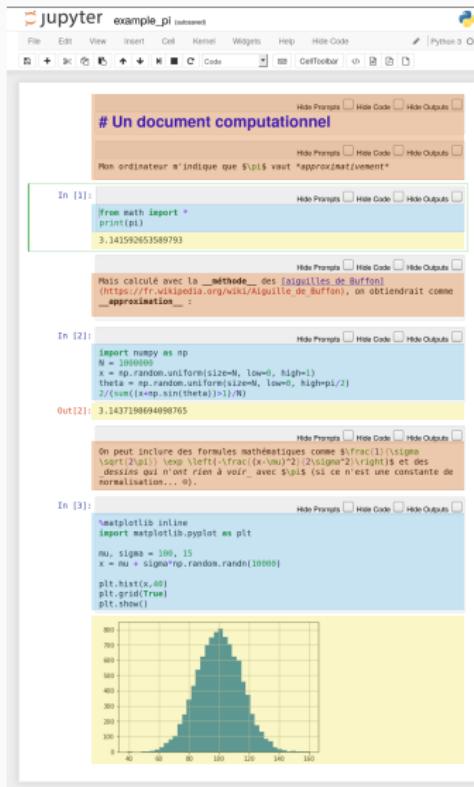
On peut inclure des formules mathématiques comme $\frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$ et

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



```
# Un document computationnel

In [1]: from math import * print(pi)
3.141592653589793

Out[1]: 3.141592653589793

Mais calculé avec la __methodes__ des éimpulles de Buffon
(https://fr.wikipedia.org/wiki/Algille\_de\_Buffon), on obtiendrait comme
__approximation__ :

In [2]: 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

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]: %matplotlib inline
import matplotlib.pyplot as plt
mu, sigma = 100, 15
x = mu + sigma*np.random.randn(100000)
plt.hist(x,99)
plt.grid(True)
plt.show()
```

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

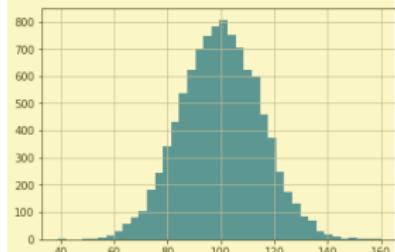
3.141592653589793

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... ☺).



Export
→

TOOL 1: COMPUTATIONAL NOTEBOOKS/LITTERATE PROGRAMMING

Document initial dans son environnement

The screenshot shows a Jupyter Notebook interface with three code cells:

- In [1]:** Prints the value of pi (3.141592653589793) and includes a note about calculating pi with Buffon's needle method.
- In [2]:** Generates random points (x, theta) and calculates an approximation of pi based on the ratio of points where sin(theta) >= x.
- In [3]:** Plots a histogram of x values, showing a bell-shaped distribution centered around 100.



Document final

Un document computationnel

Mon ordinateur m'indique que π vaut approximativement

3.141592653589793

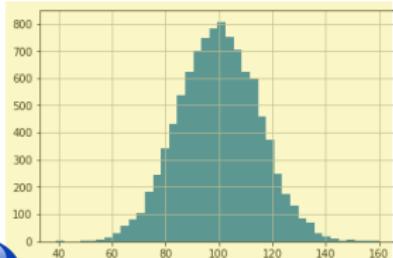
Mais calculé avec la méthode des [aiguilles de Buffon](#), on obtiendrait comme approximation :

```
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

Export

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 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

What is hiding behind a simple

```
1 import matplotlib
```

```
Package: python3-matplotlib
Version: 2.1.1-2
Depends: python3-dateutil, python-matplotlib-data (>= 2.1.1-2),
python3-pyparsing (>= 1.5.6), python3-six (>= 1.10), python3-tz,
libjs-jquery, libjs-jquery-ui, python3-numpy (>= 1:1.13.1),
python3-numpy-abi9, python3 (<< 3.7), python3 (>= 3.6~),
python3-cycler (>= 0.10.0), python3:any (>= 3.3.2-2~), libc6 (>=
2.14), libfreetype6 (>= 2.2.1), libgcc1 (>= 1:3.0), libpng16-16 (>=
1.6.2-1), libstdc++6 (>= 5.2), zlib1g (>= 1:1.1.4)
```

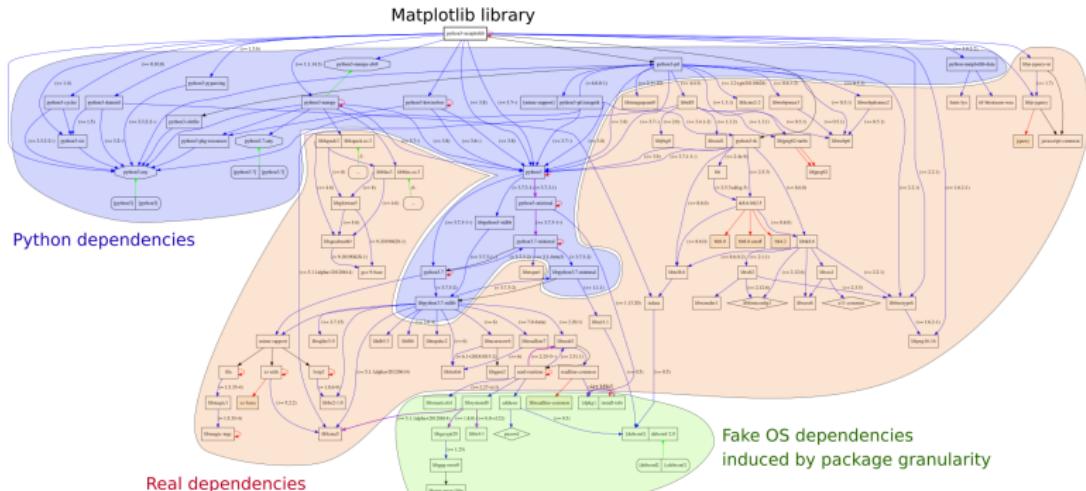
TOOL 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

What is hiding behind a simple

1

```
import matplotlib
```

Package: python3-matplotlib



TOOL 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

Python and its rapidly evolving environment

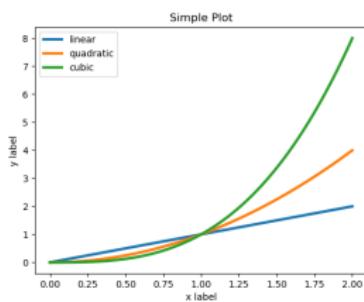
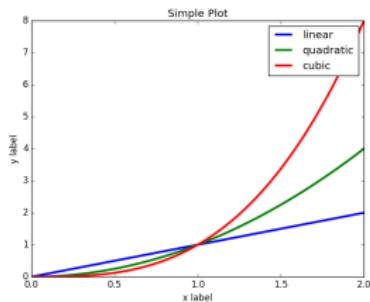
```
1 python2 -c "print(10/3)"  
2 python3 -c "print(10/3)"
```

```
3  
3.333333333333335
```

TOOL 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

Python and its rapidly evolving environment

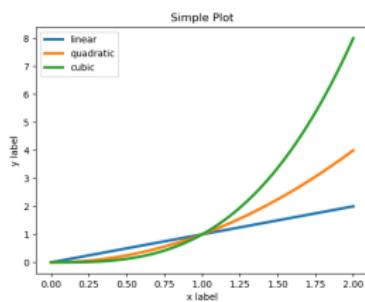
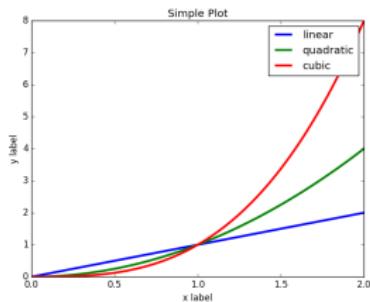
```
1 python2 -c "print(10/3)"  
2 python3 -c "print(10/3)"
```



TOOL 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

Python and its rapidly evolving environment

```
1 python2 -c "print(10/3)"  
2 python3 -c "print(10/3)"
```

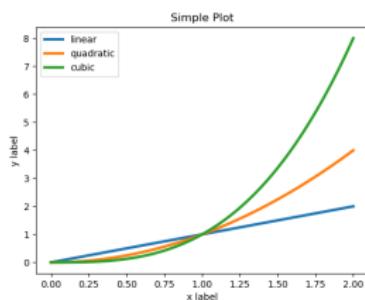
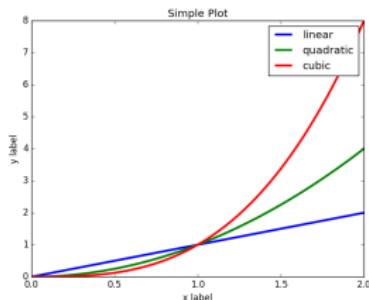


Cortical Thickness Measurements (PLOS ONE, June 2012): *FreeSurfer: differences were found between the Mac and HP workstations and between Mac OSX 10.5 and OSX 10.6.*

TOOL 2: FIGHTING SOFTWARE ENVIRONMENTS NIGHTMARE

Python and its rapidly evolving environment

```
1 python2 -c "print(10/3)"  
2 python3 -c "print(10/3)"
```



Cortical Thickness Measurements (PLOS ONE, June 2012): *FreeSurfer: differences were found between the Mac and HP workstations and between Mac OSX 10.5 and OSX 10.6.*



TOOL 3: FIGHTING INFORMATION LOSS WITH ARCHIVES

D. Spinellis. **The Decay and Failures of URL References.** CACM, 46(1), Jan 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

TOOL 3: FIGHTING INFORMATION LOSS WITH ARCHIVES

D. Spinellis. The Decay and Failures of URL References. CACM, 46(1), Jan 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



CHANGING RESEARCH PRACTICES

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



- Book on RR *Vers une recherche reproductible: Faire évoluer ses pratiques*
- MOOC on RR (3rd edition Feb. 2020)
- A new "Advanced RR" MOOC (Oct. 2020)
 - Software environment control (Docker)
 - Scientific workflow (snakemake)
 - Managing data (HDF5, archiving)

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

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...

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

REPRODUCIBLE RESEARCH = RIGOR AND TRANSPARENCY

To err is human: RR will not prevent errors nor fraud. It will help though

Good research requires time and resources

1. Train yourself and your students: RR, statistics, experiments
 - Beware of checklists and norms
 - Understand what's at stake
2. Change the norm: make publication practices evolve
3. Incentive: consider RR/open science when hiring/promoting



PERSPECTIVES AND OPPORTUNITIES

Transparency is not new. Interoperability and Reusability is newer

- This leads to fast and tremendous progress (e.g., Deep Learning)
- How Reusable is really Data and Software ?
 - Often specific, often impossible to customize despite openness
- Need for better *norms* ... but beware of brutal transitions
 - Acknowledge different kind of contributions (novelty vs. quality)
 - Independent experts and certification agencies
 - Embrace uncertainty



PERSPECTIVES AND OPPORTUNITIES

Transparency is not new. Interoperability and Reusability is newer

- This leads to fast and tremendous progress (e.g., Deep Learning)
- How Reusable is really Data and Software ?
 - Often specific, often impossible to customize despite openness
- Need for better *norms* ... but beware of brutal transitions
 - Acknowledge different kind of contributions (novelty vs. quality)
 - Independent experts and certification agencies
 - Embrace uncertainty



Lack of reproducibility is mostly linked to statistics and computers

- Computer Science has no well-defined empirical practices yet
 - What can we expect from an experiment in terms of precision/representativeness/perenniality ?
- Prepare the Future: Toward literate experimentation?
 - Reuse, reuse, reuse!
 - Shared and controled testbeds (e.g., Grid'5000/SILECS)
 - How to share Experiments ?