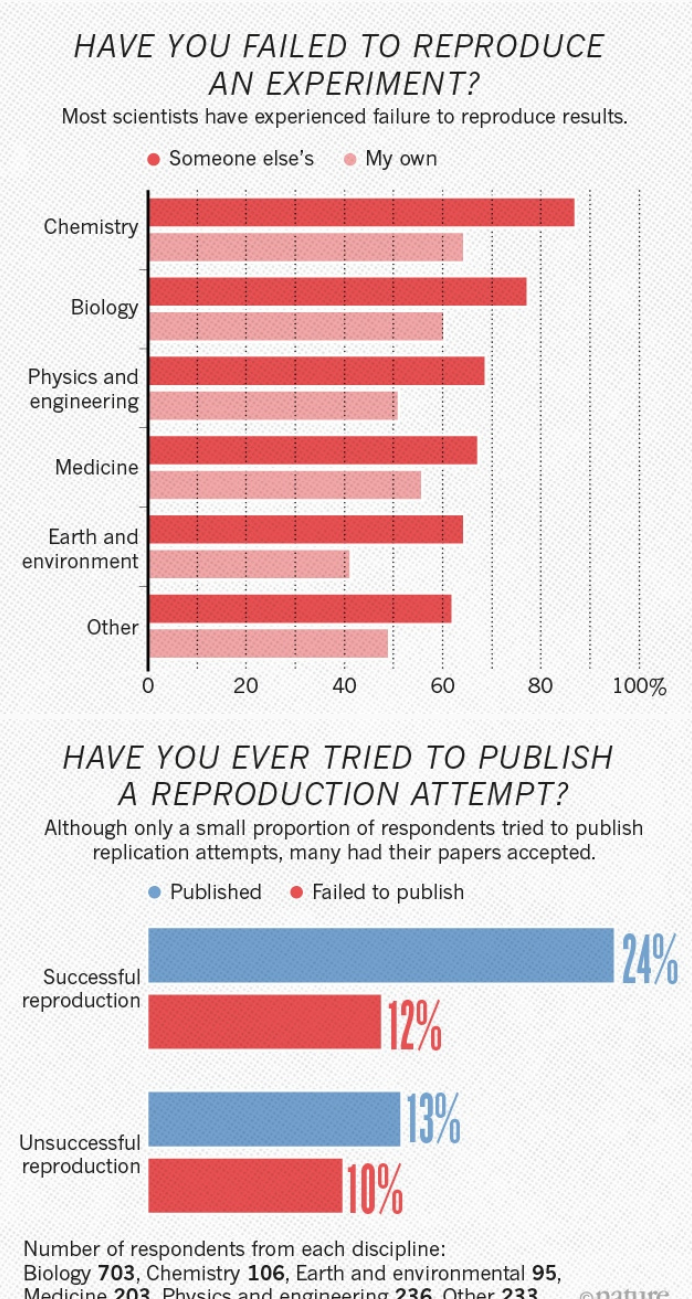
Laboratory notebook, computational document, reproducible article
Emacs/Org-mode: One ring to rule them all?

Arnaud Legrand

MaiMosine, GRICAD, SARI network, June 2023

Reproducibility "Crisis": Socio-technical Challenges



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

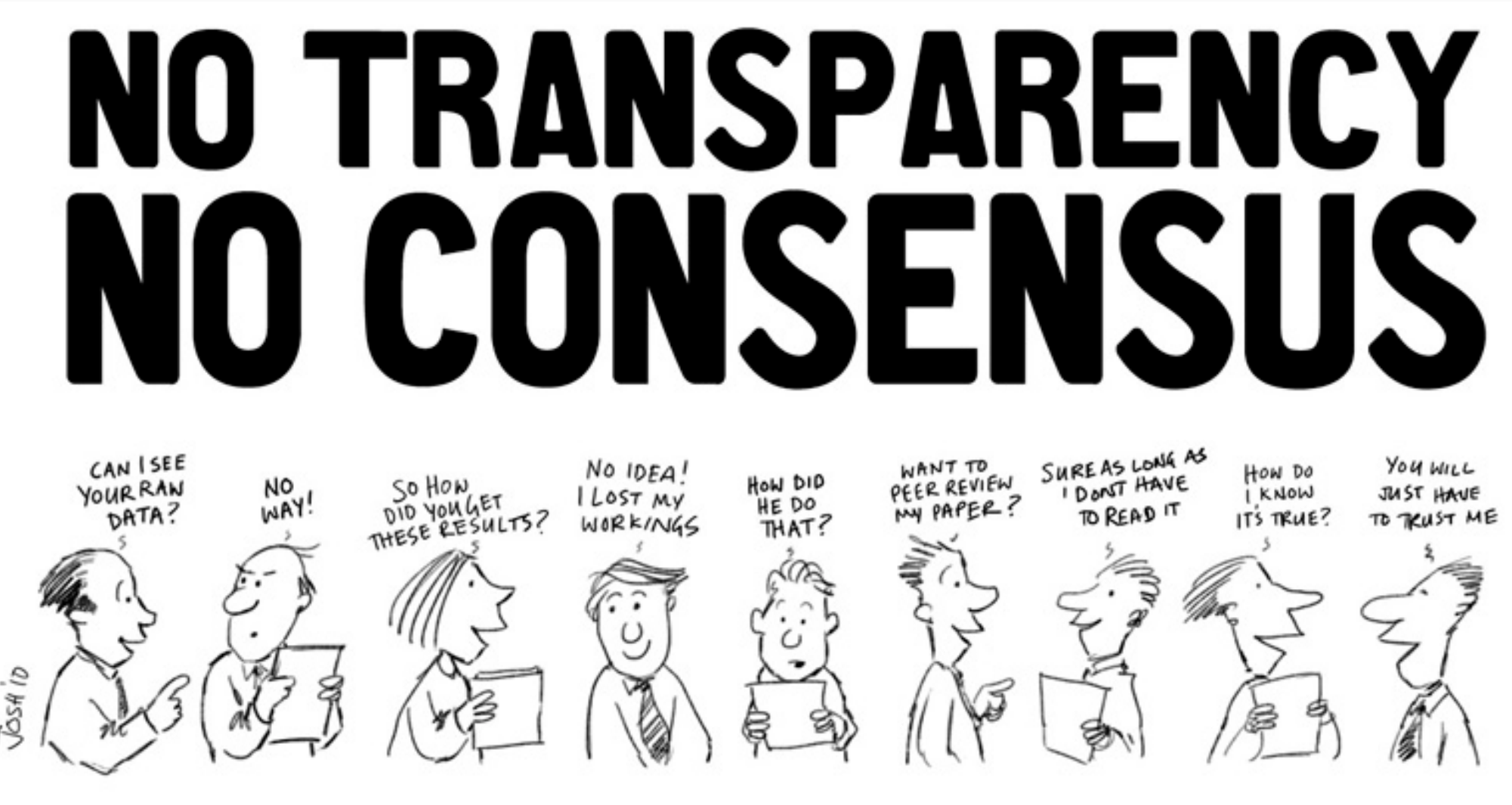
Social causes

- 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: 1M+ articles per year!
- Emerging practices: DORA/Plan S/COARA, DMP and FAIR data, artefact evaluation, reproducibility badges, reproducibility challenges, open reviews, ...

Methodological/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

Biology, Oncology sample provenance, clinical trials $\rightsquigarrow$ standardized protocols

Psychology, Nutrition HARKING, p-hacking $\rightsquigarrow$ pre-registration

Different Reproducibility Concerns in Modern Science

Biology, Oncology sample provenance, clinical trials $\rightsquigarrow$ standardized protocols

Psychology, Nutrition HARKING, p-hacking $\rightsquigarrow$ pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical chaos, parallel architectures

Different Reproducibility Concerns in Modern Science

Biology, Oncology sample provenance, clinical trials ⇝ standardized protocols

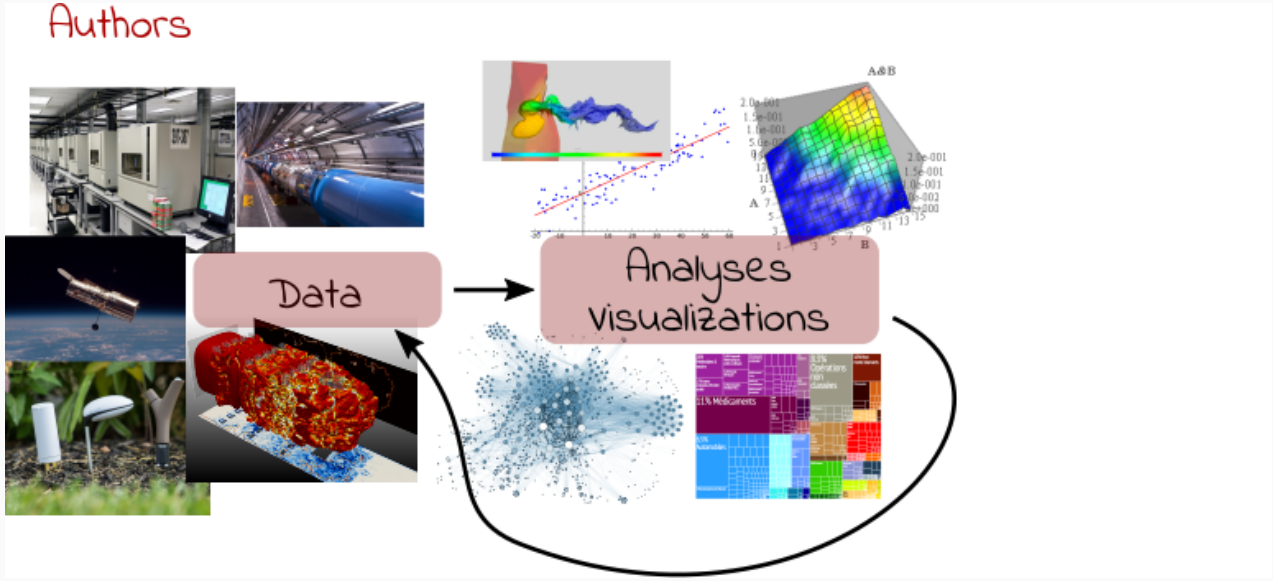
Psychology, Nutrition HARKING, p-hacking ⇝ pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical chaos, parallel architectures

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

Biology, Oncology sample provenance, clinical trials ⇝ standardized protocols

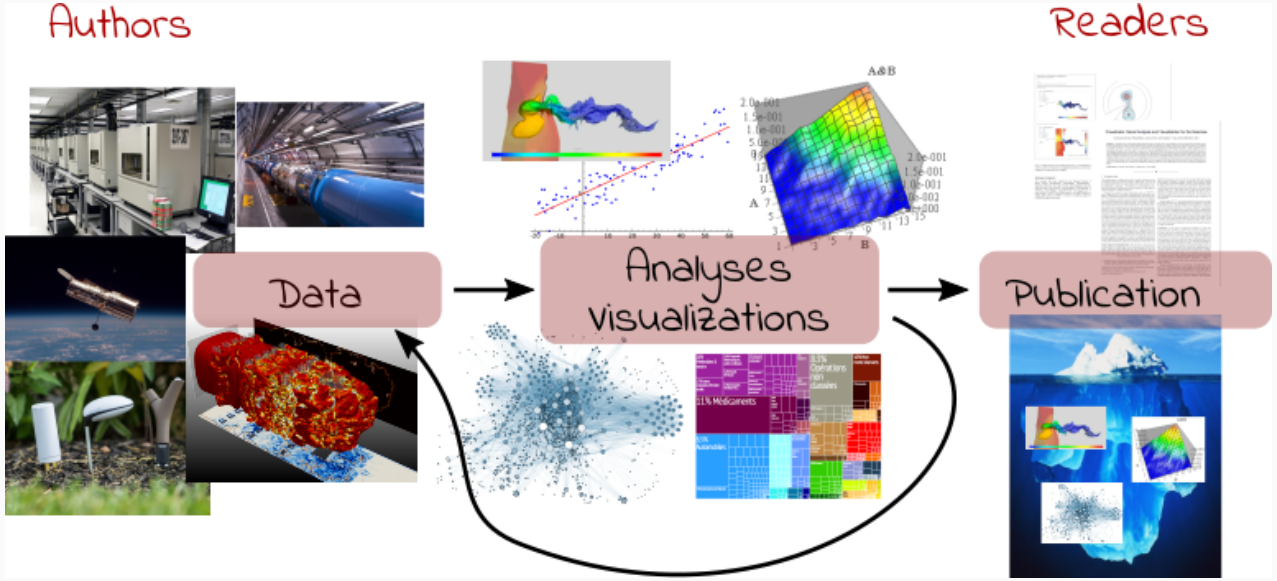
Psychology, Nutrition HARKING, p-hacking ⇝ pre-registration

Genomics software engineering, computational reproducibility, provenance

Computational fluid dynamics numerical chaos, parallel architectures

Artificial Intelligence most of the above 😉

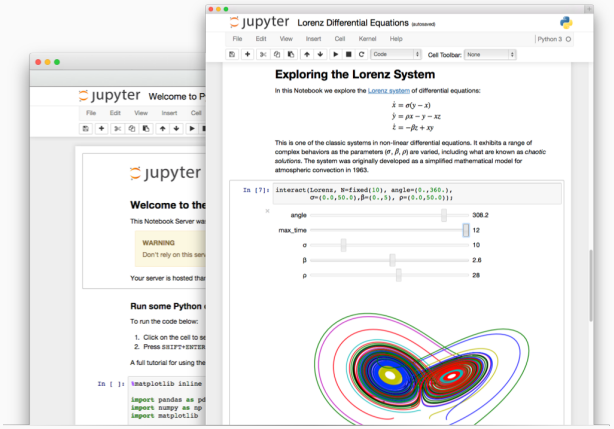
The processing steps between raw observations and findings have gotten increasingly numerous and complex



Reproducible Research = Bridging the Gap by working Transparently

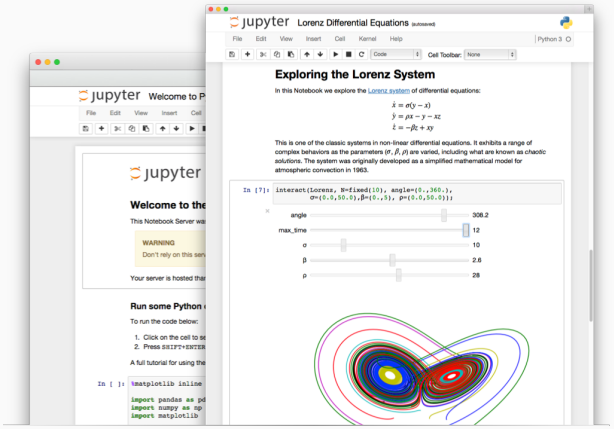
Reproducibility Issues Related to the use of Computers

Computation provenance: notebooks and workflows

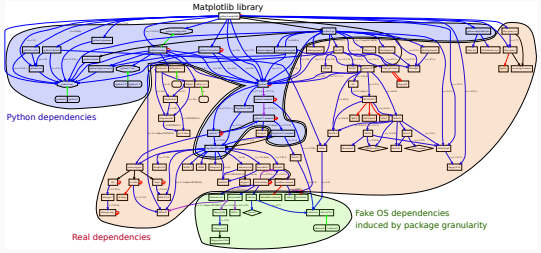


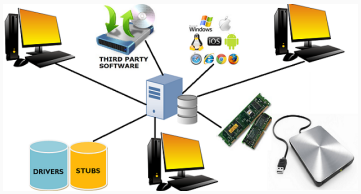
Reproducibility Issues Related to the use of Computers

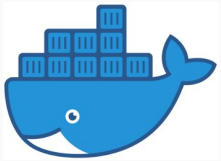
Computation provenance: notebooks and workflows



Software environments

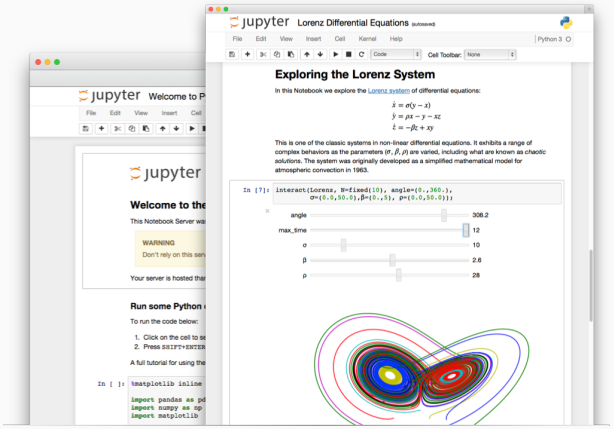




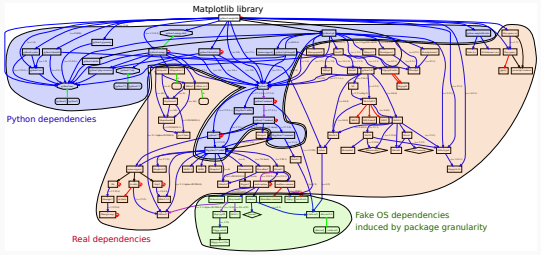


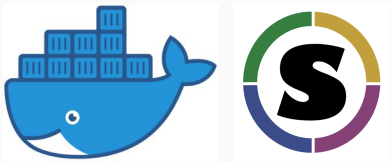
Reproducibility Issues Related to the use of Computers

Computation provenance: notebooks and workflows

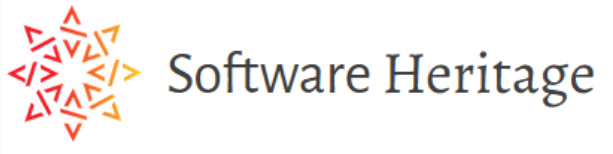


Software environments





Sharing and Archiving



Good Practice #1

Taking Notes and Documenting

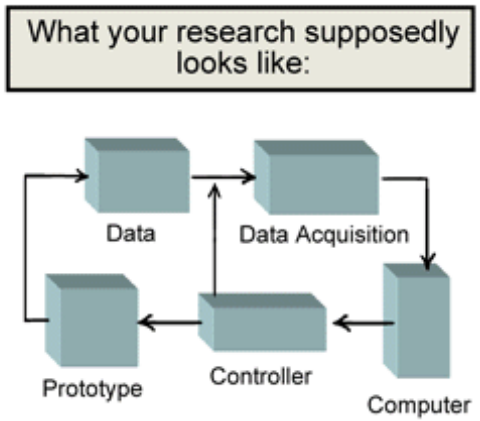


Figure 1. Experimental Diagram

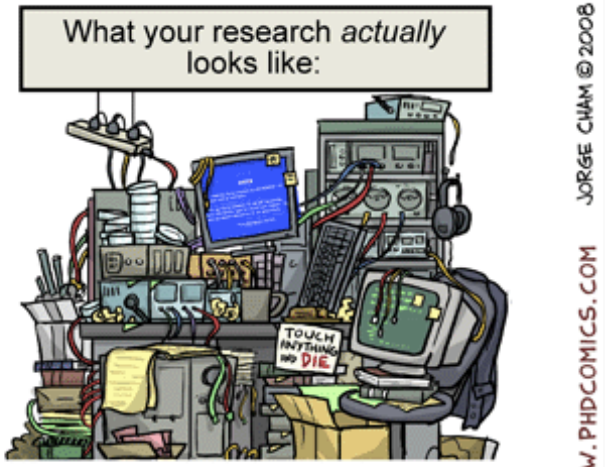
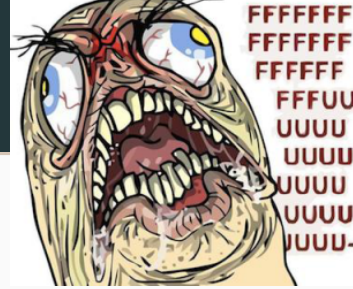


Figure 2. Experimental Mess

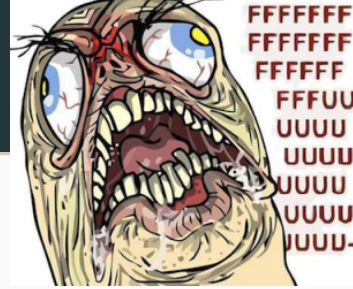
Frustration as an Author/Reviewer



Author (Calls for a *Journal*)

- 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?

Frustration as an Author/Reviewer



Author (Calls for a *Journal*)

- 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 (Calls for a *Reproducible Article*)

- 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/Literate Programming

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

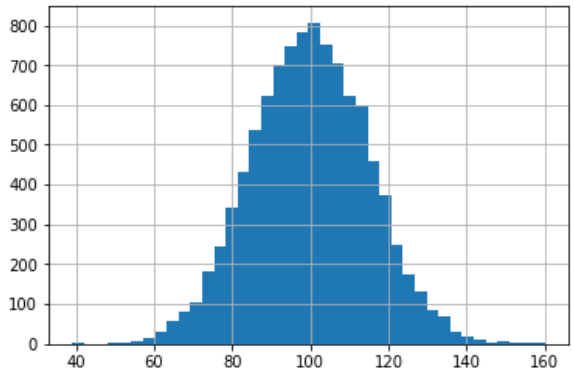
```
3.141592653589793
```

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/Literate 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... ☺).
```

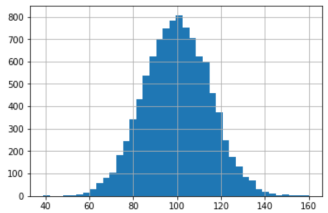
In [3]:

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*

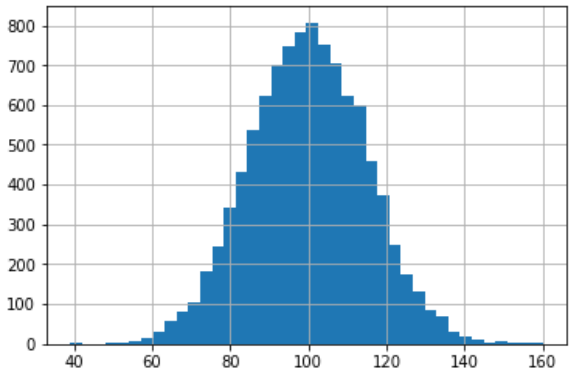
```
3.141592653589793
```

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/Literate 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... ☺).
```

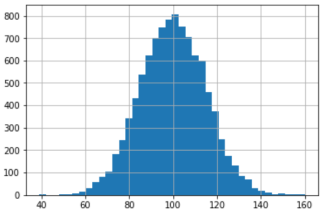
In [3]:

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()
```



MarkDown

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

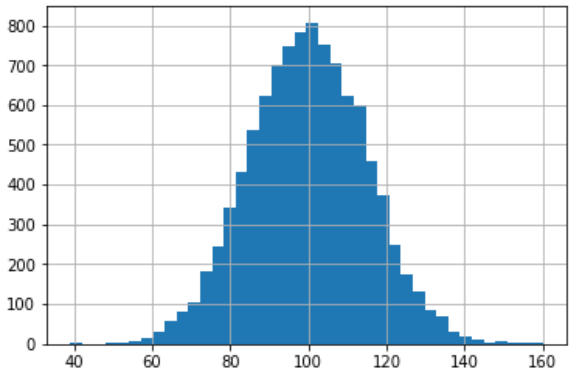
```
3.141592653589793
```

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/Literate 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... ☺).
```

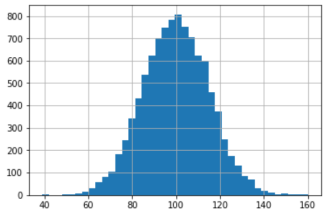
In [3]:

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()
```



Code

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

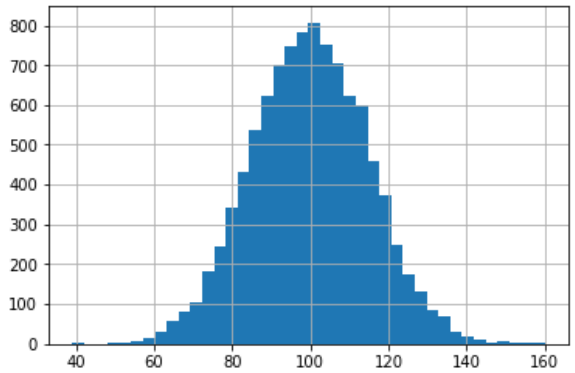
```
3.141592653589793
```

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/Literate 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... ☺).
```

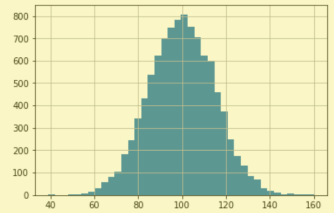
In [3]:

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()
```



Résultats

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

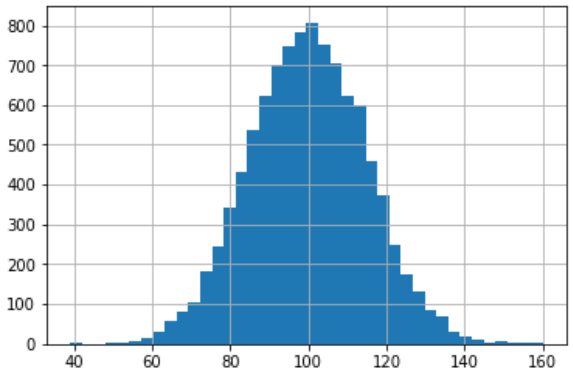
```
3.141592653589793
```

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)
```

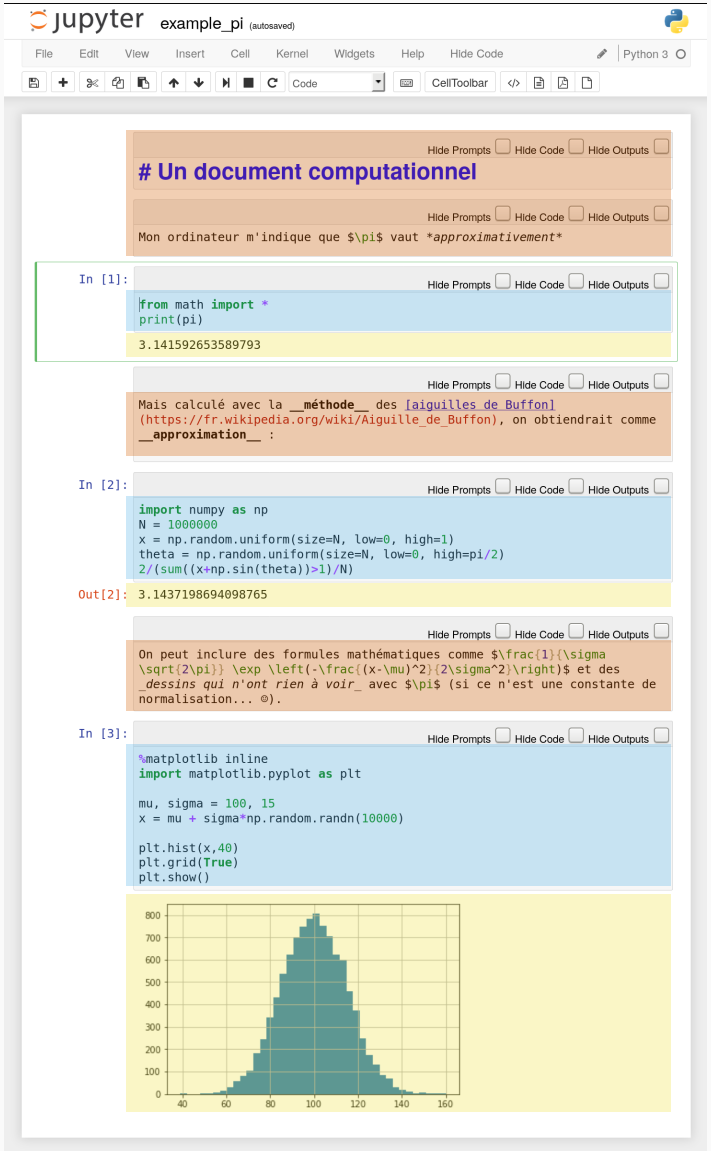
```
3.1437198694098765
```

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/Literate Programming

Document initial dans son environnement



Export

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

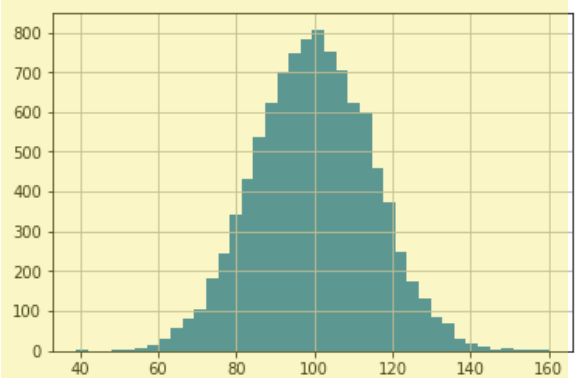
```
3.141592653589793
```

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/Literate 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... ☺).
```

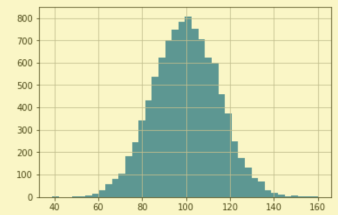
In [3]:

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*

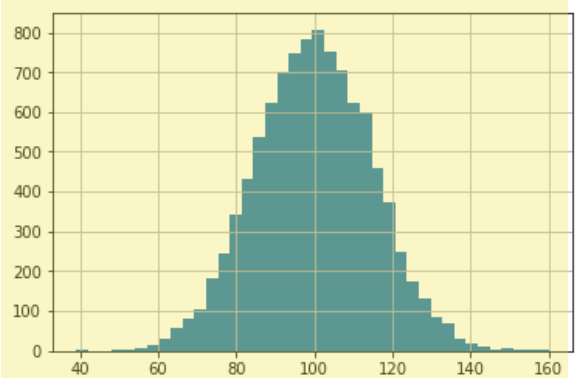
```
3.141592653589793
```

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... ☺).



Several possible usages

Code Documentation ($\neq$ code comments)

- For developers: explain code organization (data structures, algorithms, modules, class, etc.) and how to contribute
- For users: API, examples, installation, ...

Several possible usages

Code Documentation ($\neq$ code comments)

- For developers: explain code organization (data structures, algorithms, modules, class, etc.) and how to contribute
- For users: API, examples, installation, ...

Data Analysis data dependent

- Data transformation/curation
- Exploratory/sequential data analysis
- Document hypothesis, graphs/tables

Several possible usages

Code Documentation ($\neq$ code comments)

- For developers: explain code organization (data structures, algorithms, modules, class, etc.) and how to contribute
- For users: API, examples, installation, ...

Data Analysis data dependent

- Data transformation/curation
- Exploratory/sequential data analysis
- Document hypothesis, graphs/tables

Reproducible article (book with code/computations)

- Classical article/book structure with typography constraints
- Code is behind the scene (e.g., figures, tables, numbers)

Several possible usages

Code Documentation ($\neq$ code comments)

- For developers: explain code organization (data structures, algorithms, modules, class, etc.) and how to contribute
- For users: API, examples, installation, ...

Data Analysis data dependent

- Data transformation/curation
- Exploratory/sequential data analysis
- Document hypothesis, graphs/tables

Reproducible article (book with code/computations)

- Classical article/book structure with typography constraints
- Code is behind the scene (e.g., figures, tables, numbers)

"Laboratory" notebook Research reporting. Daily notes on

- Experimental parameters, specific configurations
- Meetings, seminars, lectures, ...
- Readings, bibliography
- TODOs, Ideas, Random hacks

Several possible usages

Code Documentation ($\neq$ code comments)

- For developers: explain code organization (data structures, algorithms, modules, class, etc.) and how to contribute
- For users: API, examples, installation, ...

Data Analysis data dependent

- Data transformation/curation
- Exploratory/sequential data analysis
- Document hypothesis, graphs/tables

Reproducible article (book with code/computations)

- Classical article/book structure with typography constraints
- Code is behind the scene (e.g., figures, tables, numbers)

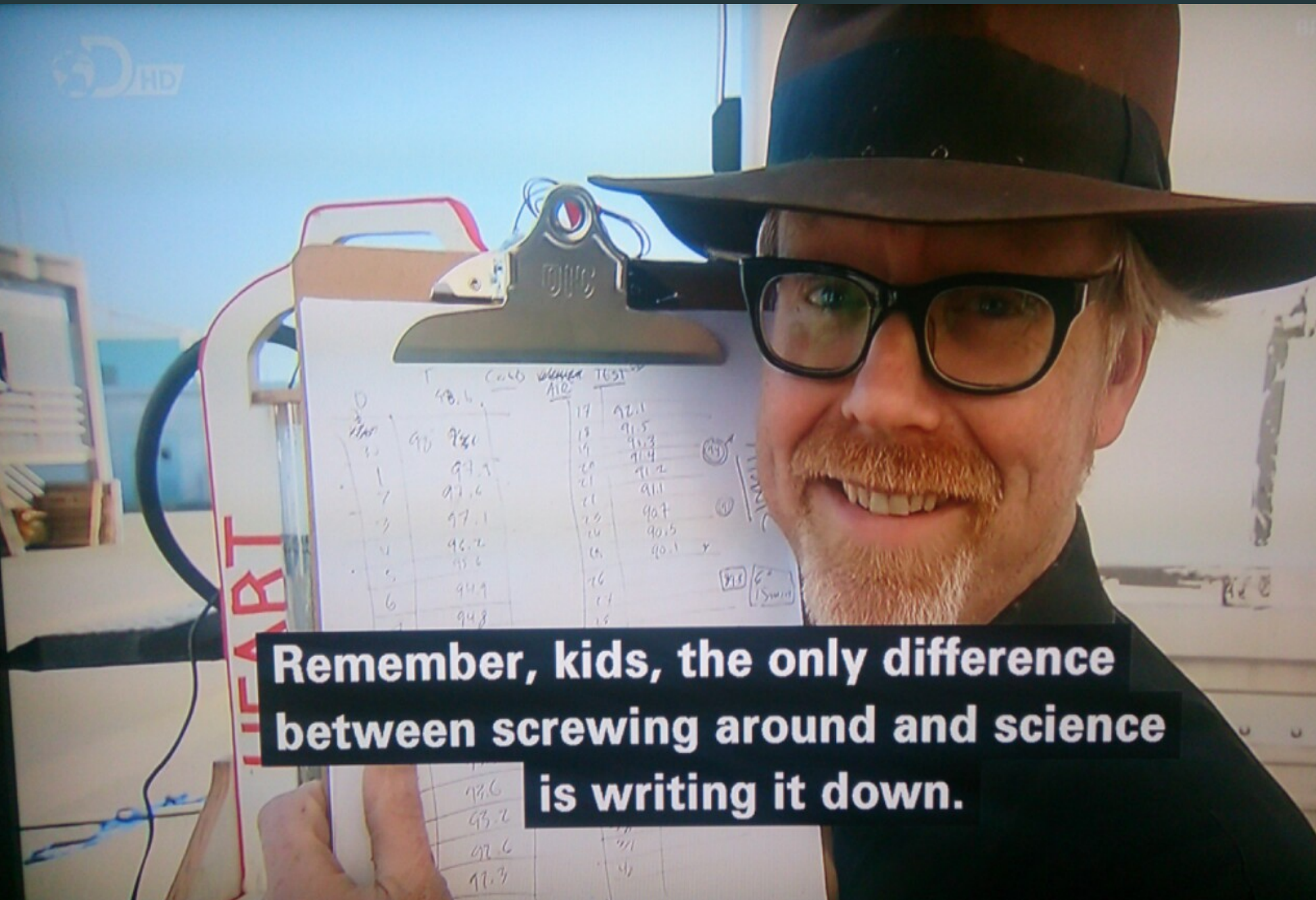
"Laboratory" notebook Research reporting. Daily notes on

- Experimental parameters, specific configurations
- Meetings, seminars, lectures, ...
- Readings, bibliography
- TODOs, Ideas, Random hacks

Teaching material, Presentations

Dynamic documents, Websites, ...

Laboratory Notebooks, Computational Documents



Quick demo of Jupyter, Rstudio, and Org-Mode

(python)

Pros Python/R, friendly, portable (web browser, client/server)

Cons
- Installation, software dependencies (minimal-notebook $\approx$ 440Mb)
- Limited control on typography (unless using *Rube Goldberg* machines like https://quarto.org/)

Quick demo of Jupyter, Rstudio, and Org-Mode

(python)

Pros Python/R, friendly, portable (web browser, client/server)

Cons
- Installation, software dependencies (minimal-notebook ≈ 440Mb)
- Limited control on typography (unless using *Rube Goldberg* machines like https://quarto.org/)

(java/R)

Pros R/Python, friendly, portable, real IDE, Good typography control

Cons
- Installation, software dependencies (rocker/rstudio ≈ 550Mb)
- Limited control on typography (unless using monsters like quarto)

Quick demo of Jupyter, Rstudio, and Org-Mode

(python)

Pros Python/R, friendly, portable (web browser, client/server)

Cons
- Installation, software dependencies (minimal-notebook ≈ 440Mb)
- Limited control on typography (unless using *Rube Goldberg* machines like https://quarto.org/)

(java/R)

Pros R/Python, friendly, portable, real IDE, Good typography control

Cons
- Installation, software dependencies (rocker/rstudio ≈ 550Mb)
- Limited control on typography (unless using monsters like quarto)

(emacs-lisp)

Pros R, Python, Perl, Ruby, C, Java, ...
- Pure text, Good typography control, "Lightweight", Programmable editor

Cons
- No default configuration ⇝rough and steep learning curve
- Big machinery: silex/emacs-alpine-ci ≈ 240Mb even though flycheck/emacs-cask ≈ 80Mb

Which tool for which usage ?

A biased opinion

	Jupyter
Coding	
Data Analysis	
Articles	
Lab Notebook	
Slides	
Dynamic docs	
~~Websites~~	

Which tool for which usage ?

A biased opinion

	Jupyter	Rstudio
Coding		 (R)
Data Analysis		
Articles		
Lab Notebook		
Slides		
Dynamic docs		
~~Websites~~		

Which tool for which usage ?

A biased opinion

	Jupyter	Rstudio	Org-mode
Coding		 (R)	
Data Analysis			
Articles			
Lab Notebook			
Slides			
Dynamic docs			
~~Websites~~			

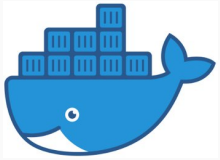
Pandora's box

Common Issues

1. Format evolution through time (increasing complexity)

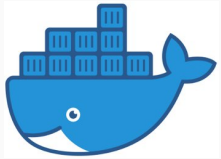
Common Issues

1. Format evolution through time (increasing complexity)
2. Environment control. Uuh ???



Common Issues

1. Format evolution through time (increasing complexity)
2. Environment control. Uuh ???



3. Scalability

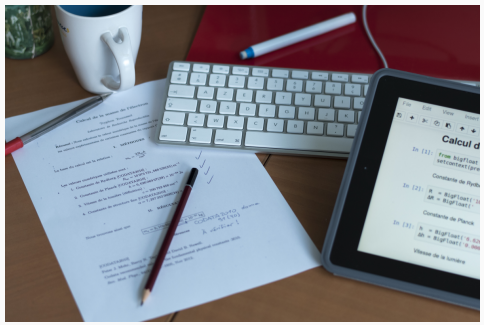
Reproducible Research = Rigor and Transparency

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

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

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

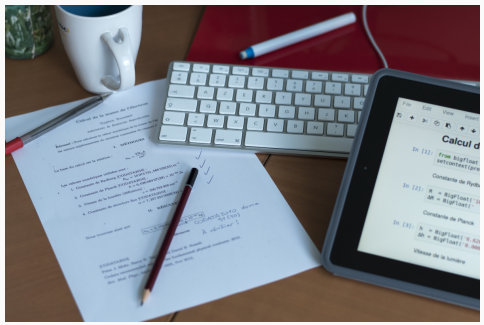


Reproducible Research = Rigor and Transparency

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

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



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

MOOC "Advanced RR" planned for Nov. 2023

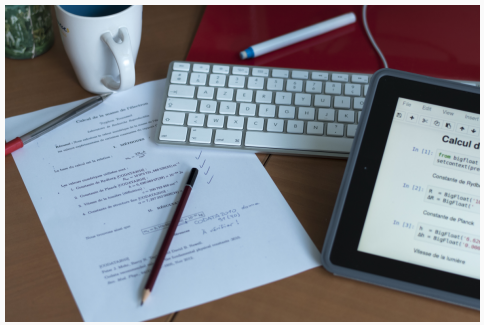
- Managing data (`FITS/HDF5`, `git annex`)
- Software environment control (`docker`, `singularity`, `guix`)
- Scientific workflow (`make`, `snakemake`)

Reproducible Research = Rigor and Transparency

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

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



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

MOOC "Advanced RR" planned for Nov. 2023

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

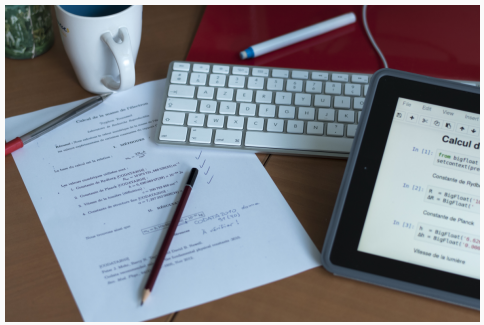
2. **Change the norm:** make publication practices evolve
3. **Incentive**: consider RR/open science when hiring/promoting

Reproducible Research = Rigor and Transparency

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

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



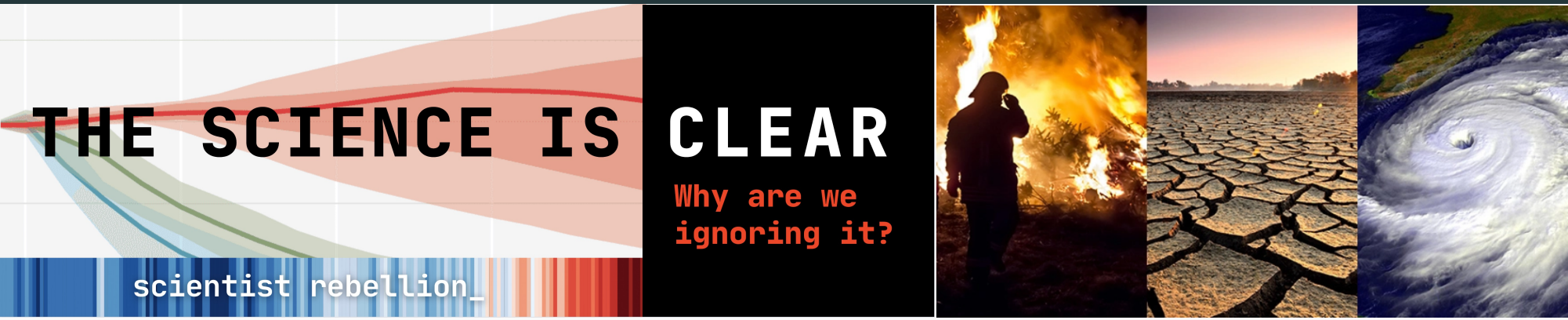
- Konrad Hinsen, Christophe Pouzat
- 3rd Edition: March 2020 – March 2023 (15,000+)

MOOC "Advanced RR" planned for Nov. 2023

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

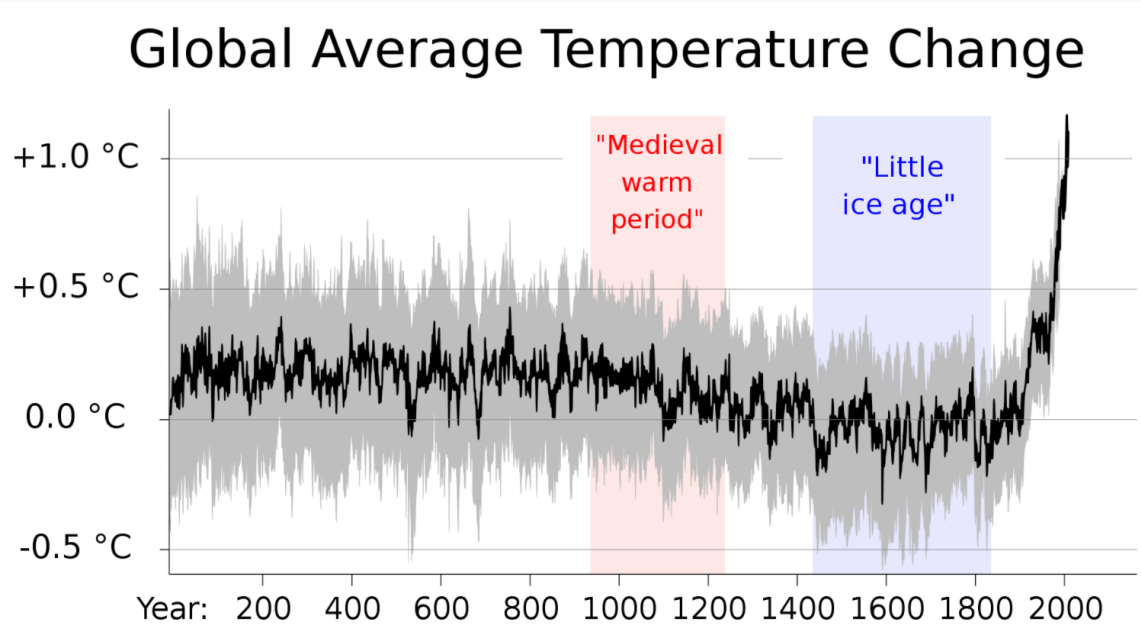
2. **Change the norm:** make publication practices evolve
3. **Incentive**: consider RR/open science when hiring/promoting
4. **Prepare the Future:**
   - Reuse, reuse, reuse!
   - Toward literate experimentation?
   - Shared and controled testbeds
   - How to share Experiments/Simulations ?

The Elephant in the Room: Climate Change

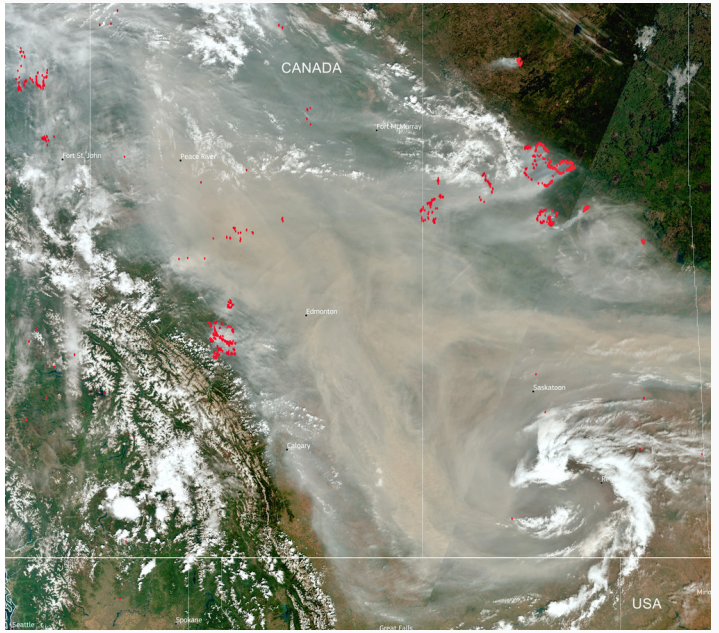


IPCC, IPBES, https://climate.nasa.gov/

1. Global climate change is **not** a future problem

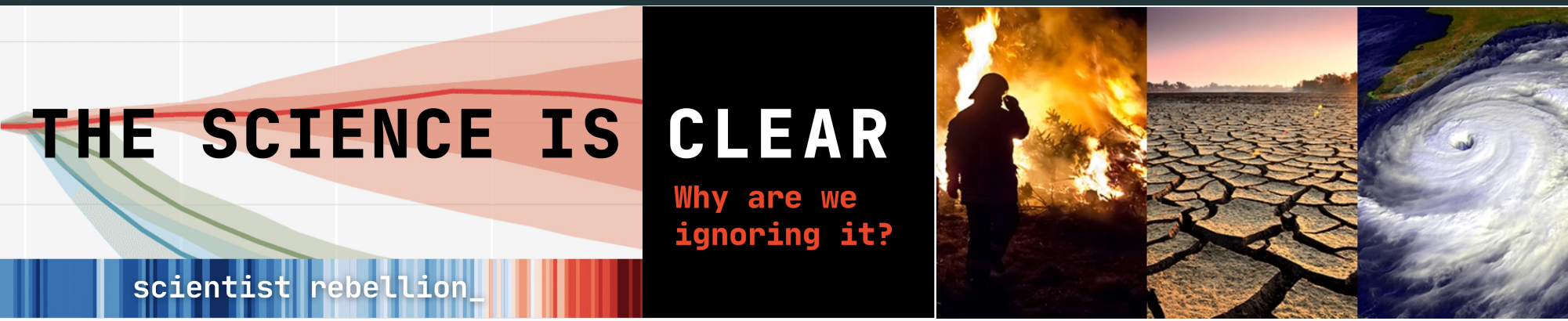


https://en.wikipedia.org/wiki/Global_temperature_record



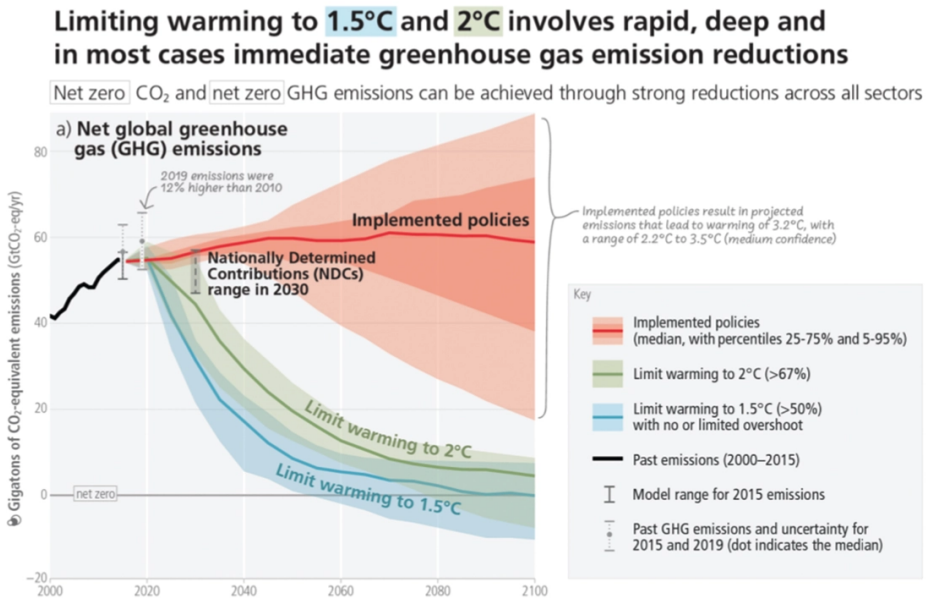
2023 Alberta wildfires (> 1 Mha)

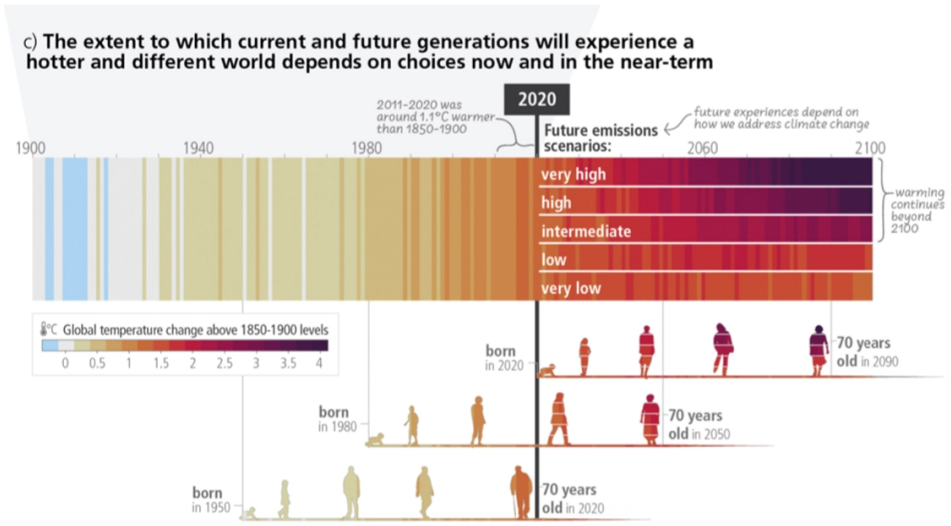
The Elephant in the Room: Climate Change



IPCC, IPBES, https://climate.nasa.gov/

1. Global climate change is **not** a future problem
2. It is **entirely** due to human activity

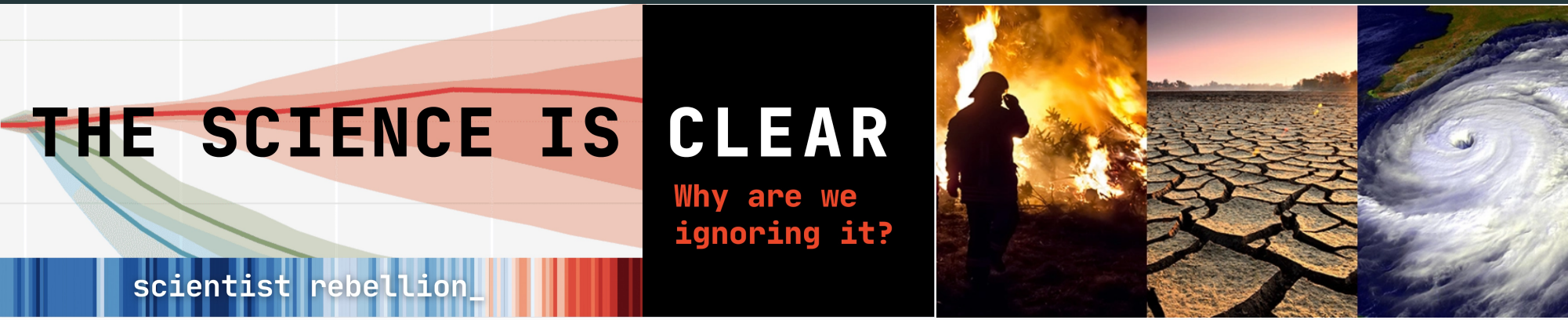




Paris Agreement'15 ∼ Net Zero by 2050

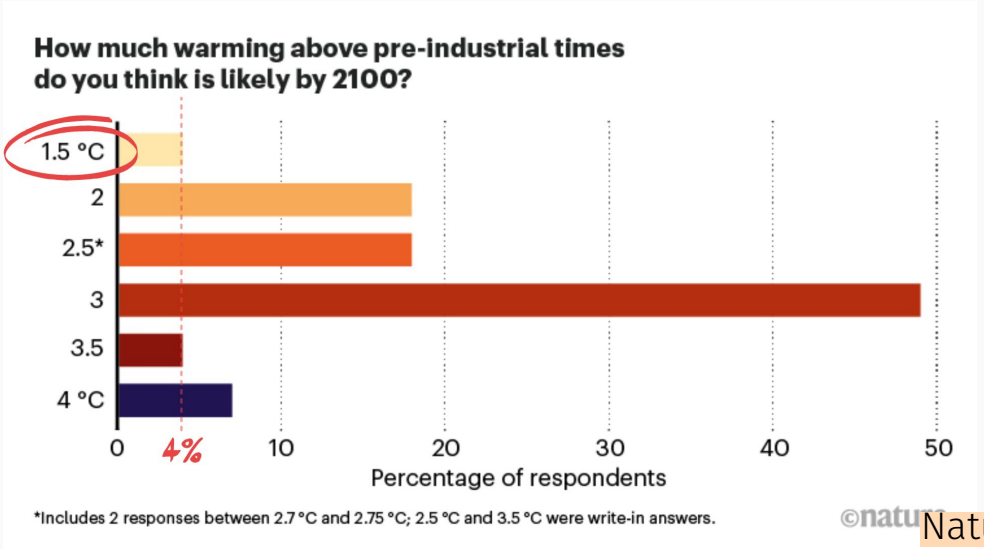
Latest IPCC report

The Elephant in the Room: Climate Change



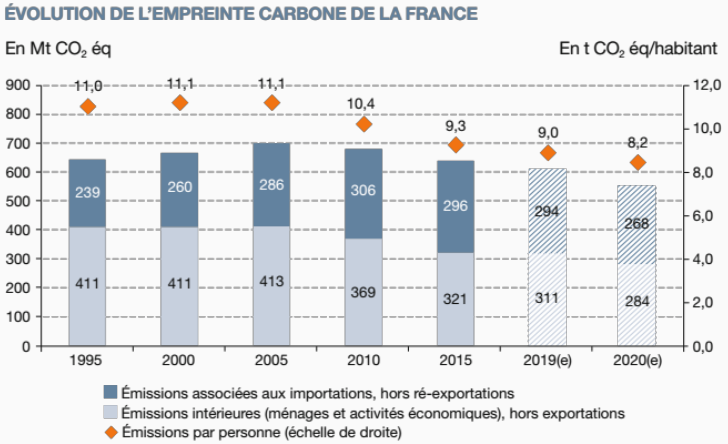
IPCC, IPBES, https://climate.nasa.gov/

1. Global climate change is **not** a future problem
2. It is **entirely** due to human activity
3. *9 out of 10 IPCC scientists believe overshoot is likely*

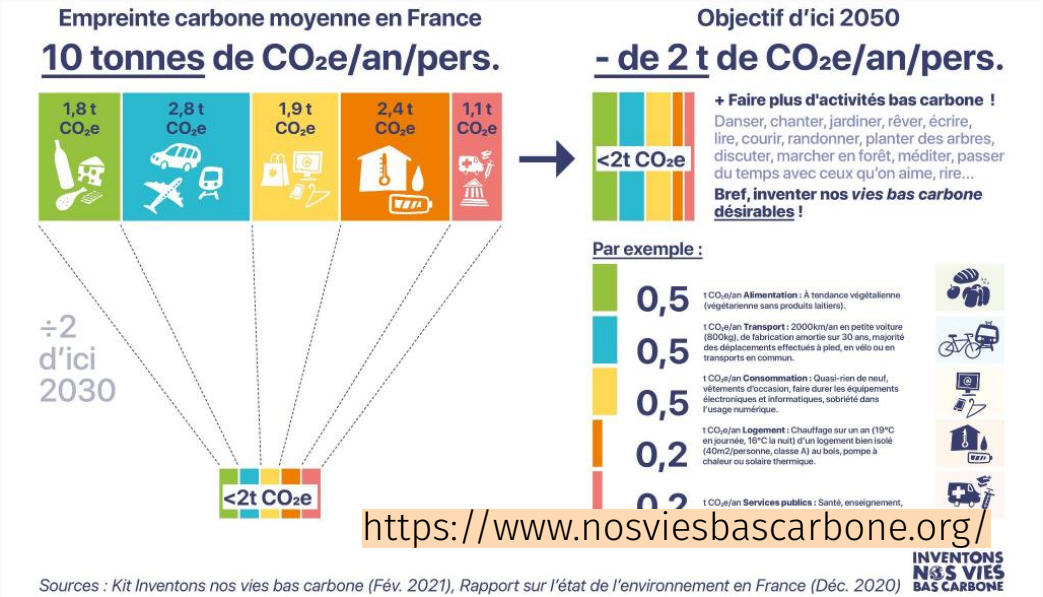


Nature survey, Nov. 2021

Put aside biodiversity loss, pollution, freshwater, land system change...



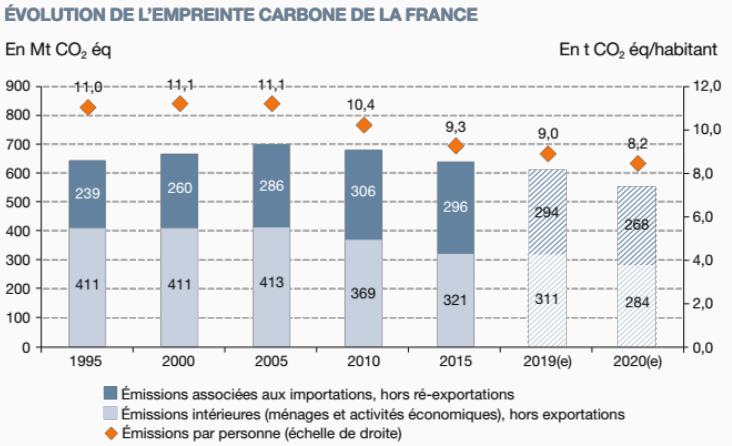
(e) = estimations.
Note : l'empreinte carbone porte sur les trois principaux gaz à effet de serre (CO_2, CH_4, N_2O). En 2021, la méthodologie a été ajustée afin de mieux tenir compte de l'évolution des coûts du pétrole brut, du gaz et du charbon ; l'ensemble de la série a ainsi été révisée, l'essentiel des ajustements portant sur les émissions importées de CH_4.
Champ : périmètre Kyoto (métropole et outre-mer appartenant à l'UE).
***Sources :** Citepa ; AIE ; FAO ; Douanes ; Eurostat ; Insee. Traitement : SDES, 2021*

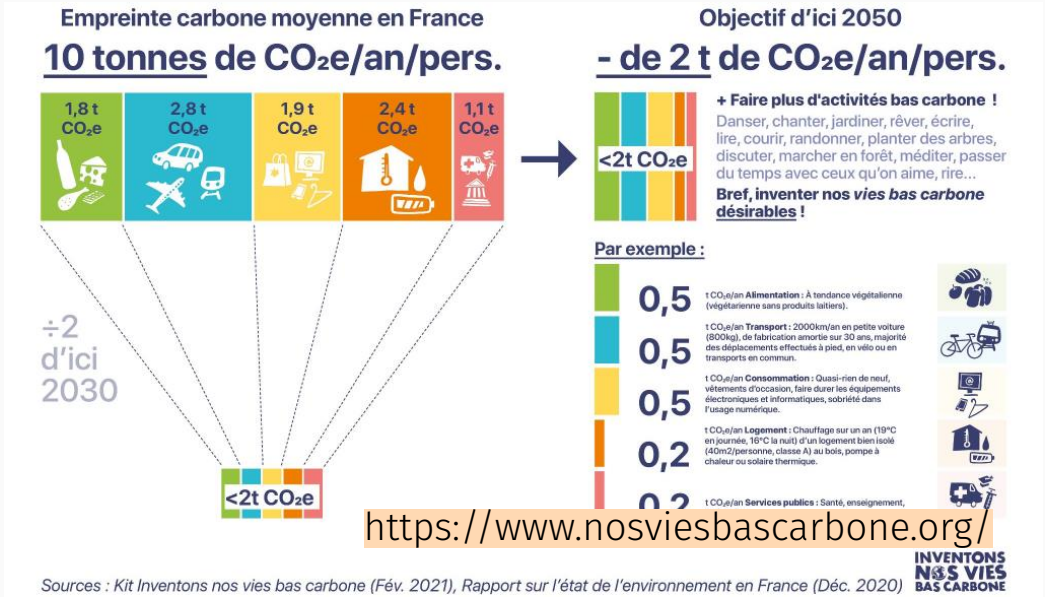


https://www.nosviesbascarbone.org/

Sources : Kit Inventons nos vies bas carbone (Fév. 2021), Rapport sur l'état de l'environnement en France (Déc. 2020)

Put aside biodiversity loss, pollution, freshwater, land system change...

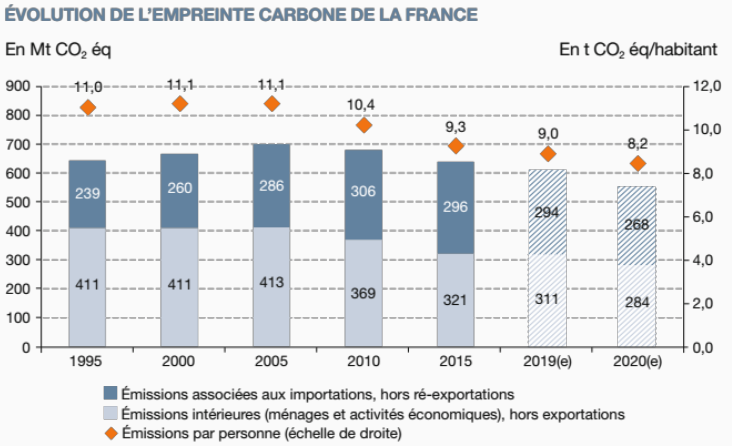


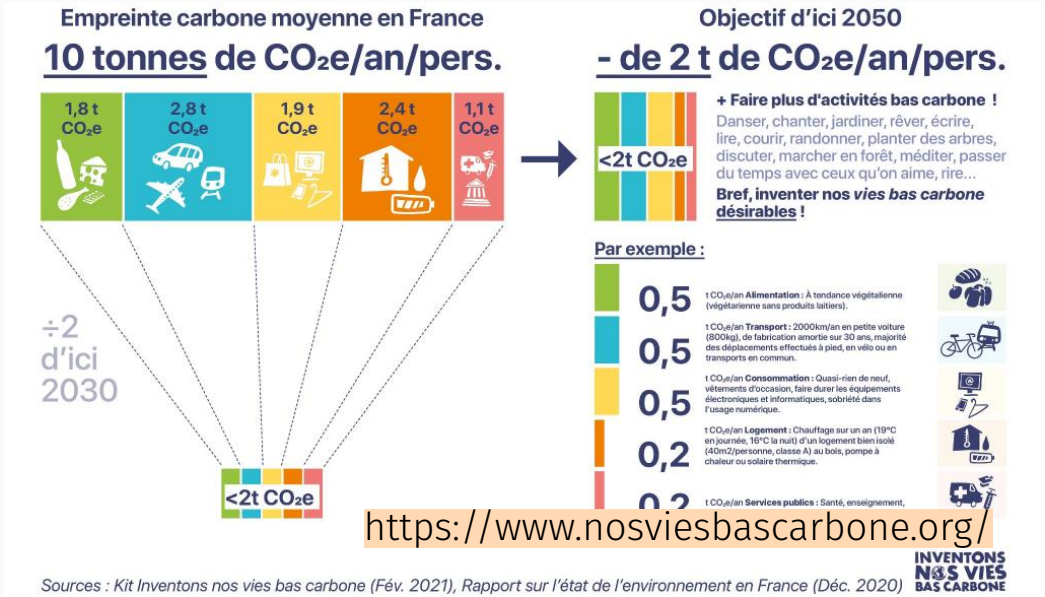


French government response

- *Verdissement de l'industrie: « pause » sur les normes environnementales*
- *Loi de programmation militaire (+41%)*
- *Nous devons préparer la France à une élévation de la température de 4 °C*
- Academia ? PEPR 5G, Cloud, NUMPEX, Quantique, IA, Agroécologie et numérique

Put aside biodiversity loss, pollution, freshwater, land system change...





French government response

- *Verdissement de l'industrie: « pause » sur les normes environnementales*
- *Loi de programmation militaire (+41%)*
- *Nous devons préparer la France à une élévation de la température de 4 °C*
- Academia ? PEPR 5G, Cloud, NUMPEX, Quantique, IA, Agroécologie et numérique

~~Several~~ scenarios on the table

- What will research/CS look like/be used for in such a world?
- Energy optimization/saving ≠ sobriety and frugality