Toward Reproducible Research in Computer Science

Arnaud Legrand
Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP

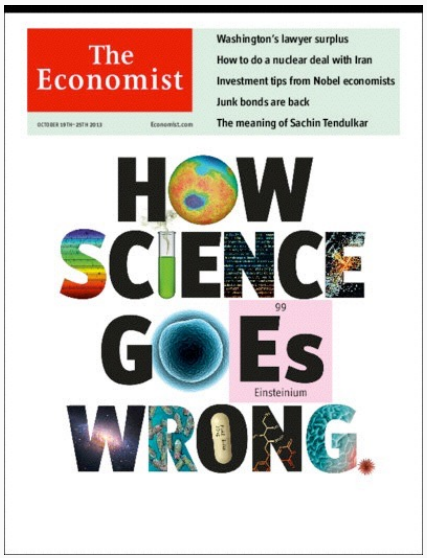
University of Neuchâtel, November 2019

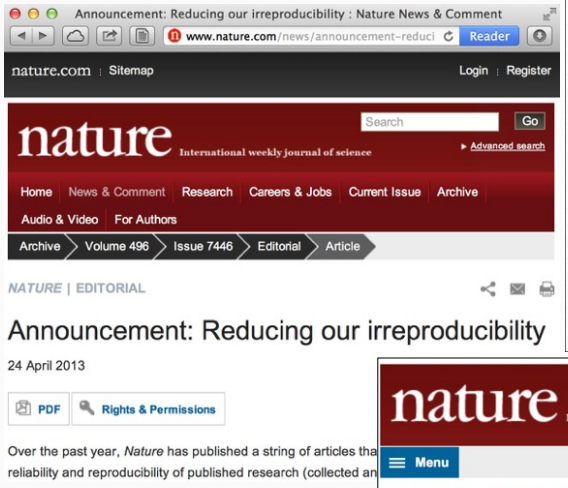
Experimental reproducibility for computer scientists

Reproducibility crisis ?

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

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.

- Ten papers coauthored in prestigious journals were retracted in Jan 2011

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.

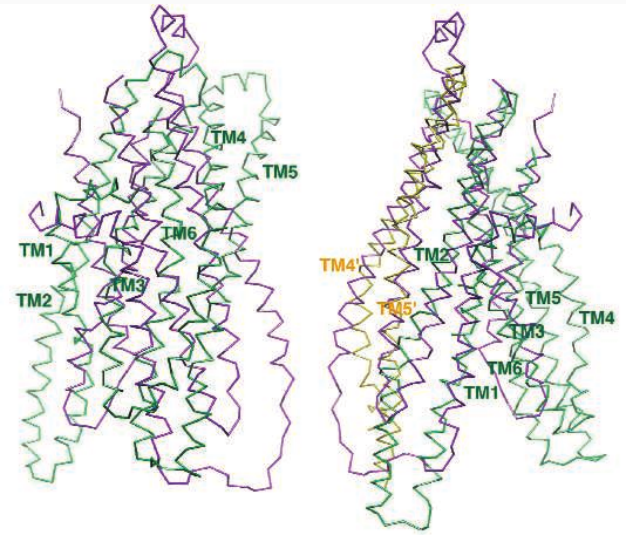
- Ten papers coauthored in prestigious journals were retracted in Jan 2011

More ? In 2015, Dong-Pyou Han was sentenced to 57 months imprisonment and fined US $7.2 million for fabricating data in HIV vaccine trials!

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



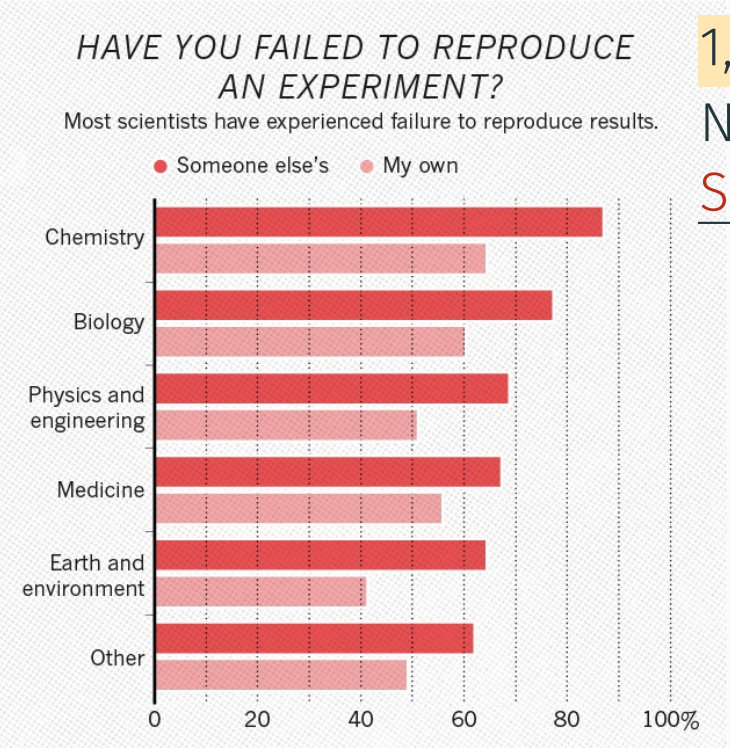
Geoffrey Chang (Scripps, UCSD) works on crystalography and studies the structure of cell membrane proteins.
He specialized in structures of multidrug resistant transporter proteins in bacteria:
MsbA de Escheria Choli (Science, 2001),
Vibrio cholera (Mol. Biology, 2003),
Salmonella typhimurium (Science, 2005)

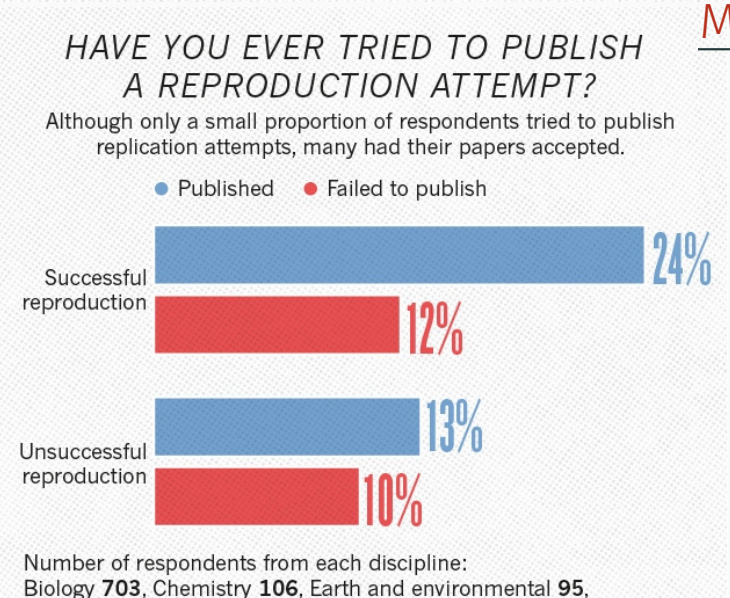
2006: Inconsistencies reveal a programming mistake

> *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 computational biology

Why are scientific studies so difficult to reproduce?





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

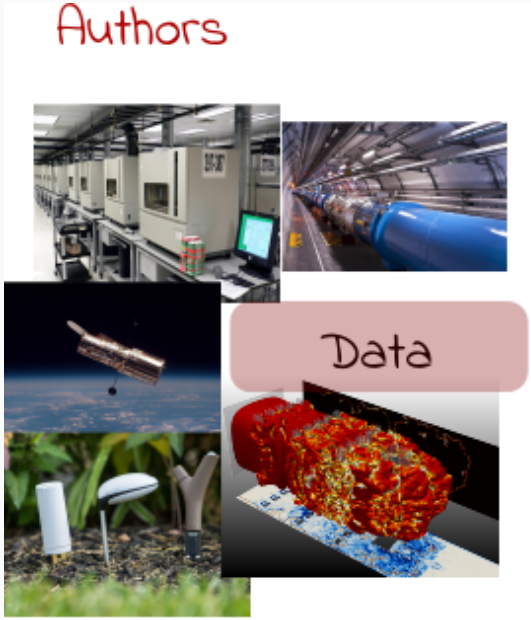
- The many biases, (apophenia, confirmation, ...), HARKing: bad designs
- Selective reporting, weak analysis (statistics, data processing, computational errors)
- Lack of information, code/raw data unavailable

Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues

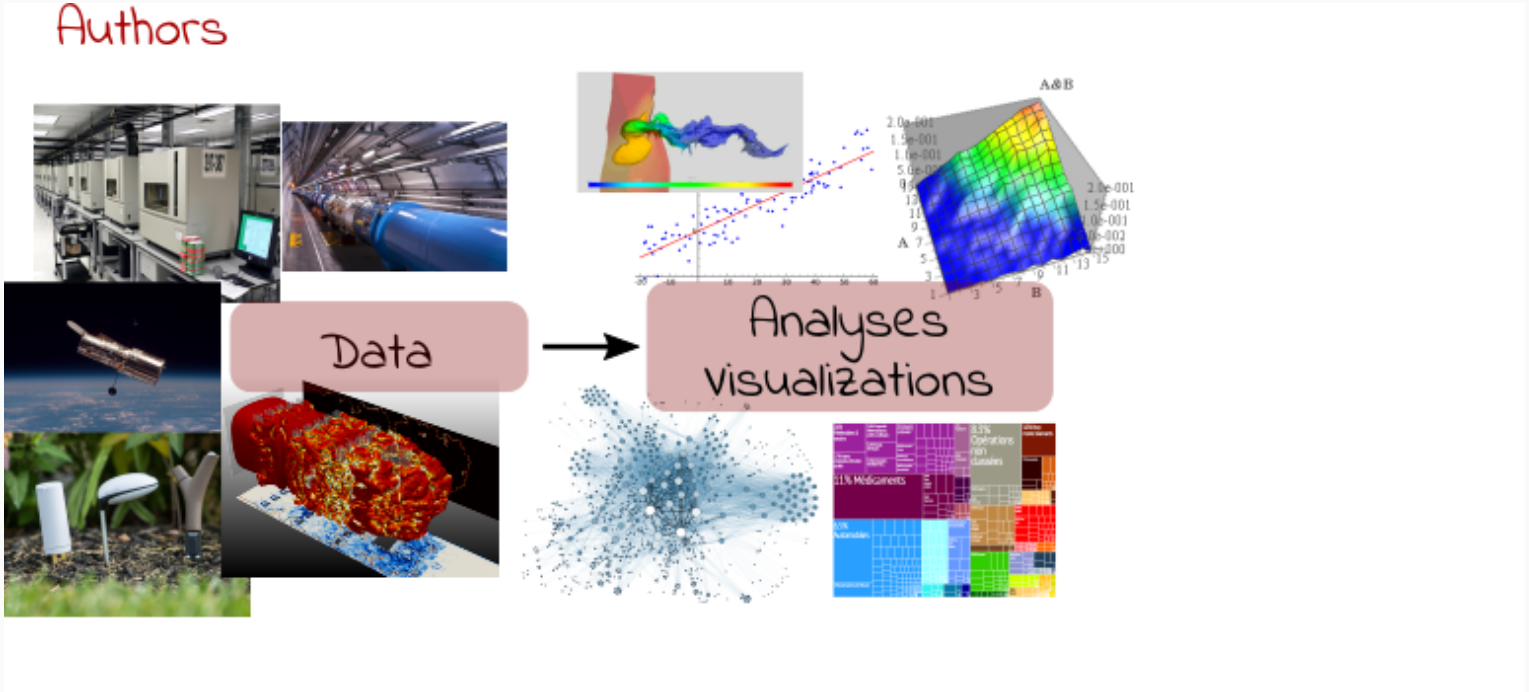


Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues

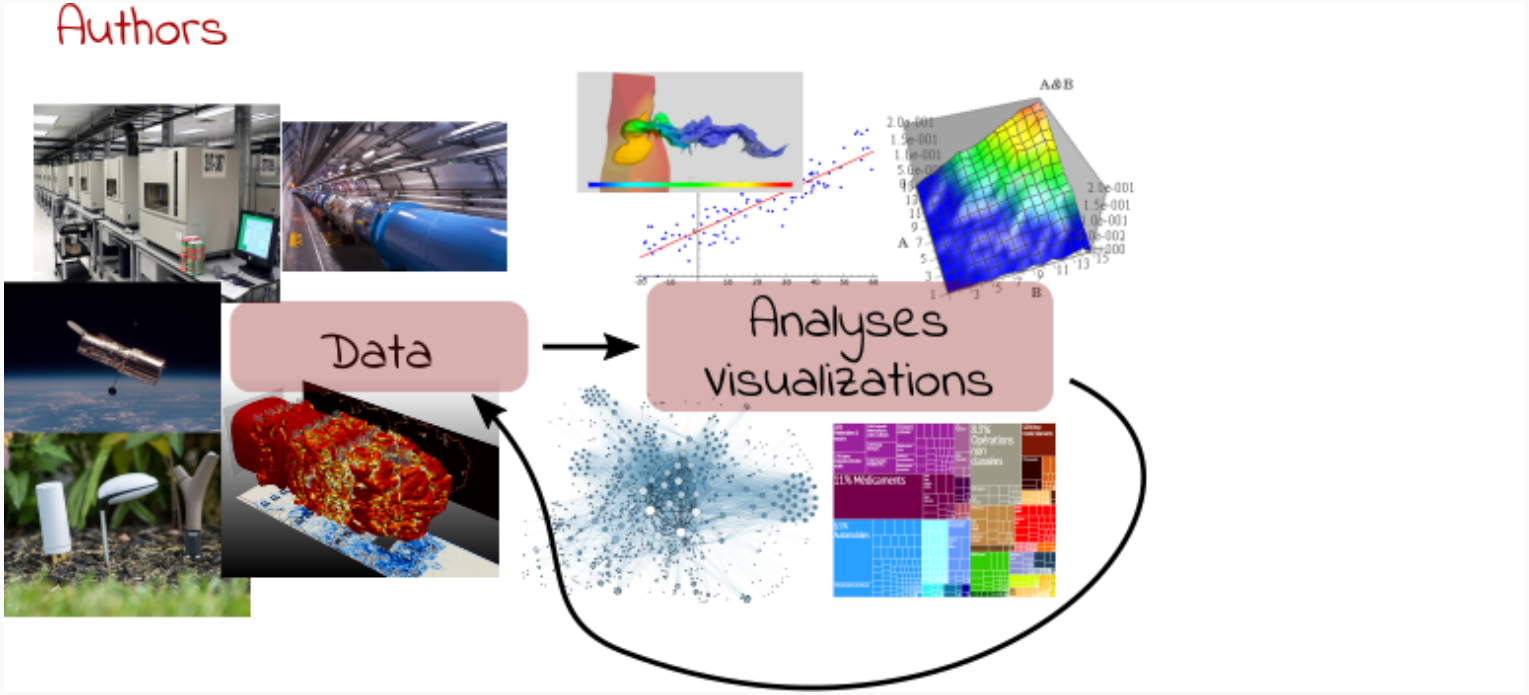


Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues

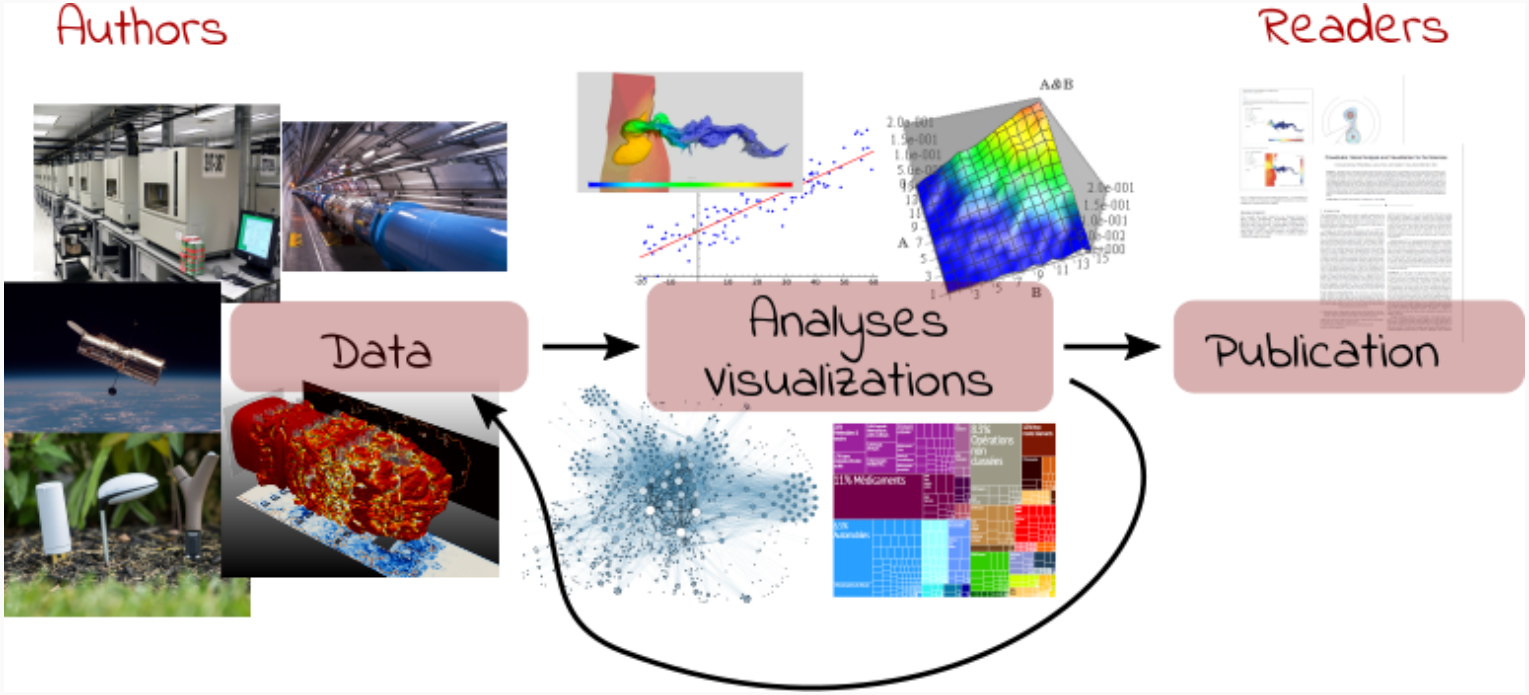


Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues

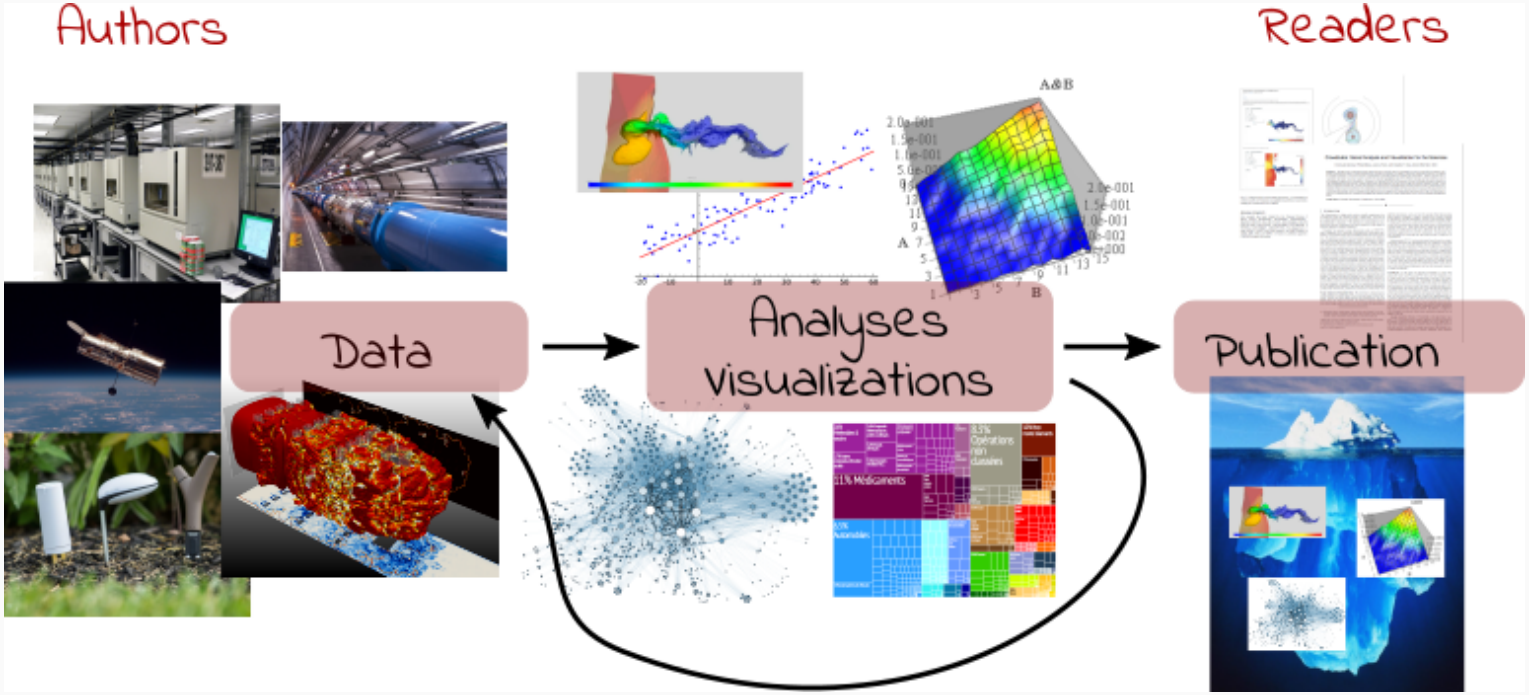


Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues

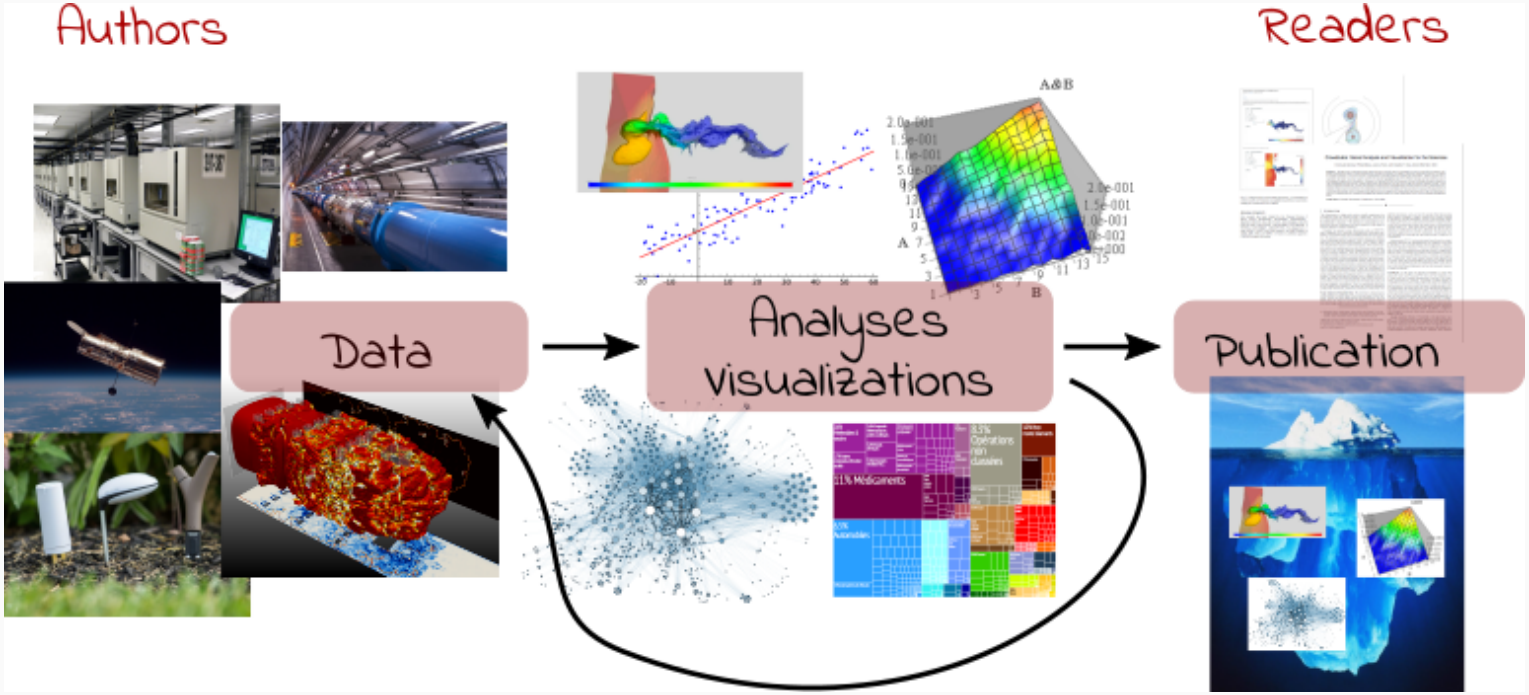


Different Reproducibility Concerns

Social Sciences, Oncology, ... methodology, statistics

Genomics software engineering, computational reproducibility, provenance, ...

Computational fluid dynamics numerical issues



Reproducible Research = Bridging the Gap by working Transparently

Reproducible Research Practices

"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:

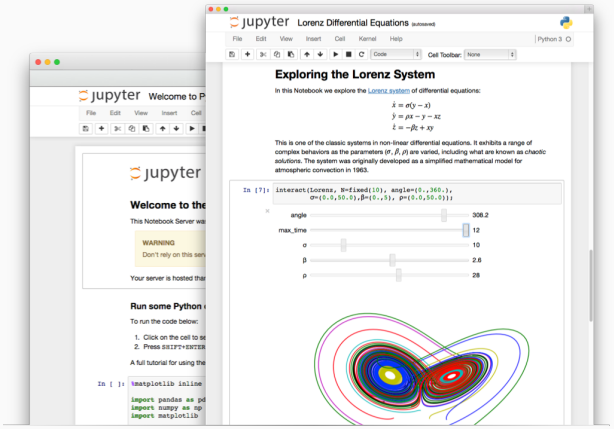
- Learn how to merge a publication with its underlying computational analysis.
- Teach researchers how to prepare a document in a form where they themselves can reproduce their own research results a year or more later by "pressing a single button".
- Learn how to leave finished work in a condition where 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.

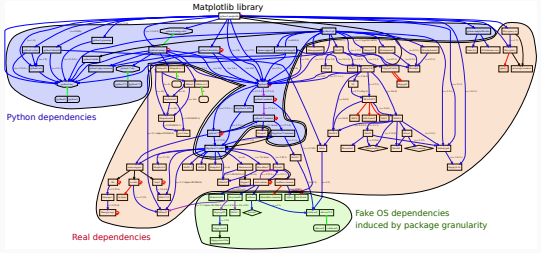
Now that we have begun using CD-ROM publication, we can go much further. Every figure caption contains a pushbutton that jumps to the appropriate science directory (folder) and initiates a figure rebuild command and then displays the figure, possibly as a movie or interactive program. We normally display 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

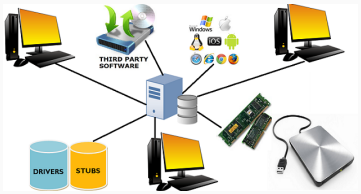
Existing Tools, Emerging Standards

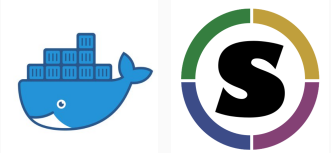
Notebooks and workflows



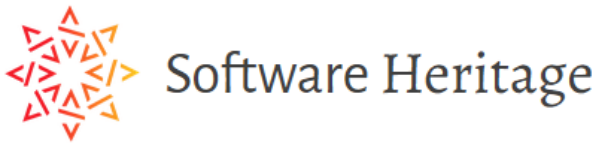
Software environments







Sharing platforms



Tool 1: Computational Notebooks/Litterate Programming

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

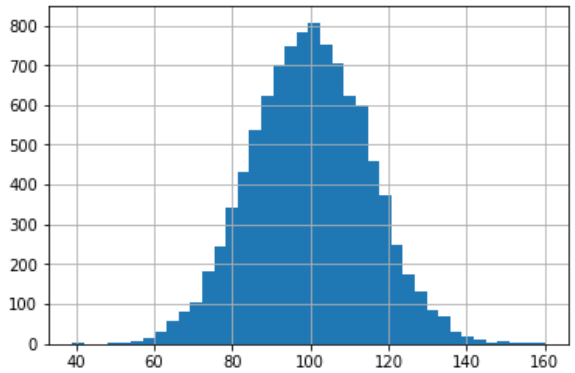
```
3.141592653589793
```

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... ☺).
```

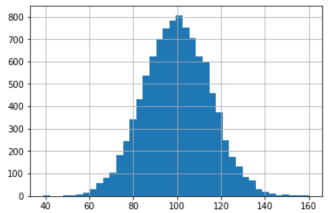
In [3]:

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*

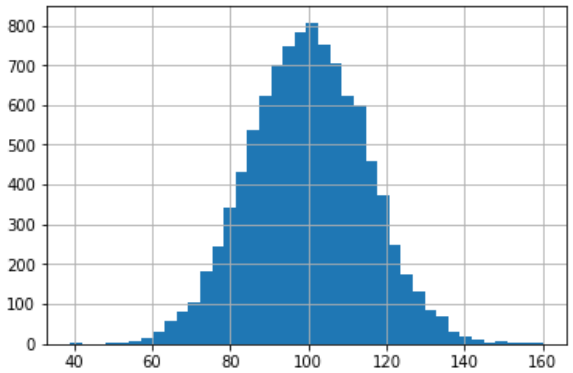
```
3.141592653589793
```

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... ☺).
```

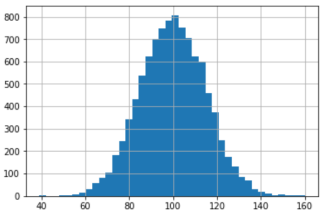
In [3]:

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()
```



MarkDown

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

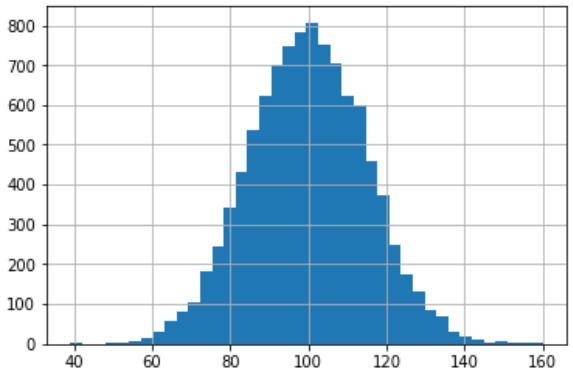
```
3.141592653589793
```

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)
```

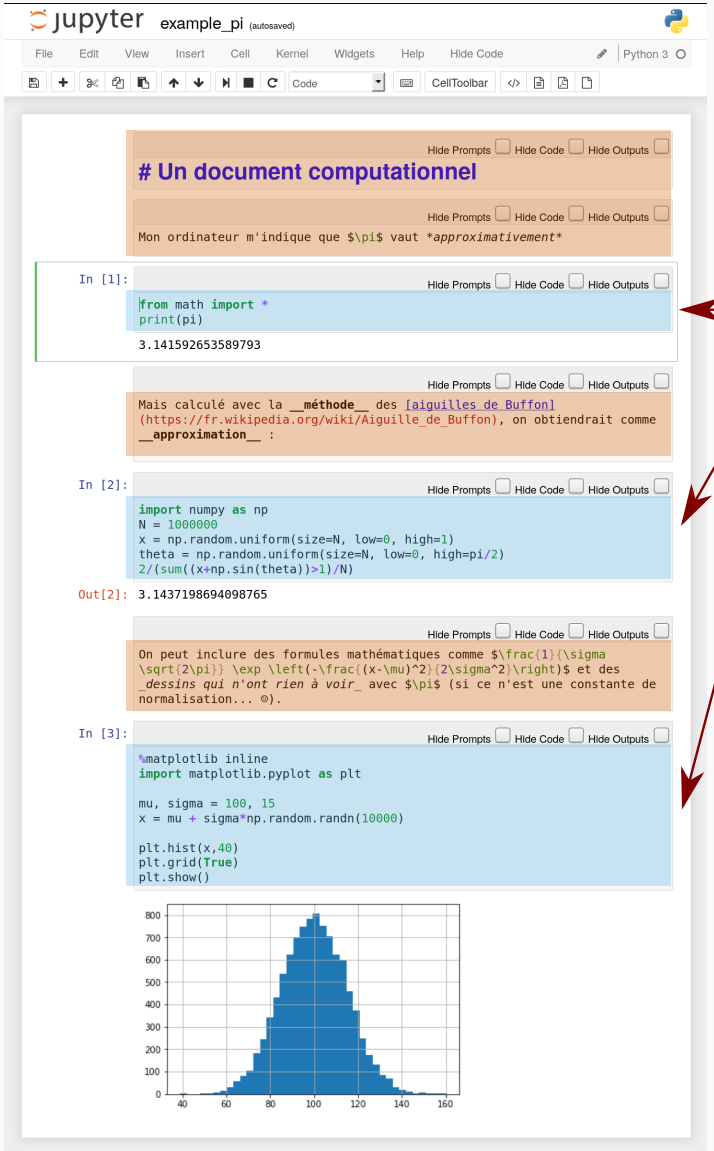
```
3.1437198694098765
```

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*

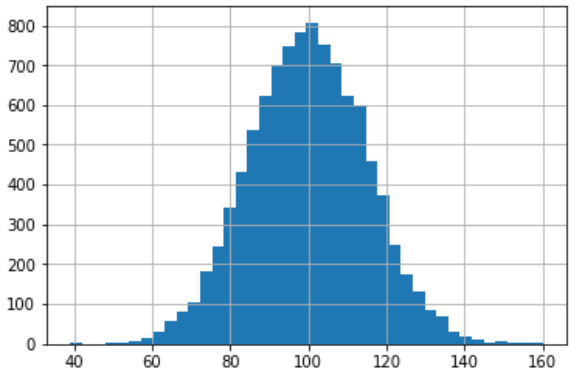
```
3.141592653589793
```

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)
```

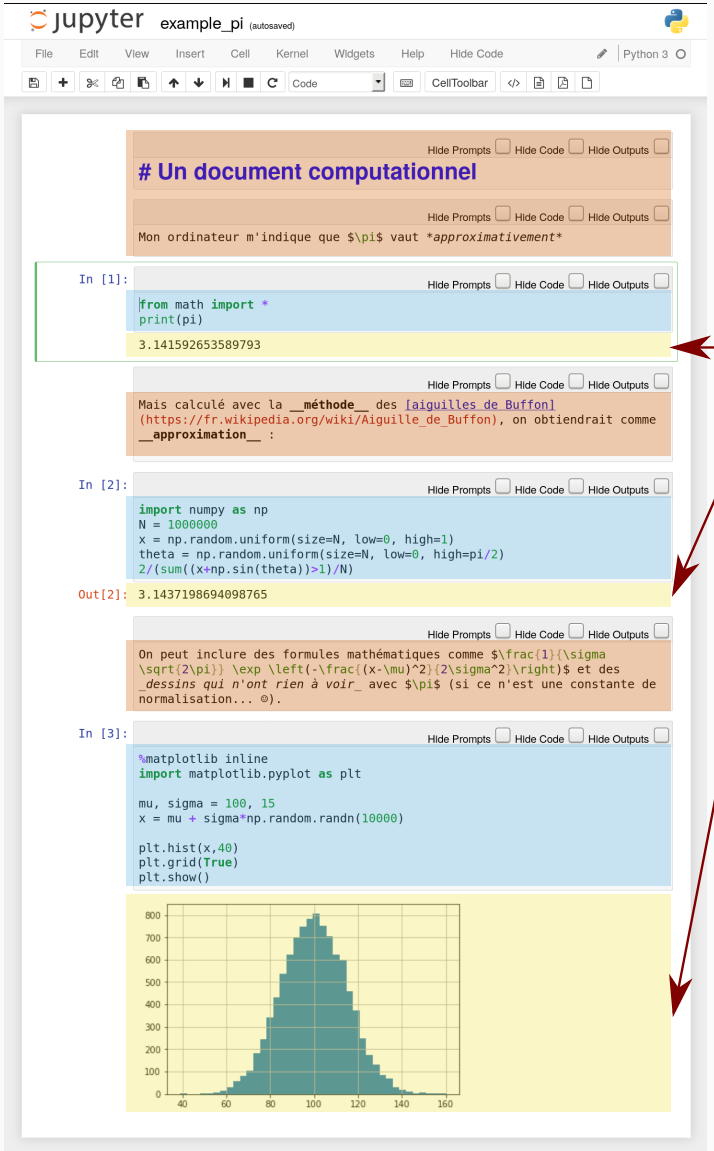
```
3.1437198694098765
```

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*

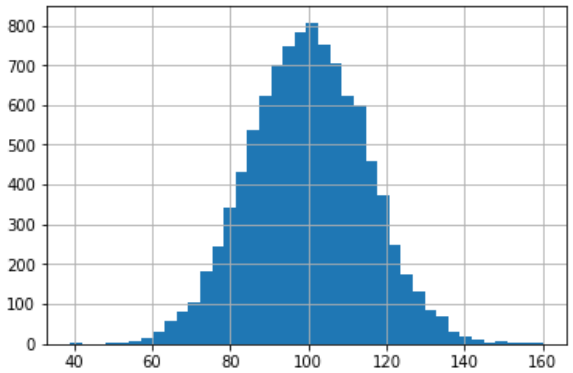
```
3.141592653589793
```

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)
```

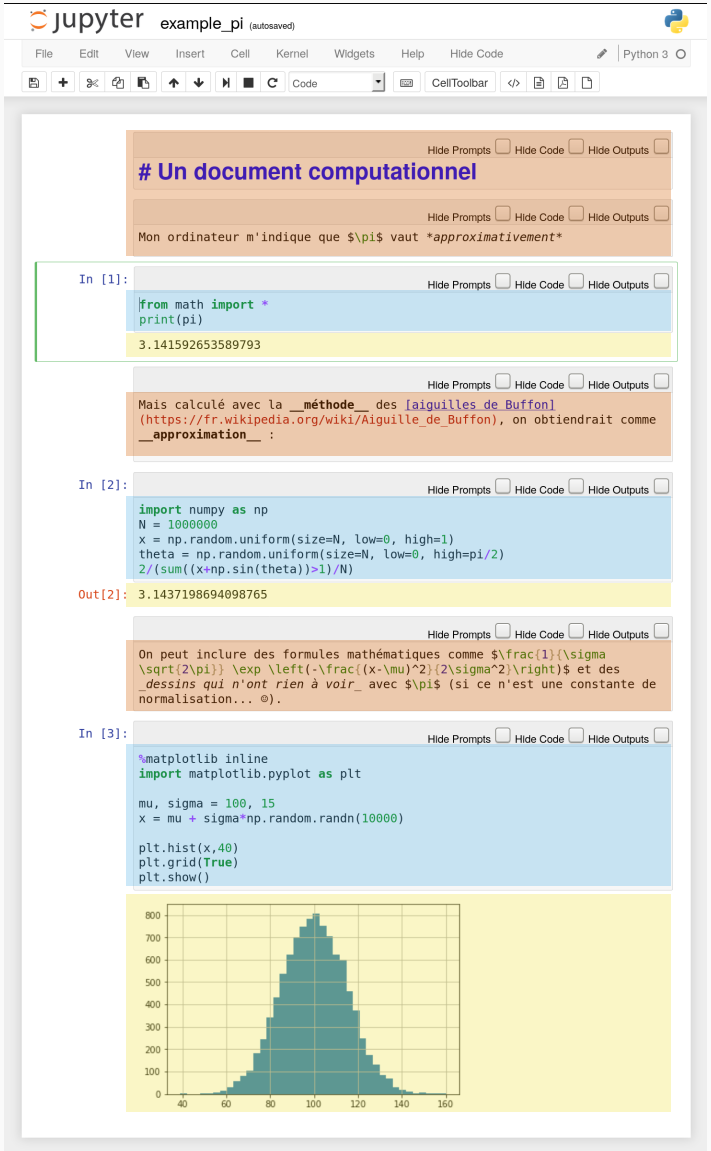
```
3.1437198694098765
```

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*

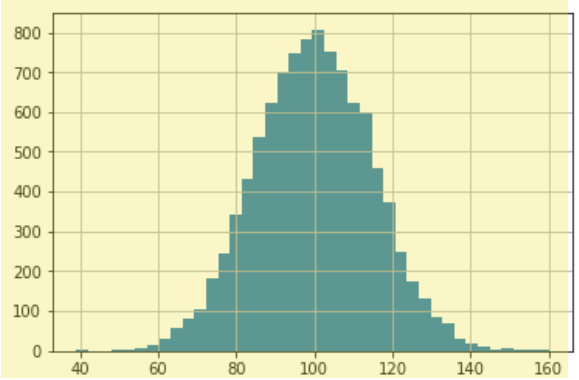
```
3.141592653589793
```

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)
```

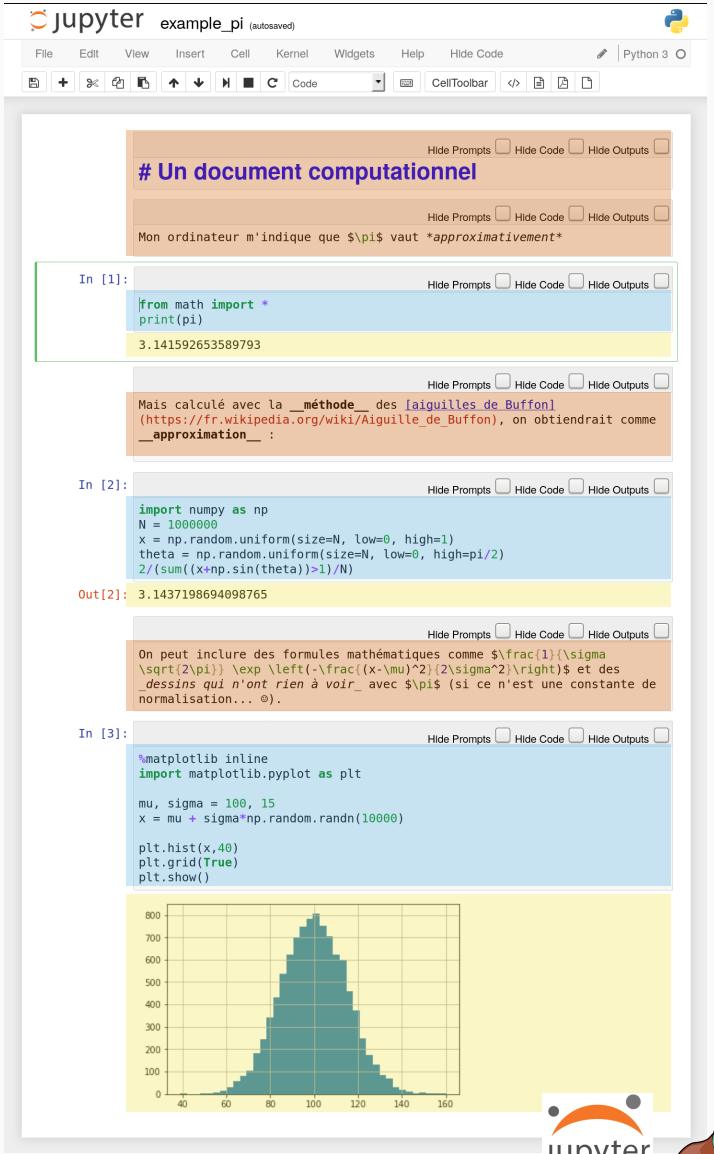
```
3.1437198694098765
```

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



Export

Document final

Un document computationnel

Mon ordinateur m'indique que π vaut *approximativement*

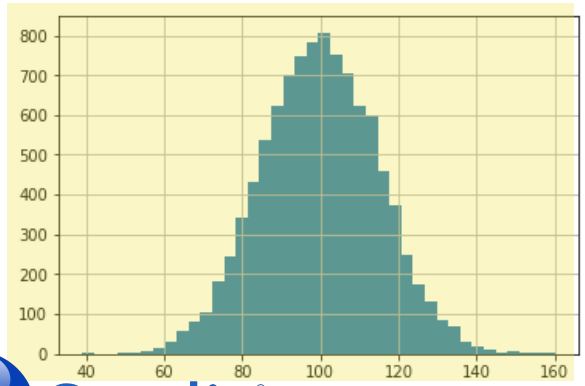
```
3.141592653589793
```

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 2: Fighting Software Environments Nightmare

What is hiding behind a simple

```
1 import matplotlib
```

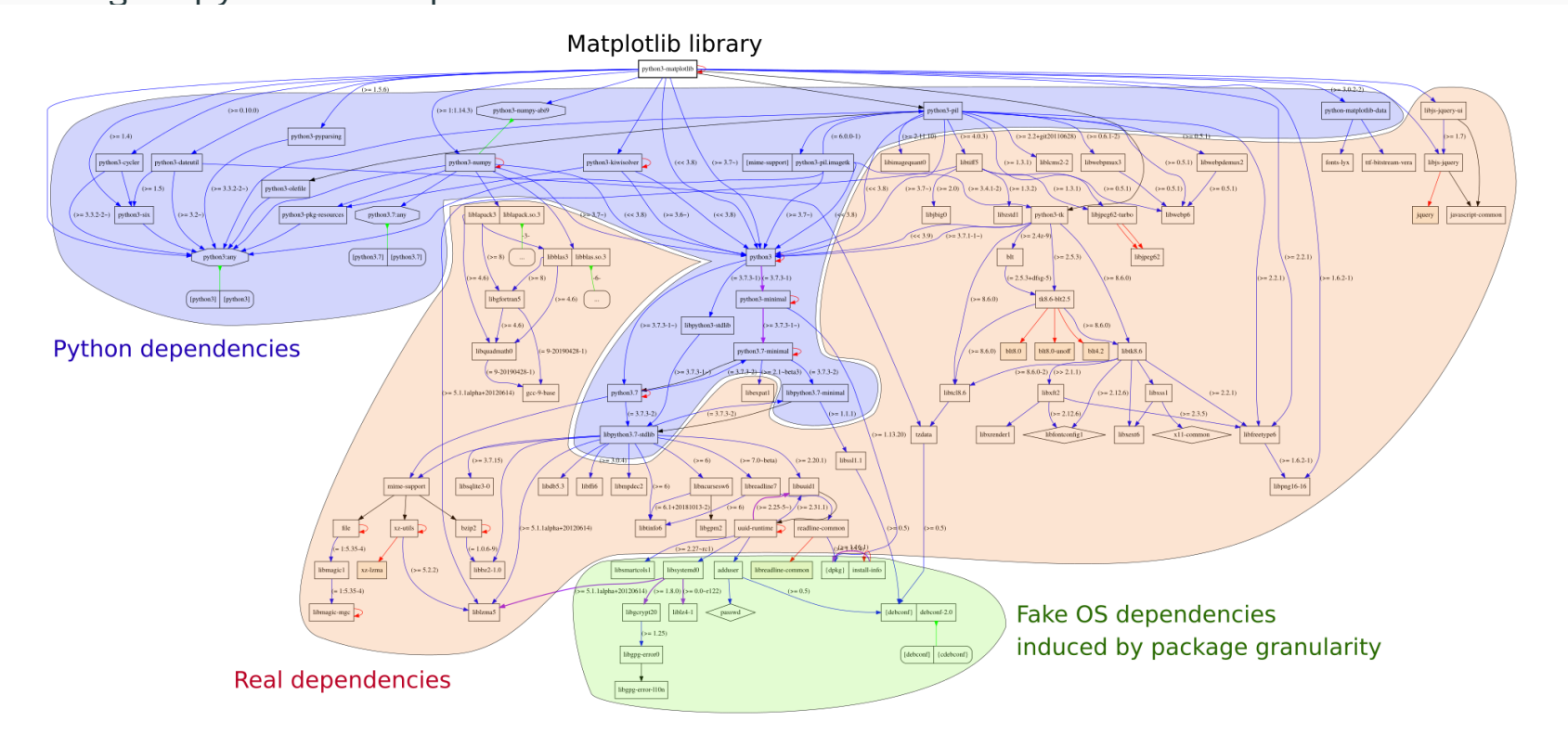
```
Package: python3-matplotlib
Version: 2.1.1-2
Depends: python3-dateutil, python-matplotlib-data (>= 2.1.1-2),
python3-pyparsing (>= 1.5.6), python3-six (>= 1.10), python3-tz,
libjs-jquery, libjs-jquery-ui, python3-numpy (>= 1:1.13.1),
python3-numpy-abi9, python3 (<< 3.7), python3 (>= 3.6~),
python3-cycler (>= 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

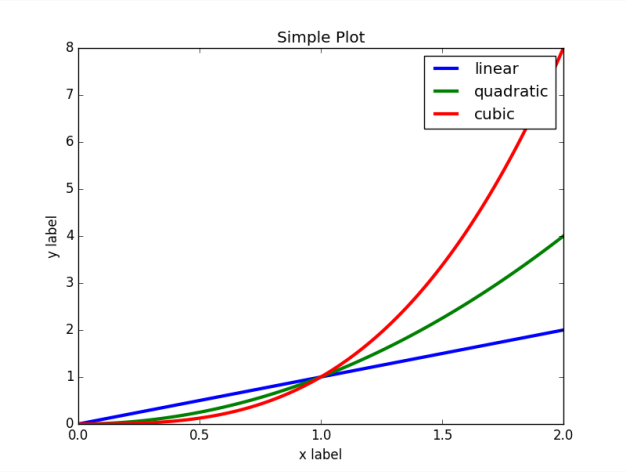
```
1 python2 -c "print(10/3)"
2 python3 -c "print(10/3)"
```

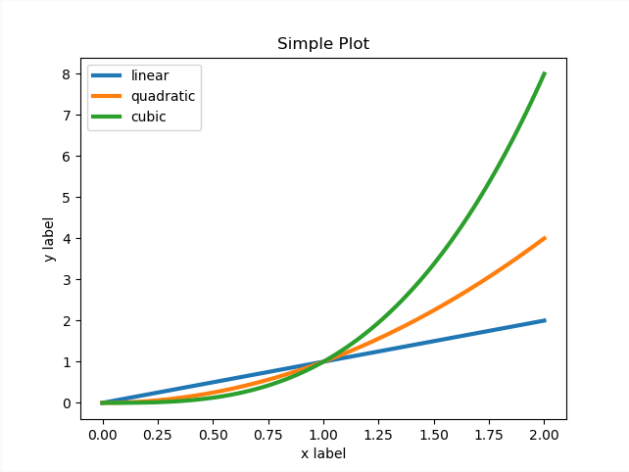
```
3
3.3333333333333335
```

Tool 2: Fighting Software Environments Nightmare

Python and its rapidly evolving environment

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

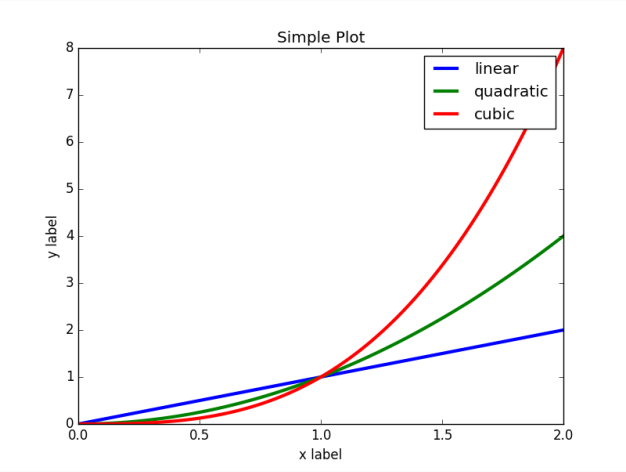


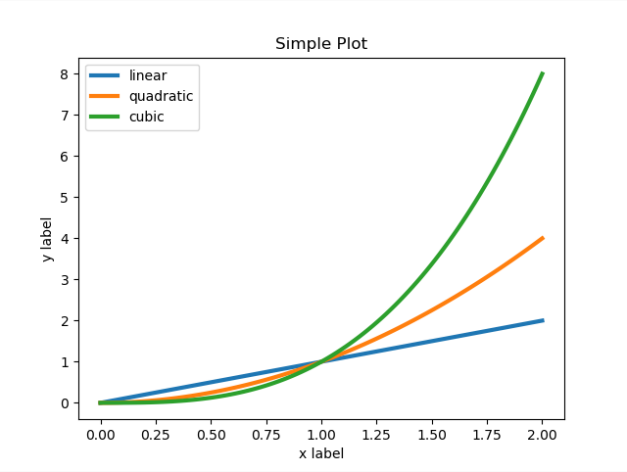


Tool 2: Fighting Software Environments Nightmare

Python and its rapidly evolving environment

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



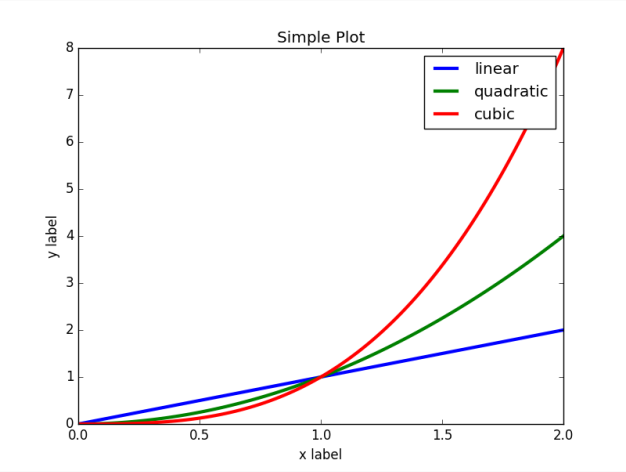


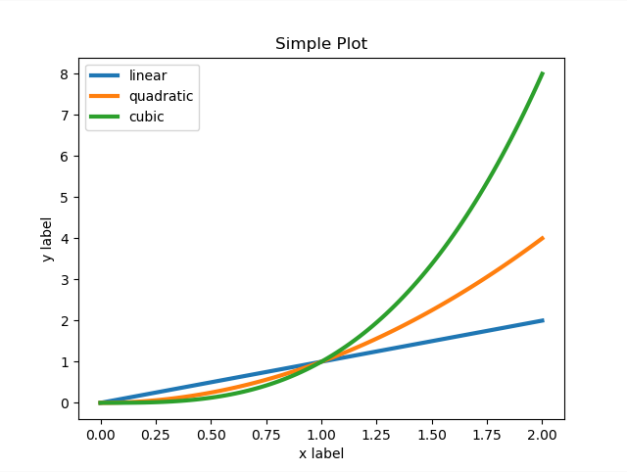
Cortical Thickness Measurements (PLOS ONE, June 2012): *FreeSurfer*: *differences were found between the Mac and HP workstations and between Mac OSX 10.5 and OSX 10.6.*

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.*

ReproZip

Guix

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*

Article archives arXiv.org HAL archives-ouvertes.fr

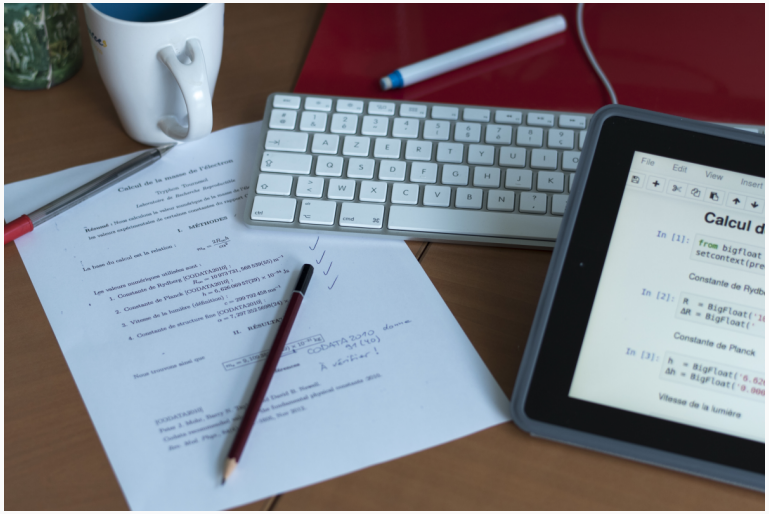
Data archives figshare zenodo

Software Archive Software Heritage

or = awesome collaborations ≠ archive

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...

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

What about Computer Science ?

All this is about Computational Sciences. Should we care ?

Computer Science is young and inherits from Mathematics, Engineering, Nat. Sciences, Linguistic, ...

Purely theoretical scientists whose practice is close to mathematics *may* not be concerned (can't publish a math article without releasing the proofs).

- Have a look at talk by Vladimir Voevodsky in 2014 at Princeton 😉

All this is about Computational Sciences. Should we care ?

Computer Science is young and inherits from Mathematics, Engineering, Nat. Sciences, Linguistic, ...

Purely theoretical scientists whose practice is close to mathematics *may* not be concerned (can't publish a math article without releasing the proofs).

- Have a look at talk by Vladimir Voevodsky in 2014 at Princeton 😉

  Computer science is not more related to computers than Astronomy to telescopes

  – *Dijkstra (mis-attributed)*

Right, why should we care about computers? They are deterministic machines after all, right? 😉

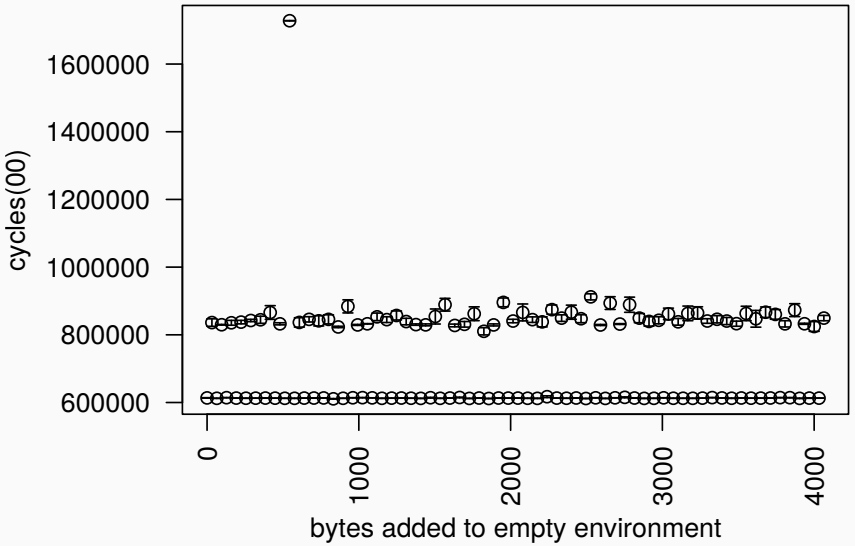
Model $\neq$ **Reality**. Although designed and built by human beings, computer systems are so complex that mistakes easily slip in...

Experimenting with computers

Machines are real!

Brendan Gregg: Shouting in the data center

Machines are complicated



Mytkowicz et al. *Producing wrong data without doing anything obviously wrong!* ACM SIGPLAN Not. 44(3), March 2009

Our reality evolves!!! The hardware keeps evolving so most results on old platforms quickly become obsolete (although, we keep building on such results 😉).

We need to regularly revisit and allow others to build on our work!

Computer performance ? Well, I design algorithms!

- ”Real” problems are all NP-hard, Log-APX, etc.
- Real workload = ~~NP-completeness proof widgets~~, regularities and properties (difficult to formally state but that should be exploited)

Algorithms are evaluated on particular workloads that impact both their running time and the quality of the solutions

Computer performance ? Well, I design algorithms!

- ”Real” problems are all NP-hard, Log-APX, etc.
- Real workload = ~~NP-completeness proof widgets~~, regularities and properties (difficult to formally state but that should be exploited)

Algorithms are evaluated on particular workloads that impact both their running time and the quality of the solutions

Image Processing: True horror stories, E. Meinhardt-Llopis, CANUM 2016

- *The proposed multigrid algorithm converges to the solution of the problem in O(N)* using biharmonic functions
- Surprisingly, our naive multi-scale Gauss-Seidel converges much faster

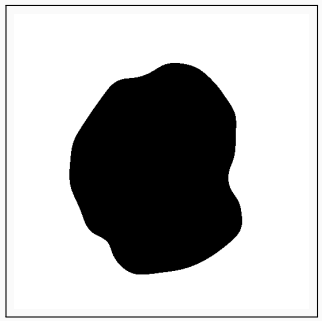
Computer performance ? Well, I design algorithms!

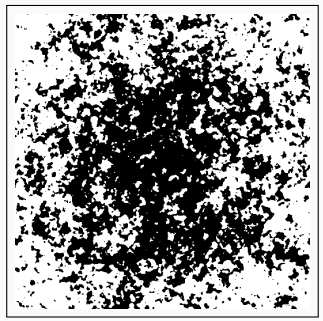
- ”Real” problems are all NP-hard, Log-APX, etc.
- Real workload = ~~NP-completeness proof widgets~~, regularities and properties (difficult to formally state but that should be exploited)

Algorithms are evaluated on particular workloads that impact both their running time and the quality of the solutions

Image Processing: True horror stories, E. Meinhardt-Llopis, CANUM 2016

- *The proposed multigrid algorithm converges to the solution of the problem in O(N)* using biharmonic functions
- Surprisingly, our naive multi-scale Gauss-Seidel converges much faster





I design Machine Learning Algorithms

Machine Learning: Trouble at the lab, The Economist 2013

> *According to some estimates, three-quarters of published scientific papers in the field of machine learning are bunk because of this "overfitting".* – *Sandy Pentland (MIT)*

Every month in CACM, there is an article about the ethical consequences of Machine Learning on:

- Car driving, Autonomous guns, Law enforcement (risk assessment, predictive policing), ...

  It's Not the Algorithm, It's the Data (CACM, Feb. 2017)
- Advertising, Loan attribution, Selection at University, Organ transplant

Increasing society concern about fairness and transparency

All I care about is the algorithm output

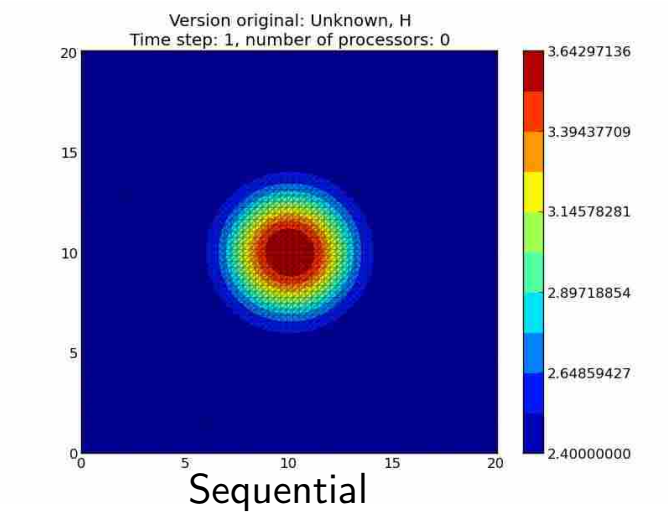
Did I mention we have parallel machines nowadays? 😉

Telemac2D: the simplest gouttedo simulation

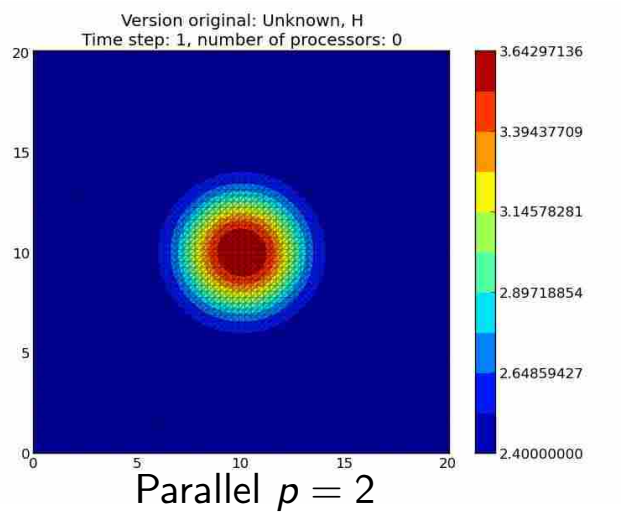
The gouttedo test case

- 2D-simulation of a water drop fall in a square bassin
- Unknown: water depth for a 0.2 sec time step
- Triangular mesh: 8978 elements and 4624 nodes

Expected numerical reproducibility (time step = 1, 2, ...)



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

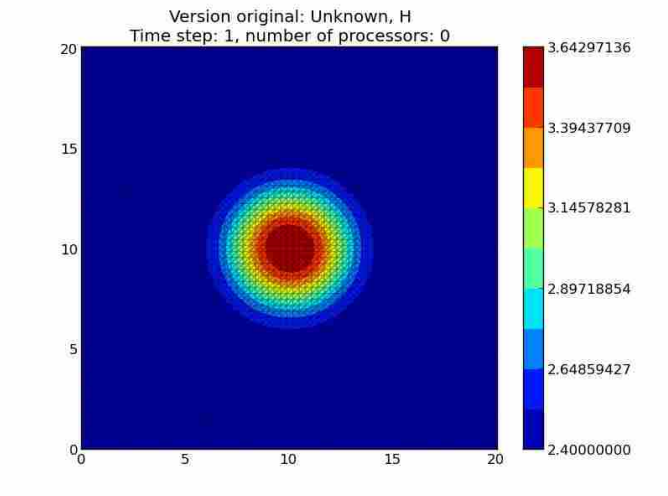
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

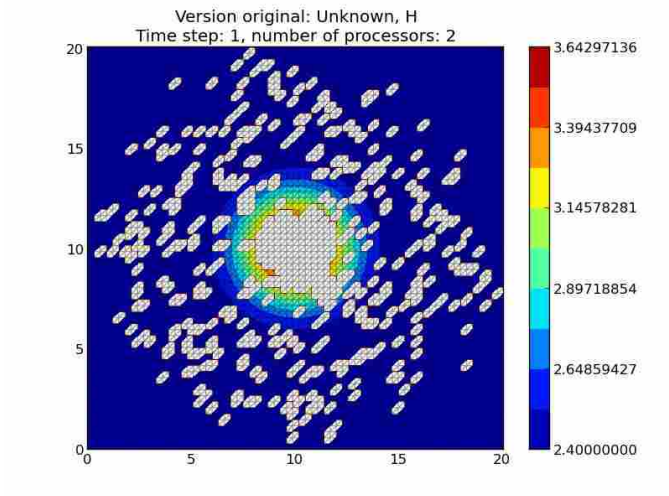
A white plot displays a non-reproducible value

Numerical reproducibility?

time step $= 1$



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

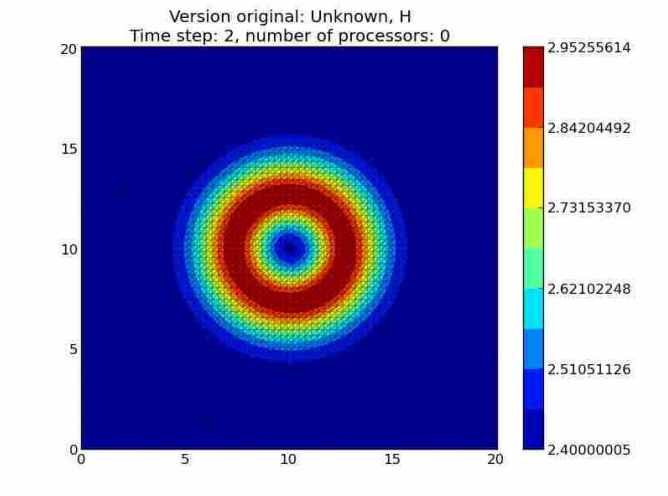
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

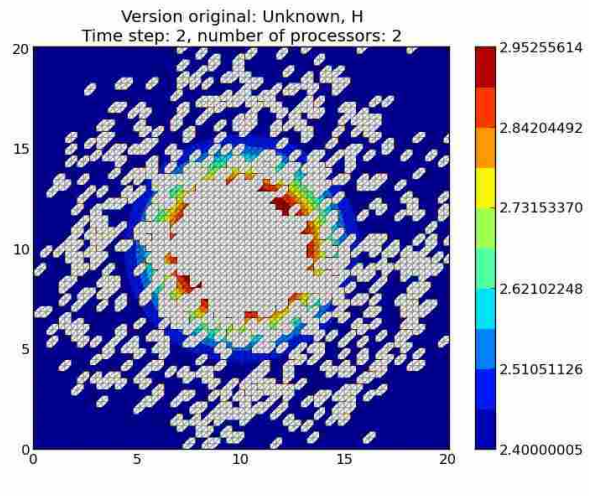
A white plot displays a non-reproducible value

Numerical reproducibility?

time step $= 2$



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

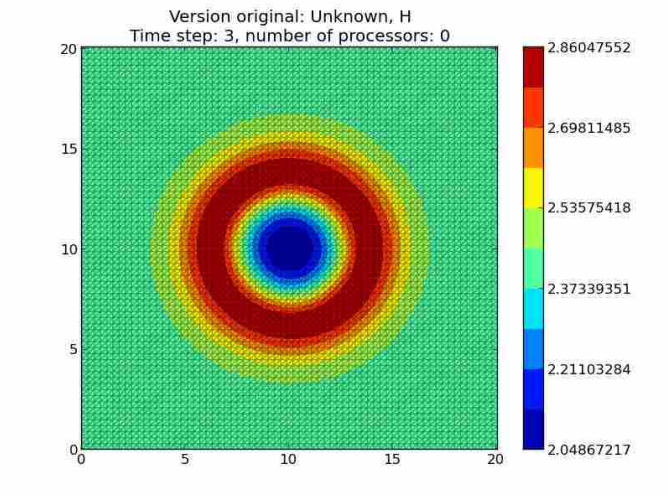
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

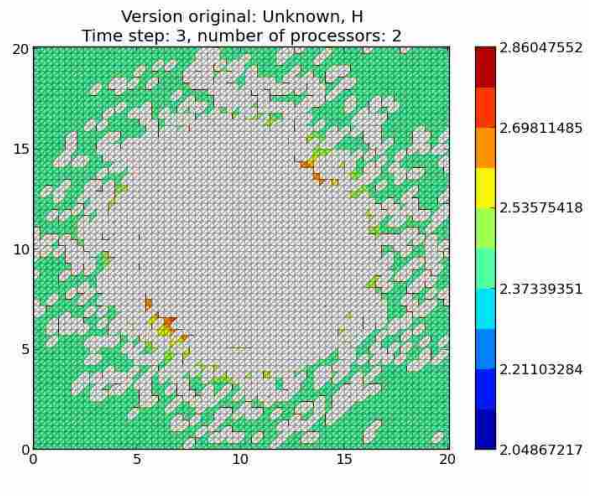
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 3



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

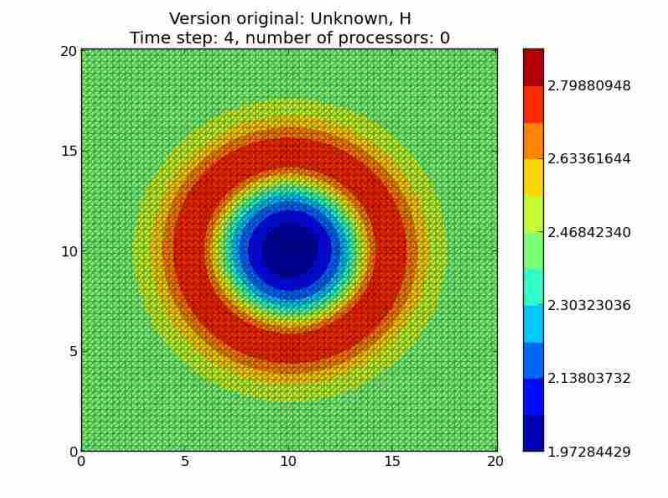
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

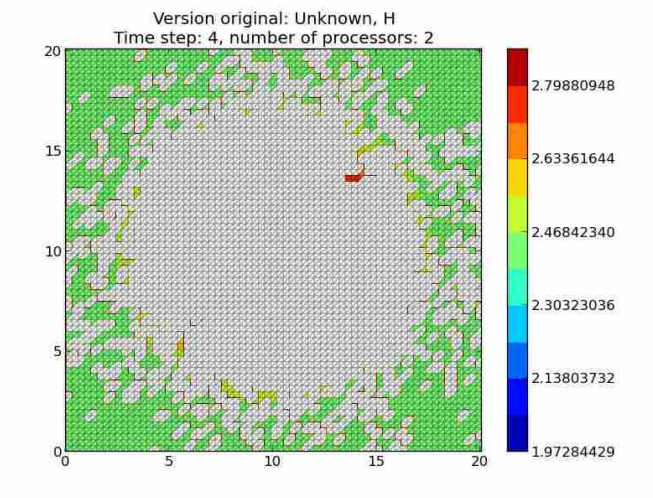
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 4



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

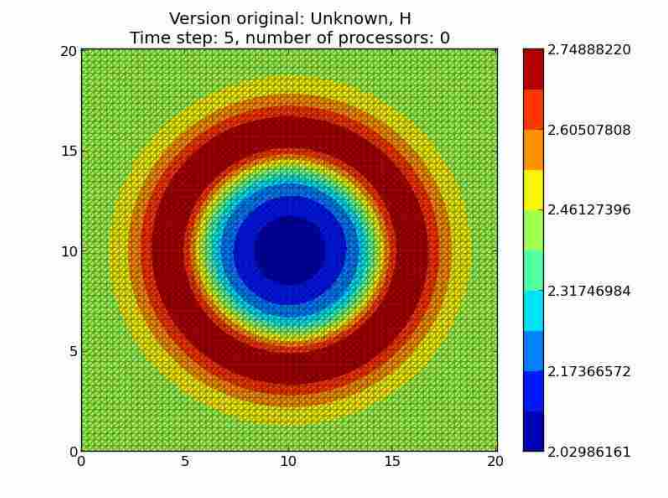
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

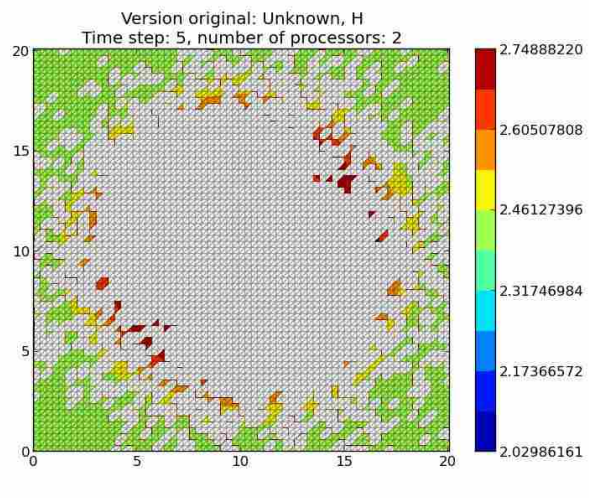
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 5



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

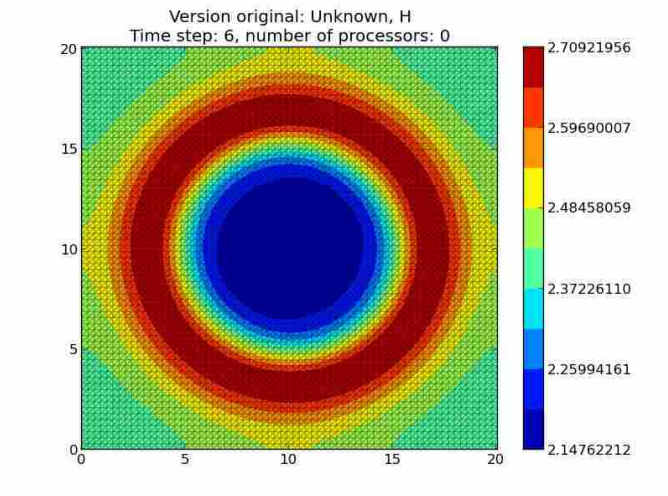
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

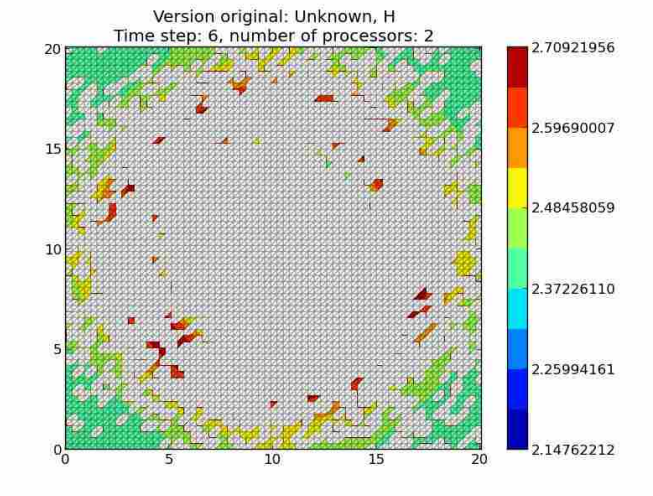
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 6



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

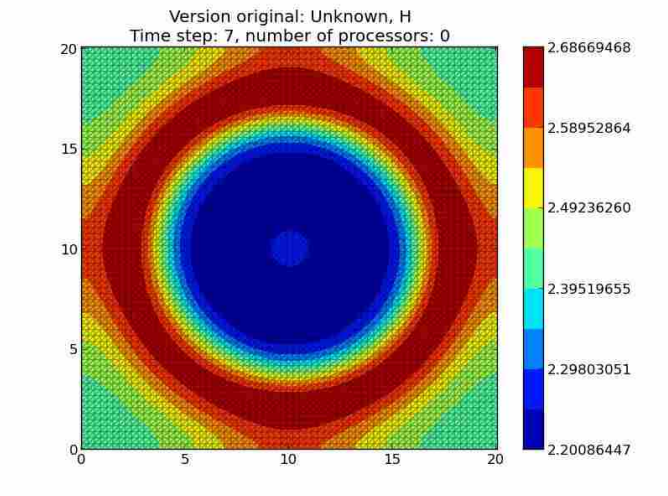
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

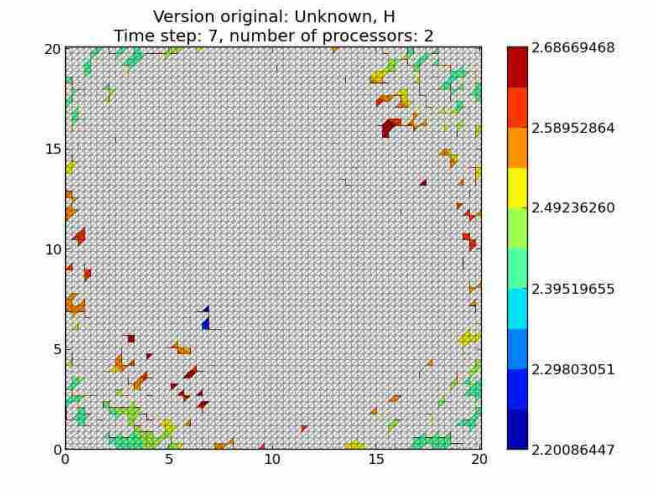
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 7



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

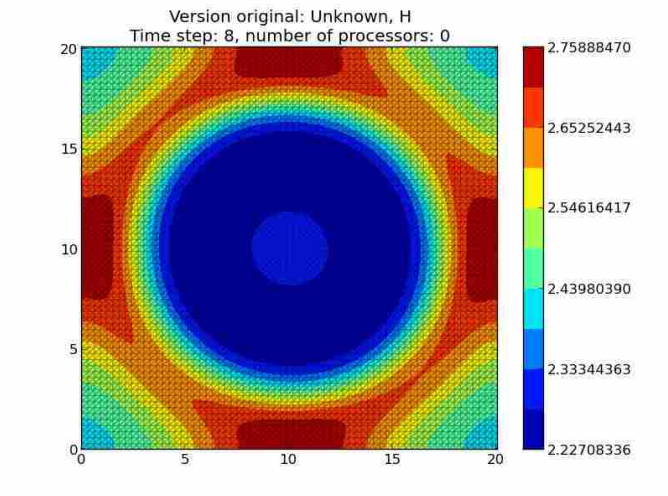
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

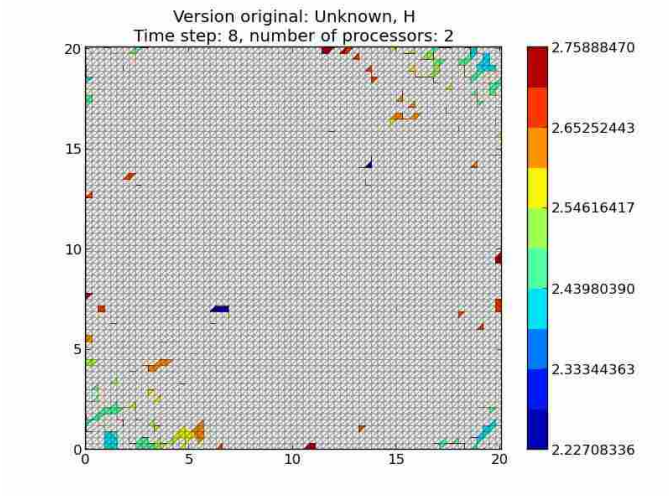
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 8



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

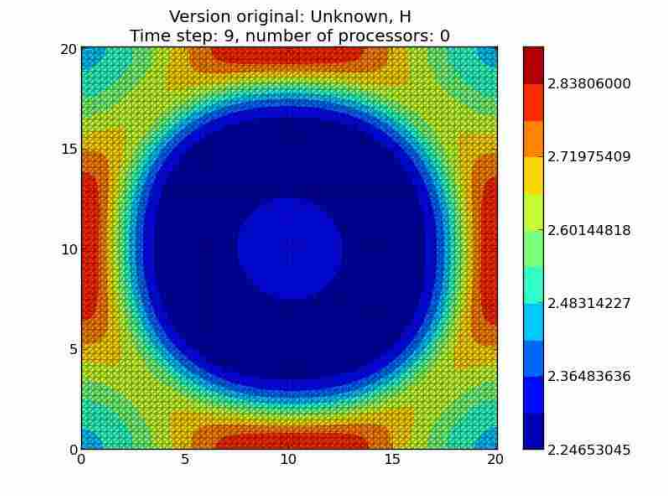
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

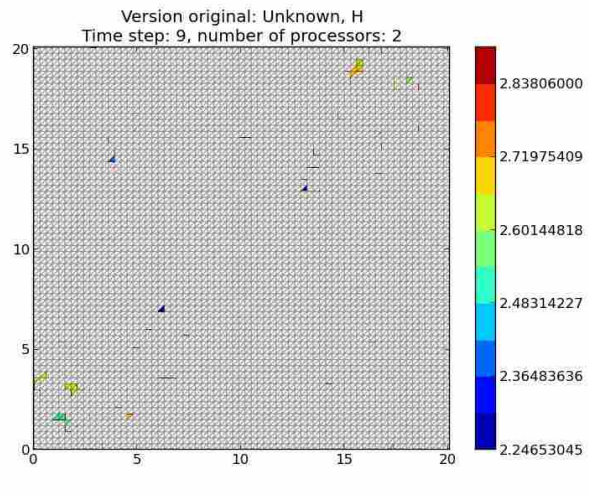
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 9



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

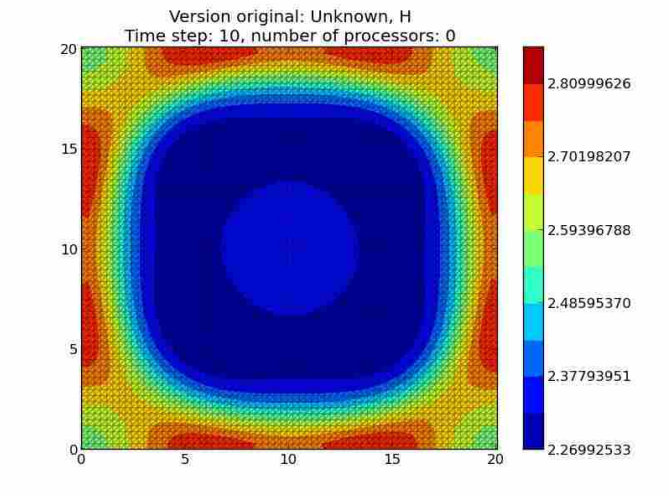
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

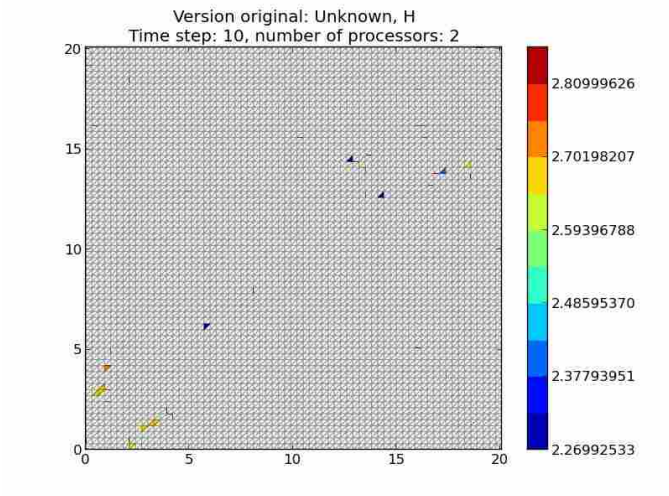
A white plot displays a non-reproducible value

Numerical reproducibility?

time step $= 10$



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

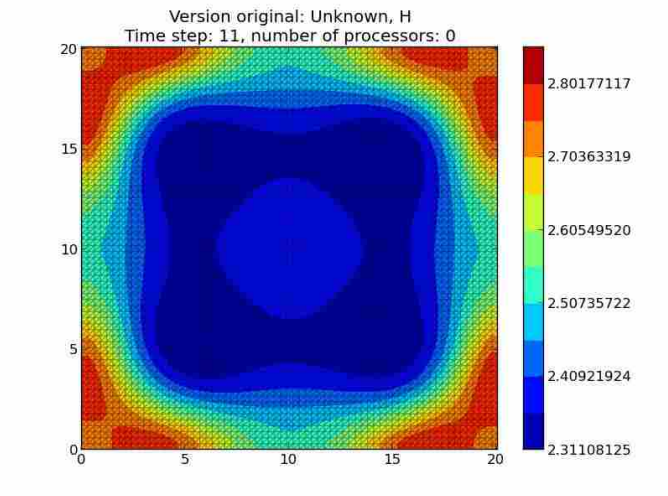
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

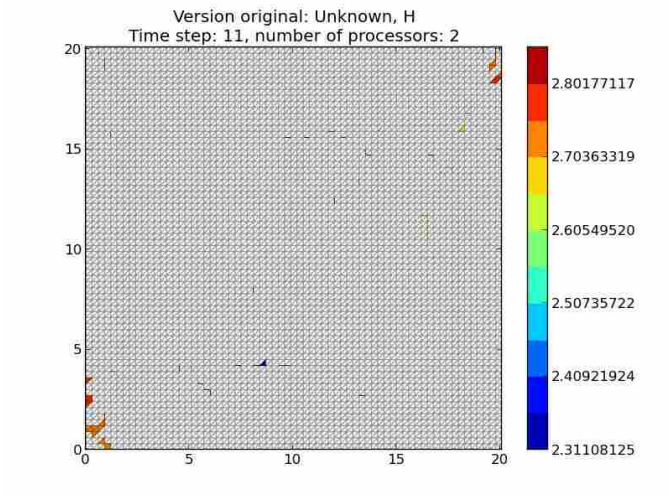
A white plot displays a non-reproducible value

Numerical reproducibility?

time step $= 11$



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

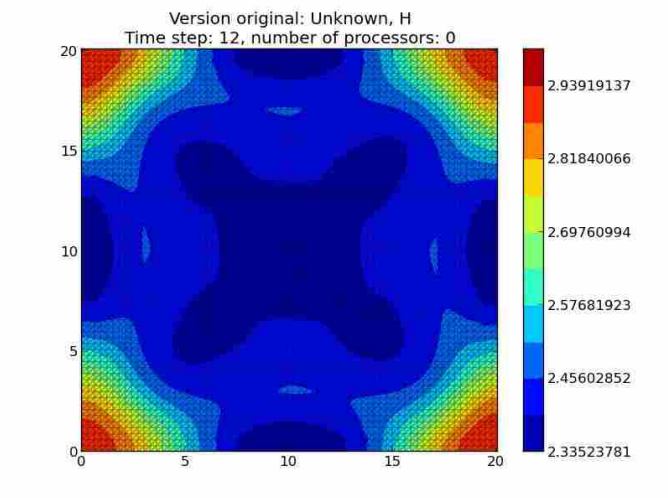
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

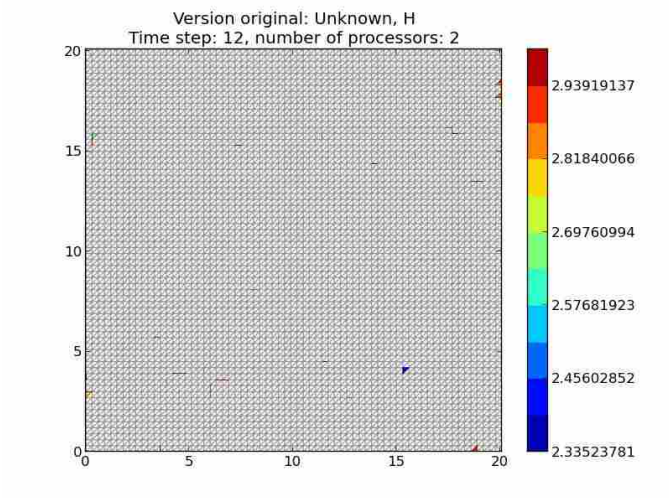
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 12



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

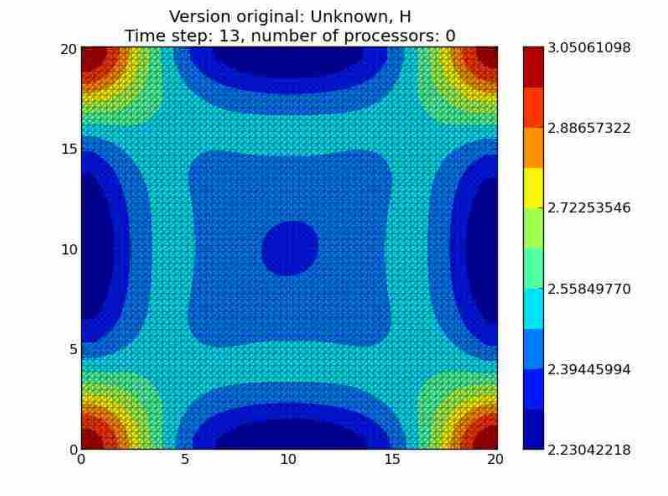
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

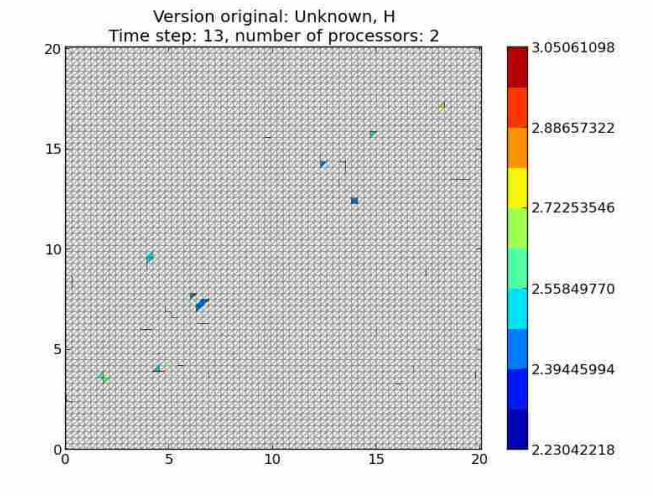
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 13



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

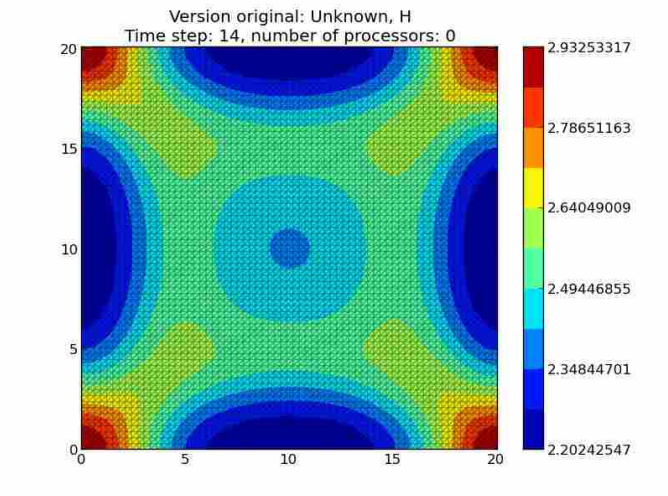
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

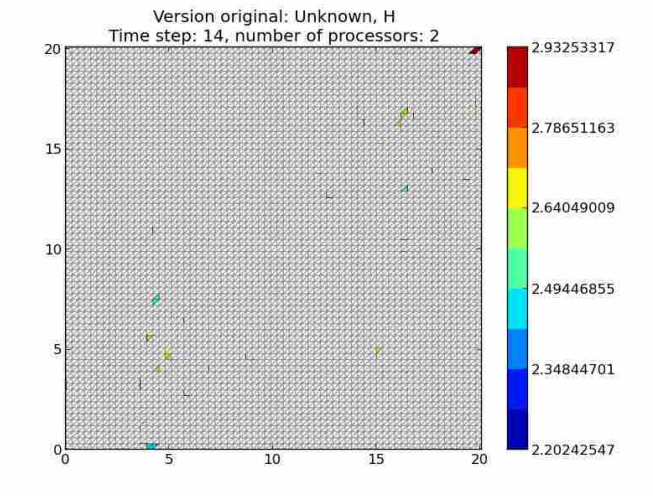
A white plot displays a non-reproducible value

Numerical reproducibility?

time step = 14



Sequential



Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

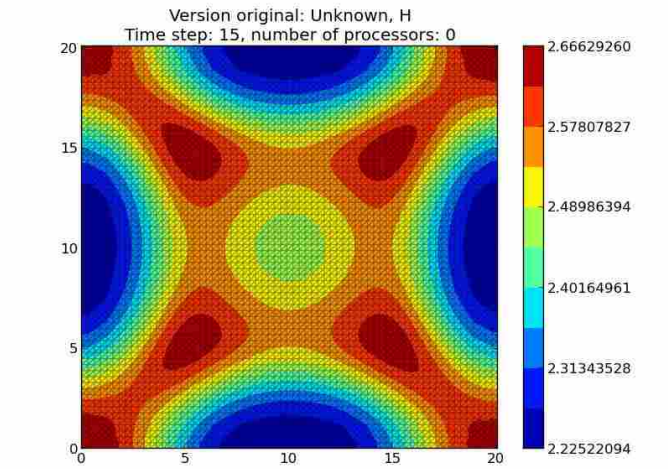
All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉

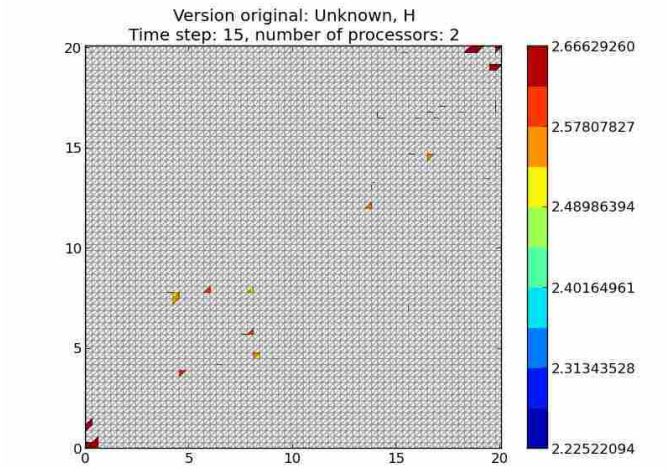
A white plot displays a non-reproducible value

NO numerical reproducibility!

time step = 15



Sequential

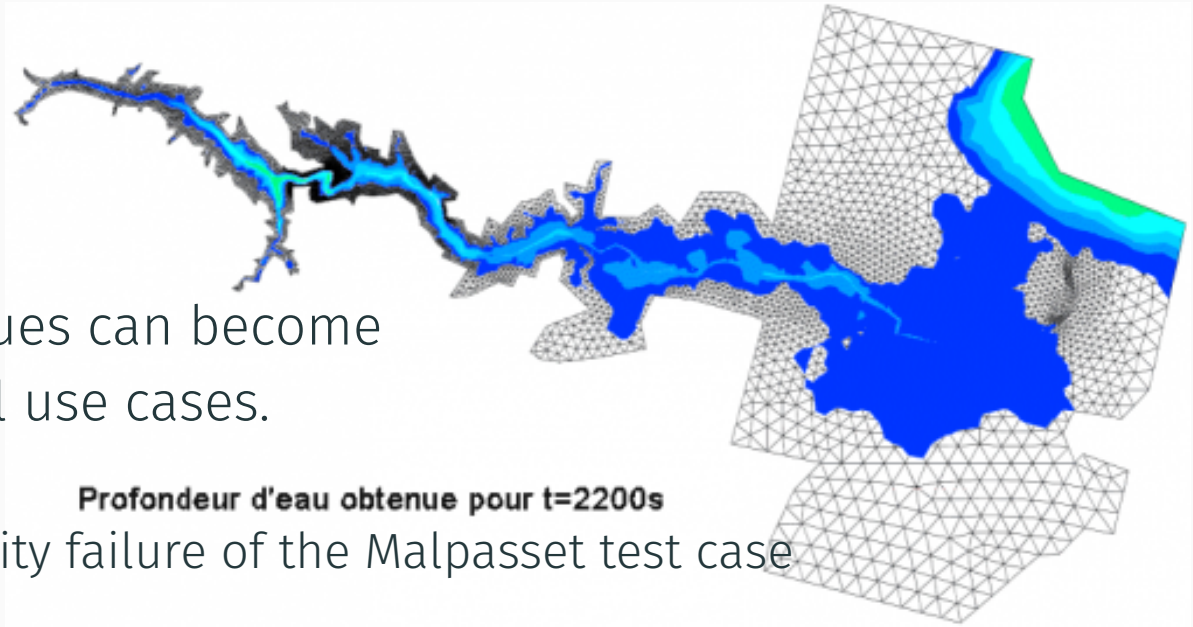


Parallel $p = 2$

Courtesy of P. Langlois and R. Nheili

All I care about is the algorithm output

Did I mention we have parallel machines nowadays? 😉



These numerical issues can become quite harmful in real use cases.

TABLE 1.1: Reproducibility failure of the Malpasset test case

	The sequential run	a 64 procs run	a 128 procs run
depth H	0.3500122E-01	0.2748817E-01	0.1327634E-01
velocity U	0.4029747E-02	0.4935279E-02	0.4512116E-02
velocity V	0.7570773E-02	0.3422730E-02	0.7545233E-02

Numerical reproducibility?: Approximations in the model, in in the algorithm, in its implementation, in its execution.

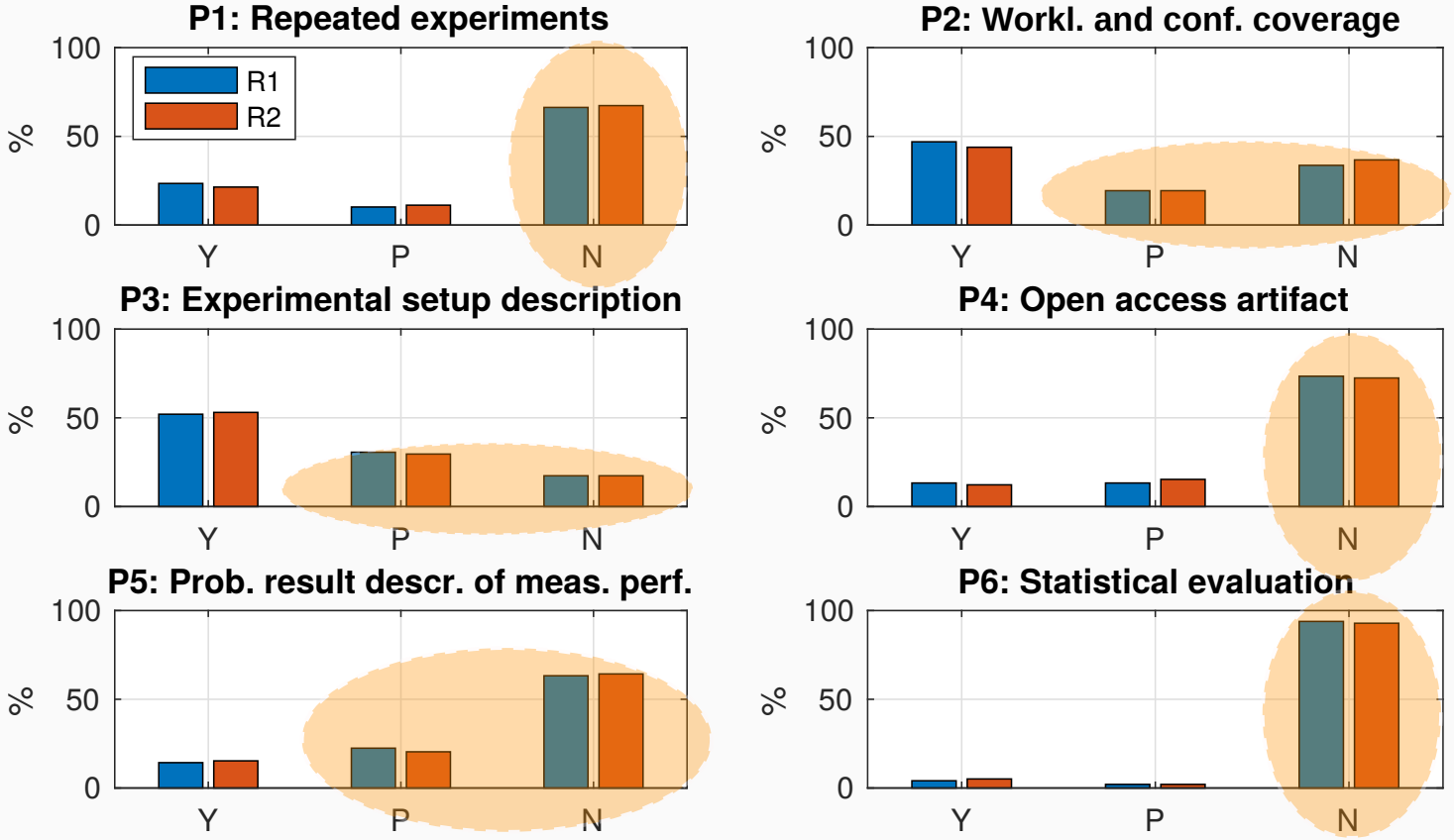
The whole chain needs to be revisited.

Courtesy of P. Langlois and R. Nheili

Toward Reproducible Computer Science Experiments ?

Key Concerns for our Community (Room for Improvement)

- Awareness of Experiments and Statistics *How are cloud performance currently obtained and reported?*, March 2019

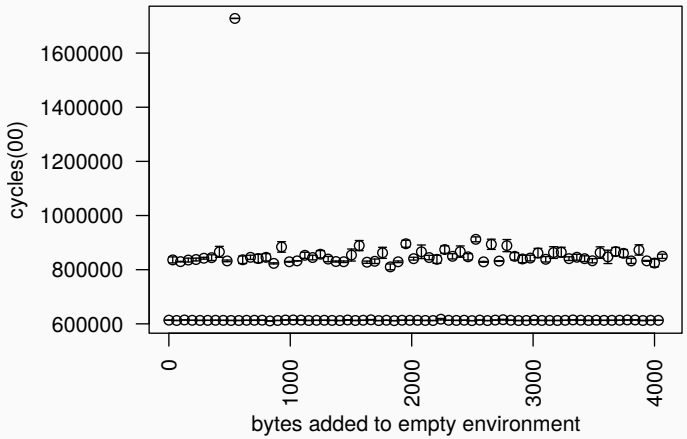


Key DoE principles: Replicate to increase reliability. Randomize to reduce bias ⇝ Evaluate statistical confidence.

Taming the Influence of Memory Layout

Remember Mytkowicz et al. *Producing wrong data without doing anything obviously wrong!* ACM SIGPLAN Not. 44(3), March 2009

> *changing the size of environment variables can trigger performance degradation as high as 300%; simply changing the link order of object files can cause performance to decrease by as much as 57%.*



C. Curtsinger and E. Berger *Stabilizer: Statistically Sound Performance Evaluation*, ASPLOS 2013.

> *Stabilizer forces executions to sample the space of memory configurations by repeatedly rerandomizing layouts of code, stack, and heap objects at runtime. [..] Re-randomization ensures that layout effects follow a Gaussian distribution, enabling the use of statistical tests like ANOVA.*

MPI Performance Characterization: The many biases

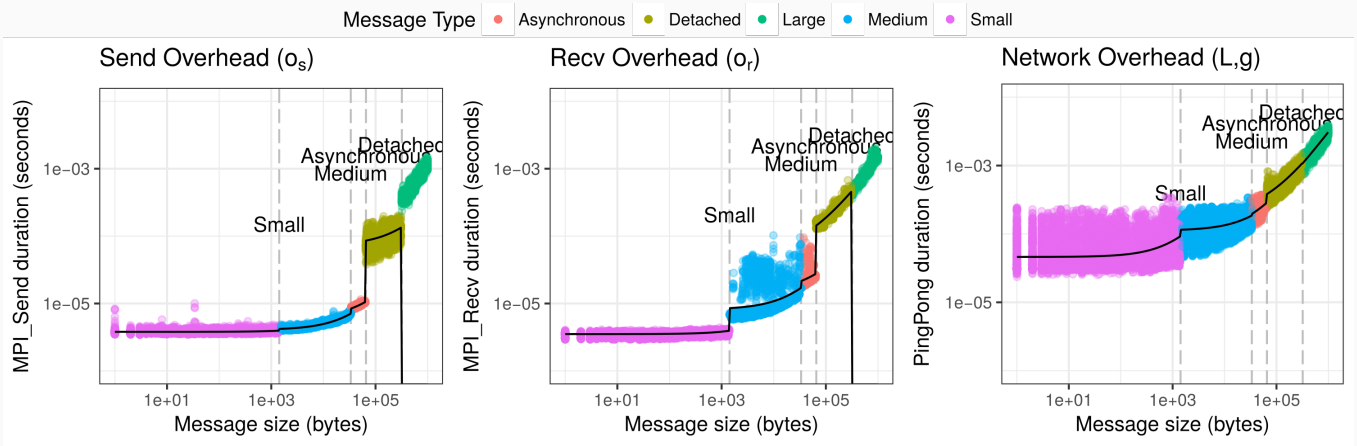
- Powers of two for message sizes ? Linear ?
- Sensitivity to temporal perturbations (in order `N_rep`)
- Breakpoint detection (increasing message size, minimal range length, ...)
- Outlier removal (assumes normality, ignores uncertainty of previous measurements)

Measurement proposal:

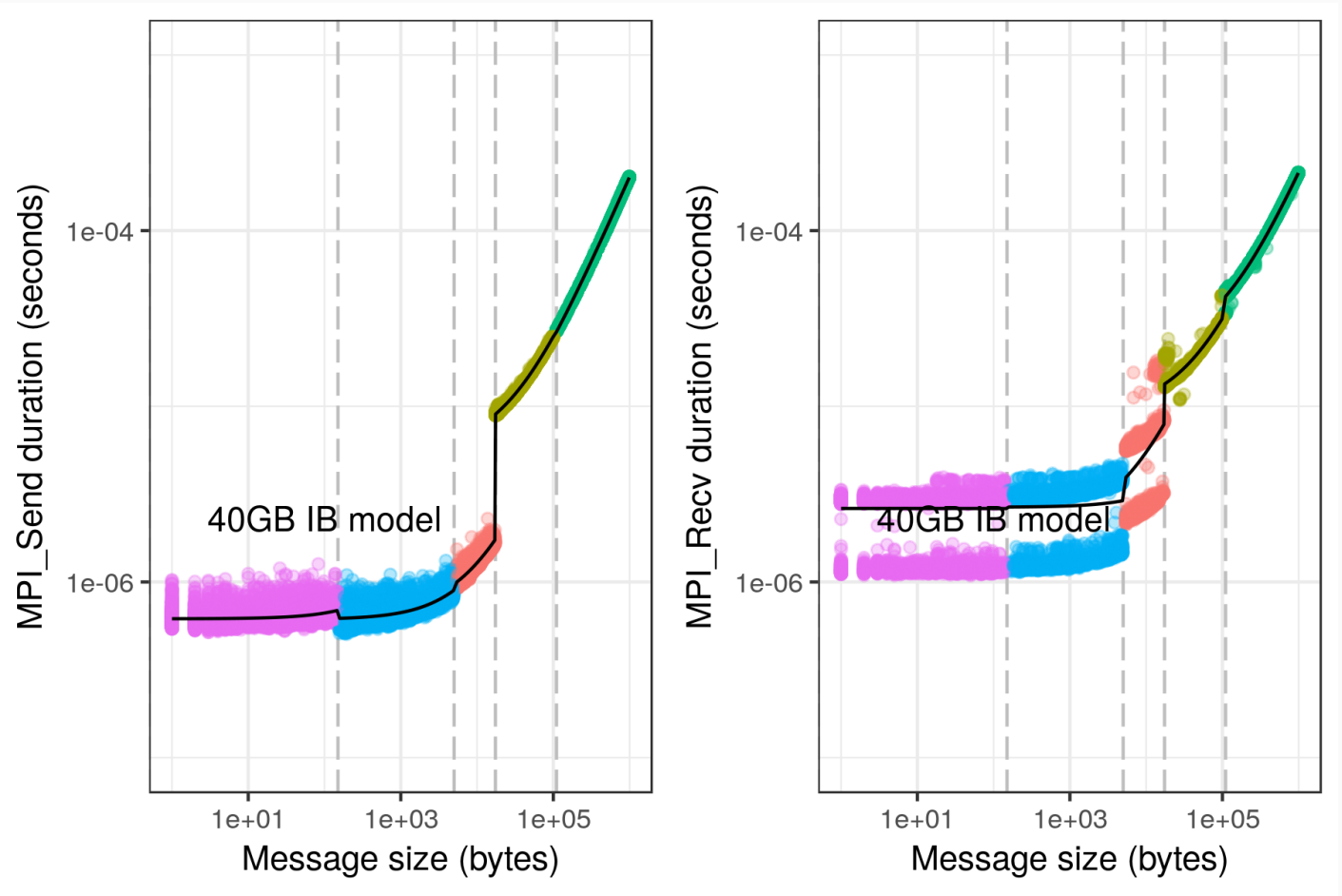
- Ping-Pong `data_size` $= 10^X$, where $X \sim \mathrm{Unif}(\log_{10}(a), \log_{10}(b))$.
- Record the time taken in every `MPI_Send` and `MPI_Receive` operation

Analysis proposal:

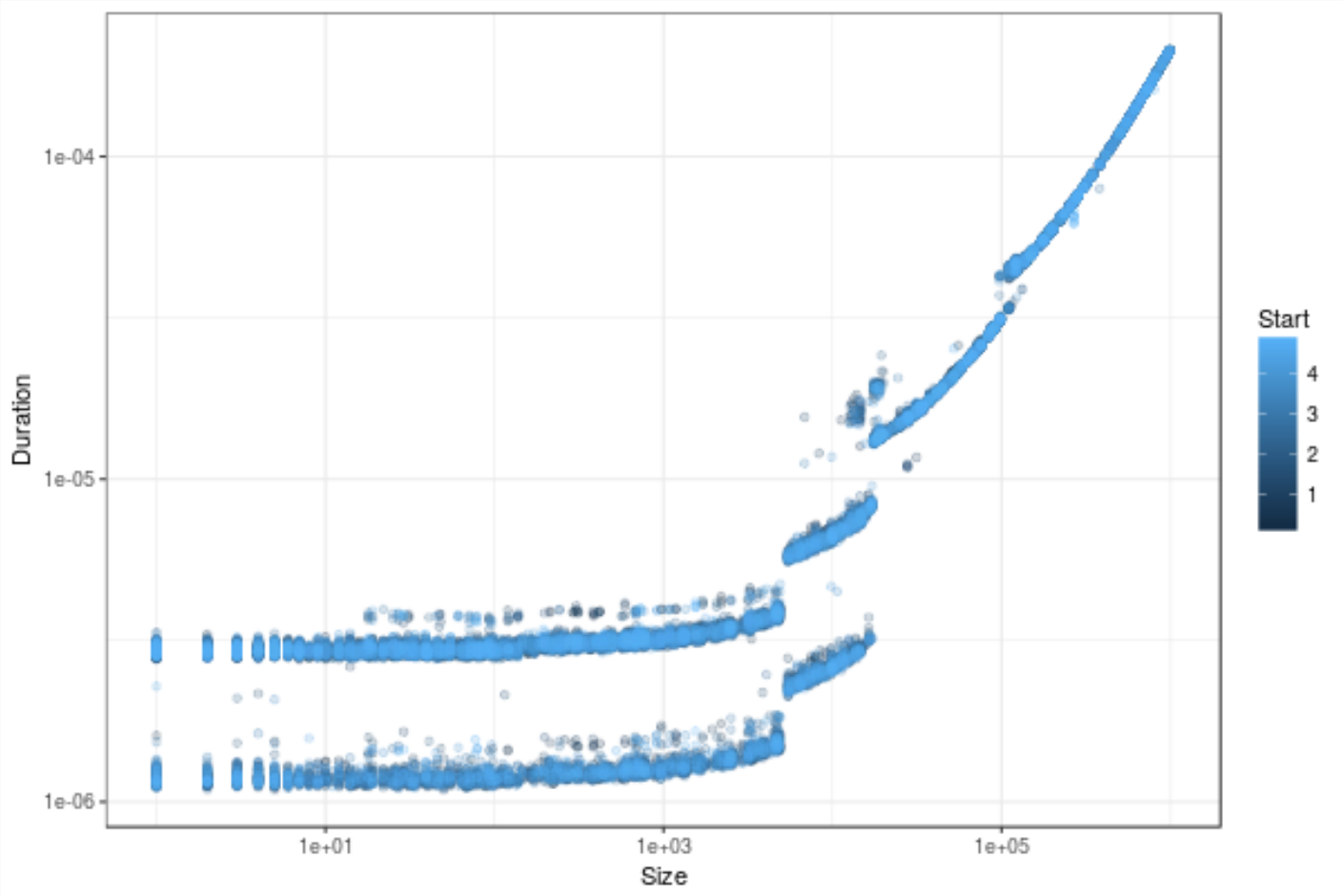
- Manually provided breakpoints
- Regression in R
- Inspect regression output and hypothesis (linearity, noise, "outliers")



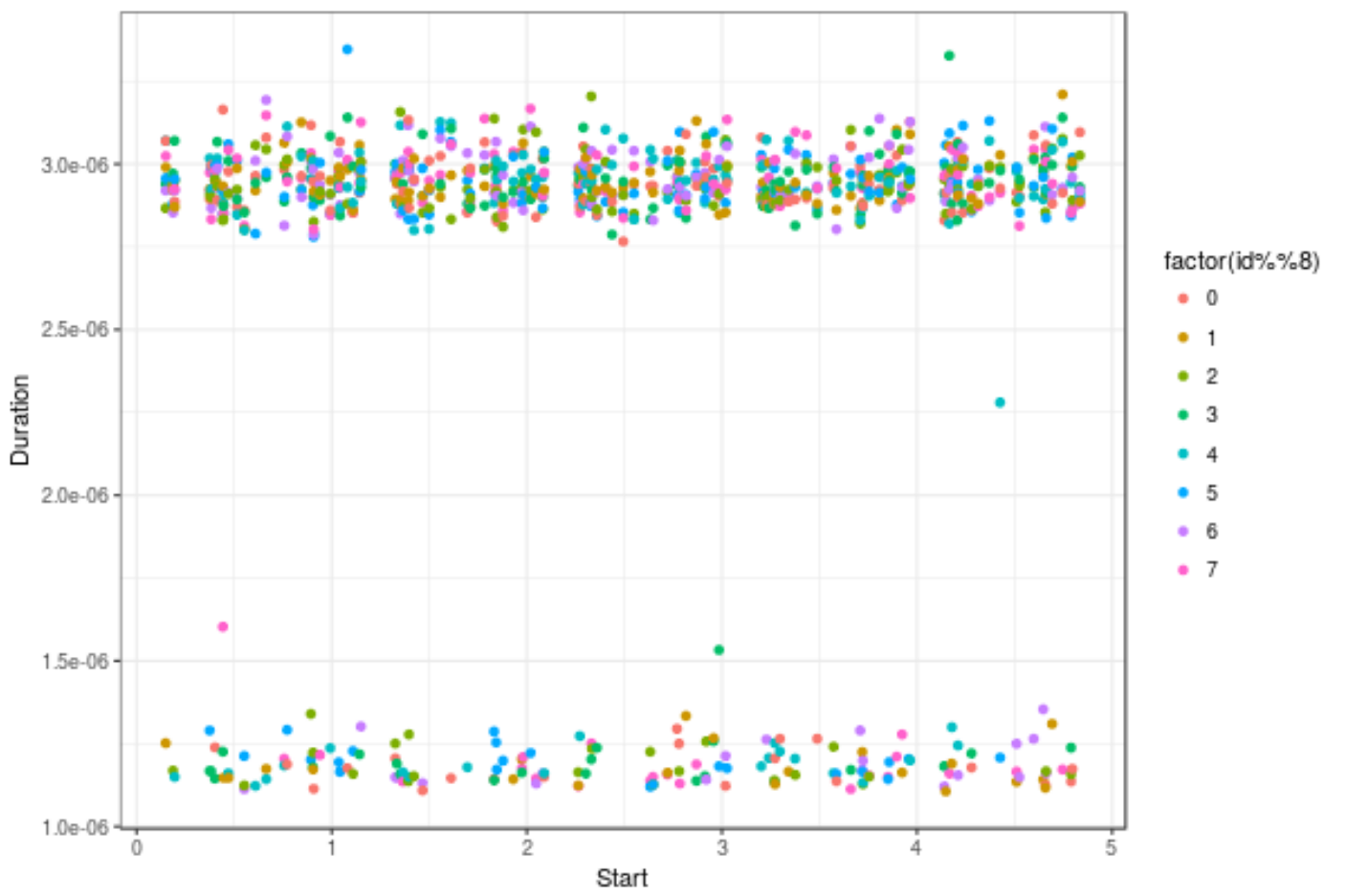
MPI Measurement: Randomization in action (Stampede@TACC)



MPI Measurement: Randomization in action (Stampede@TACC)

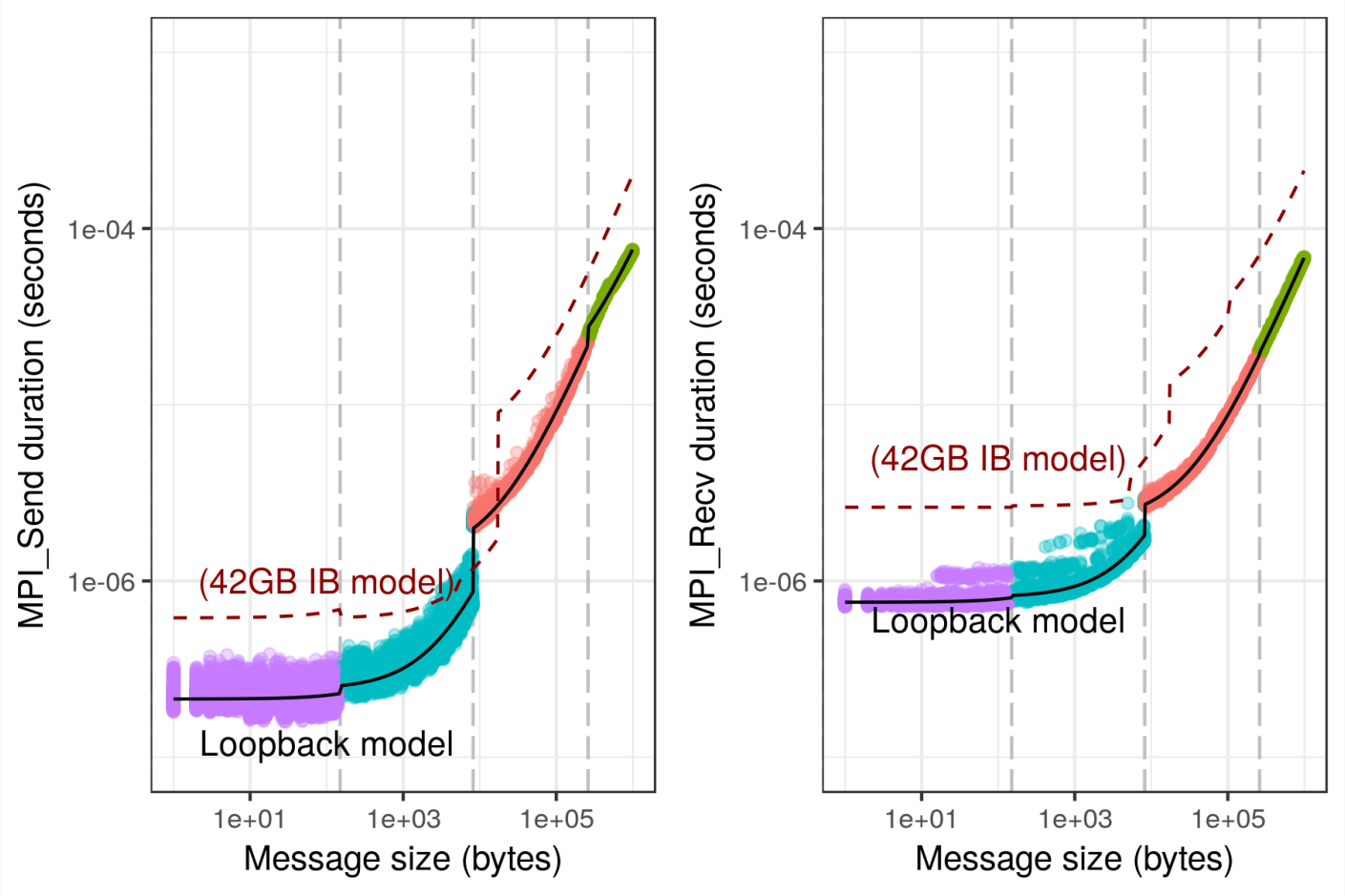


MPI Measurement: Randomization in action (Stampede@TACC)

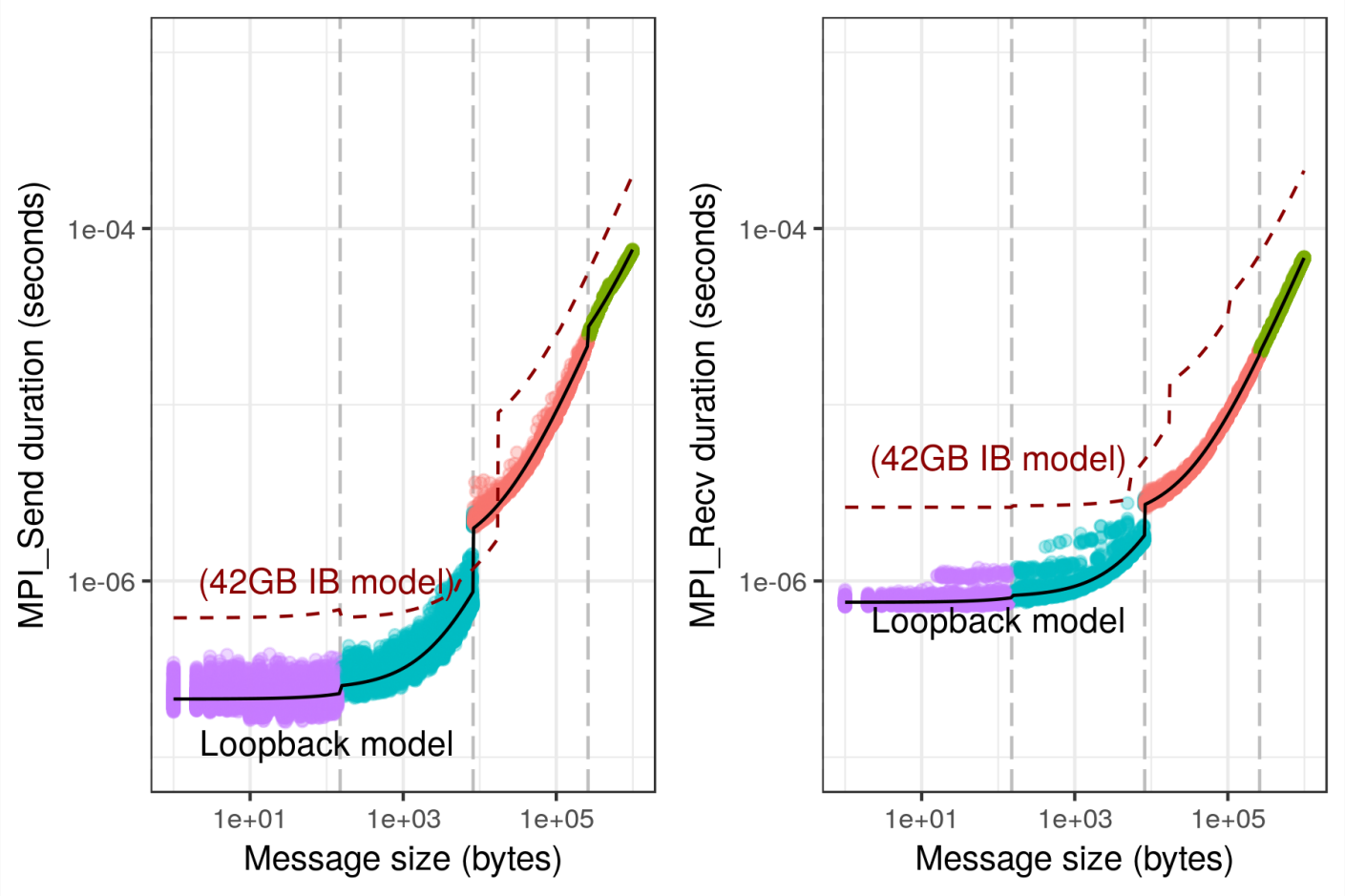


c558-[203,304]

MPI Measurement: Randomization in action (Stampede@TACC)

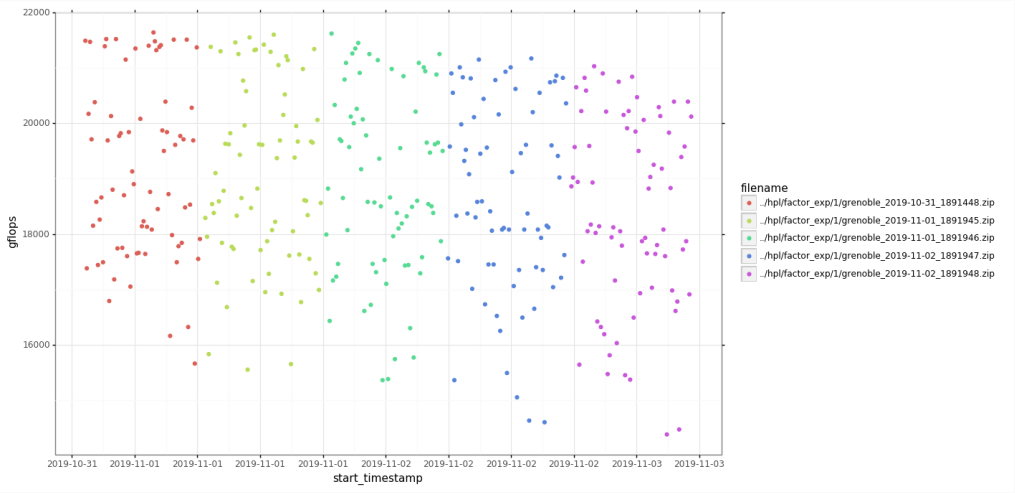


MPI Measurement: Randomization in action (Stampede@TACC)



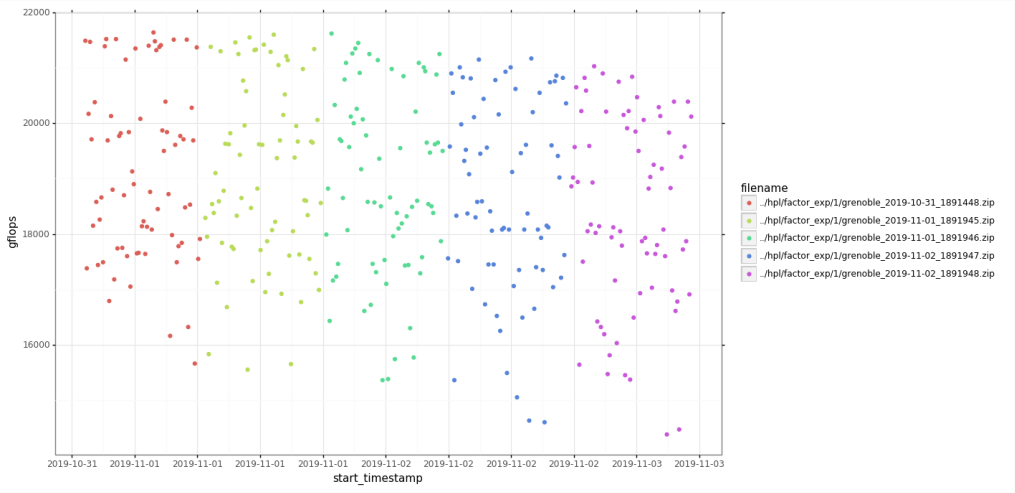
Even the simpler physical quantities can be very tricky to measure because our models and protocols are often naive

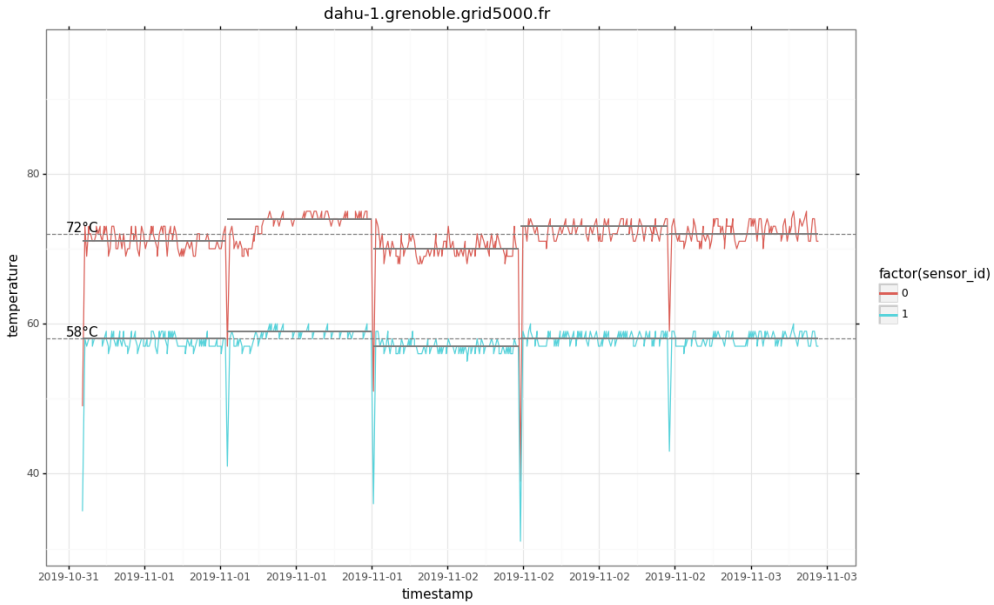
Randomization of a Factorial Design



Courtesy of T. Cornebize

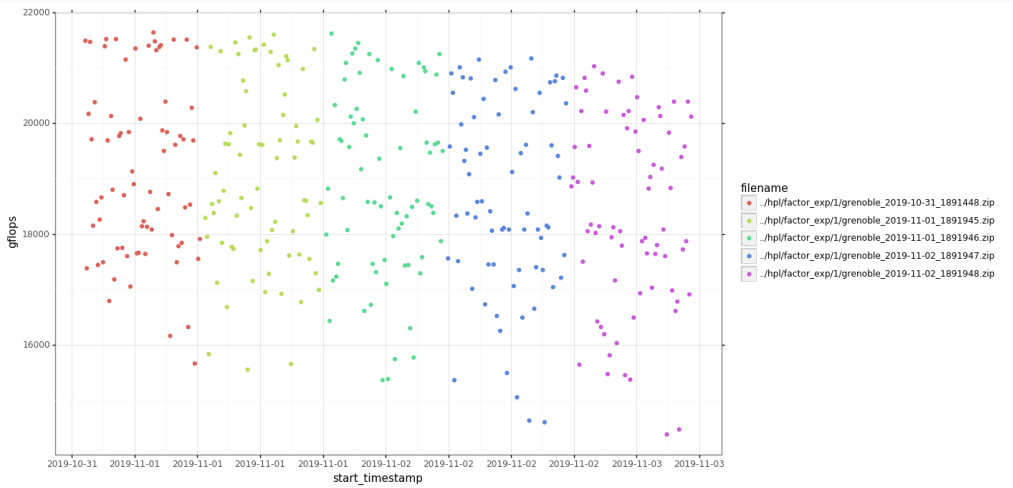
Randomization of a Factorial Design

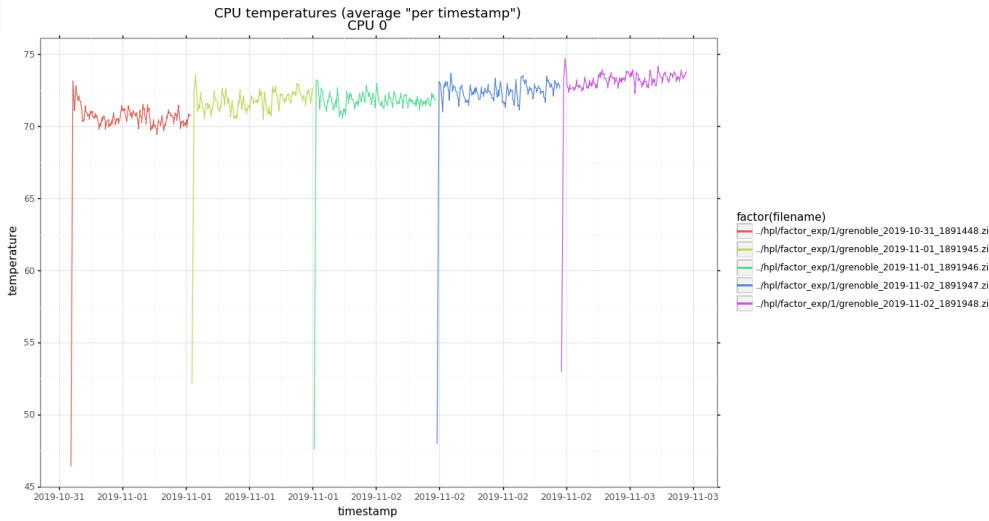


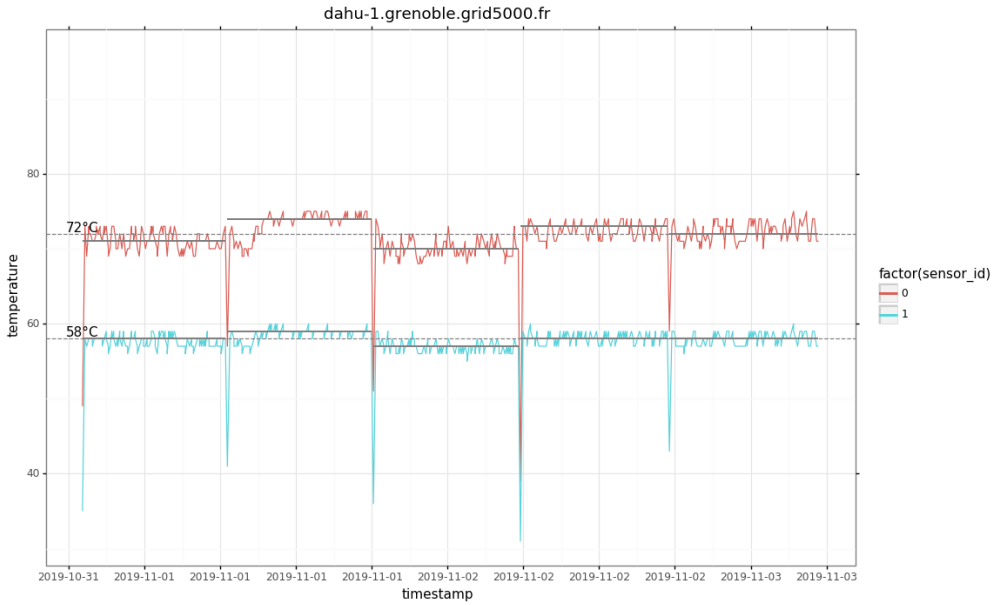


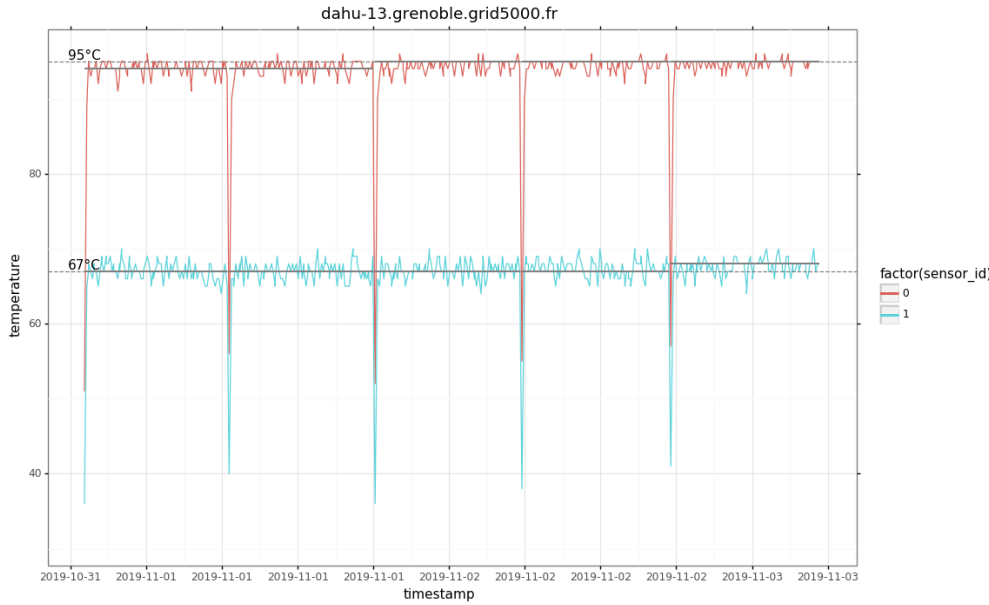
Courtesy of T. Cornebize

Randomization of a Factorial Design



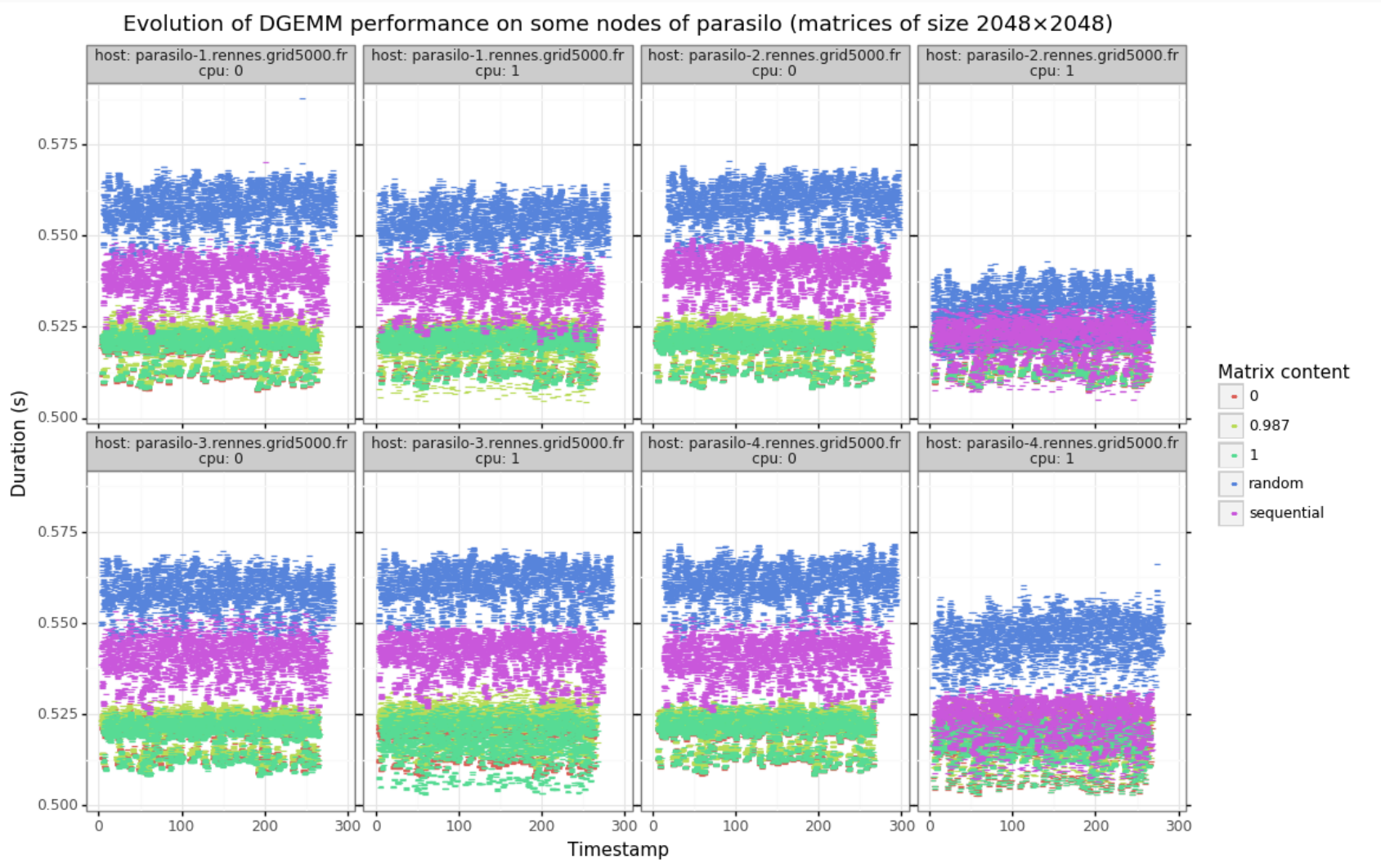






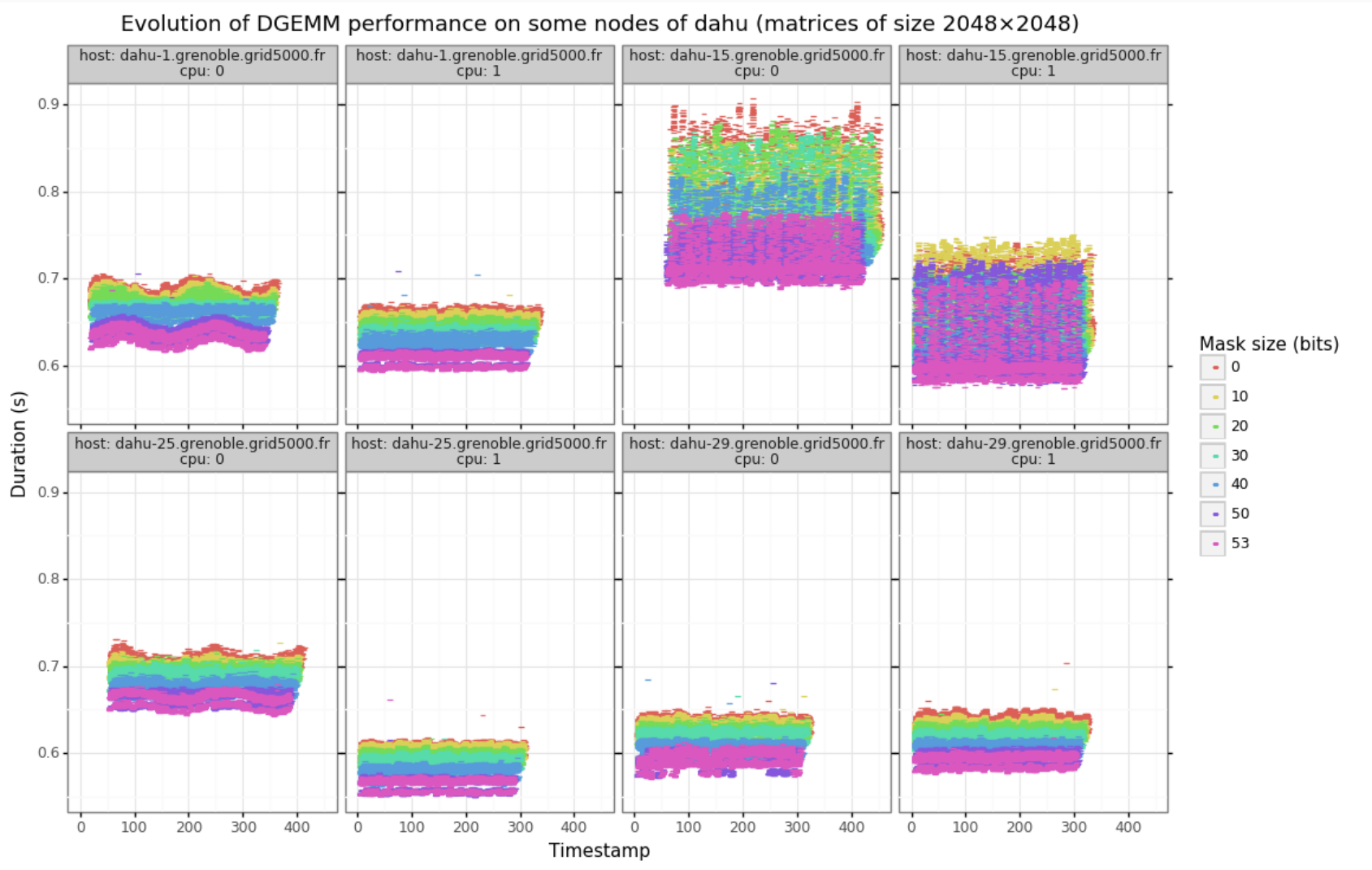
Courtesy of T. Cornebize

Randomize Everything! (Influence of Data)



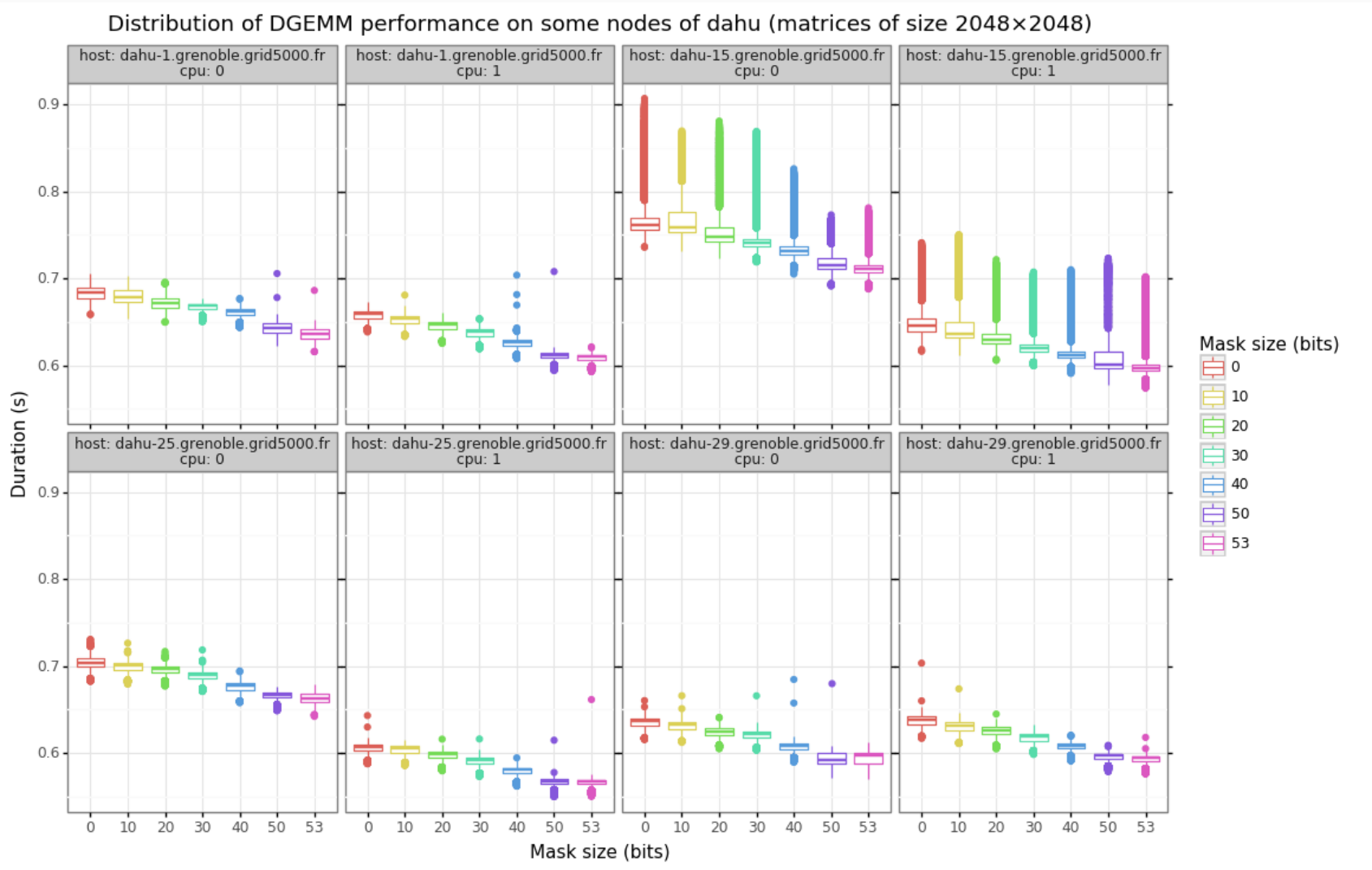
Courtesy of T. Cornebize

Randomize Everything! (Influence of Data)



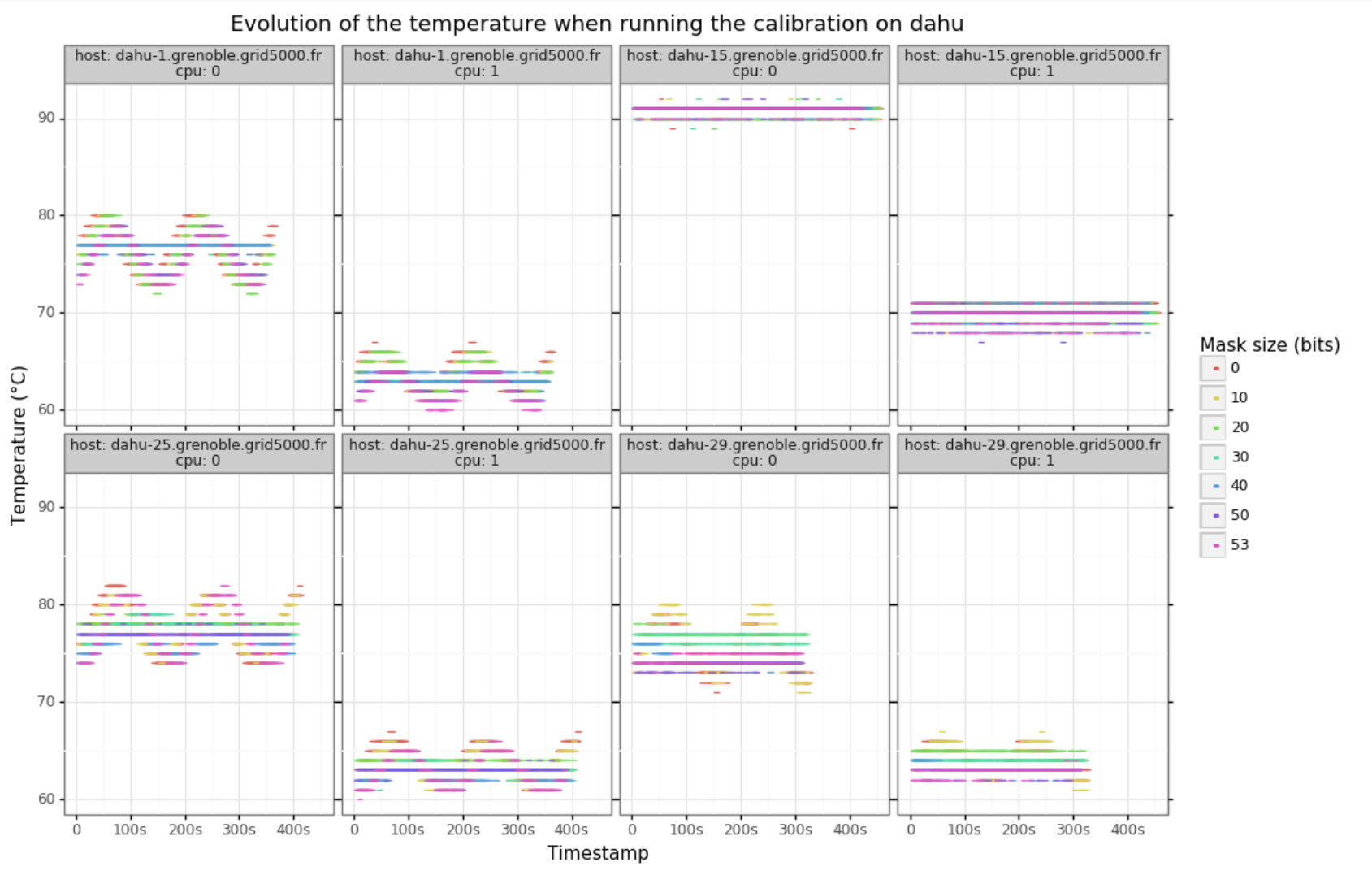
Courtesy of T. Cornebize

Randomize Everything! (Influence of Data)



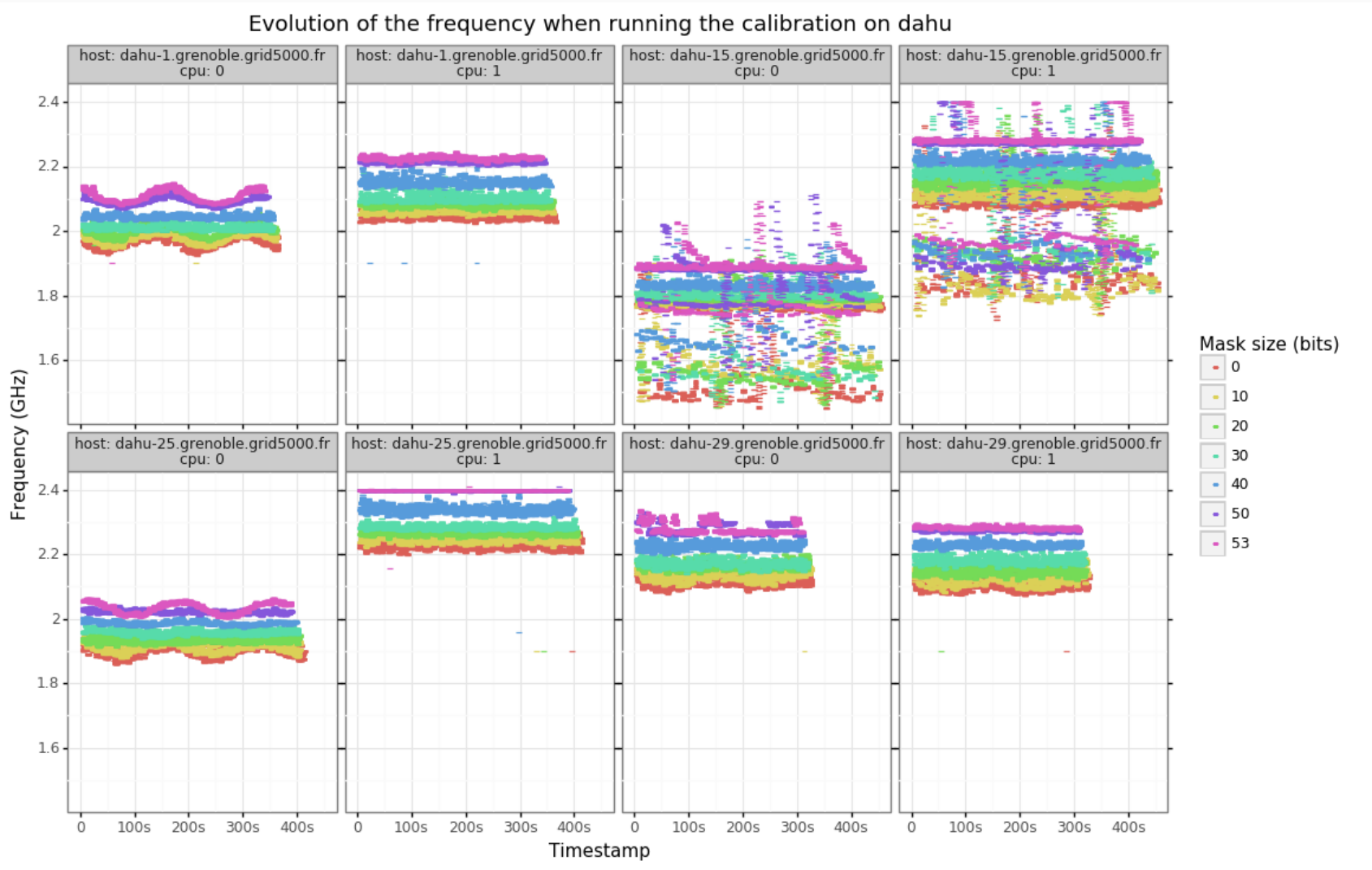
Courtesy of T. Cornebize

Randomize Everything! (Influence of Data)



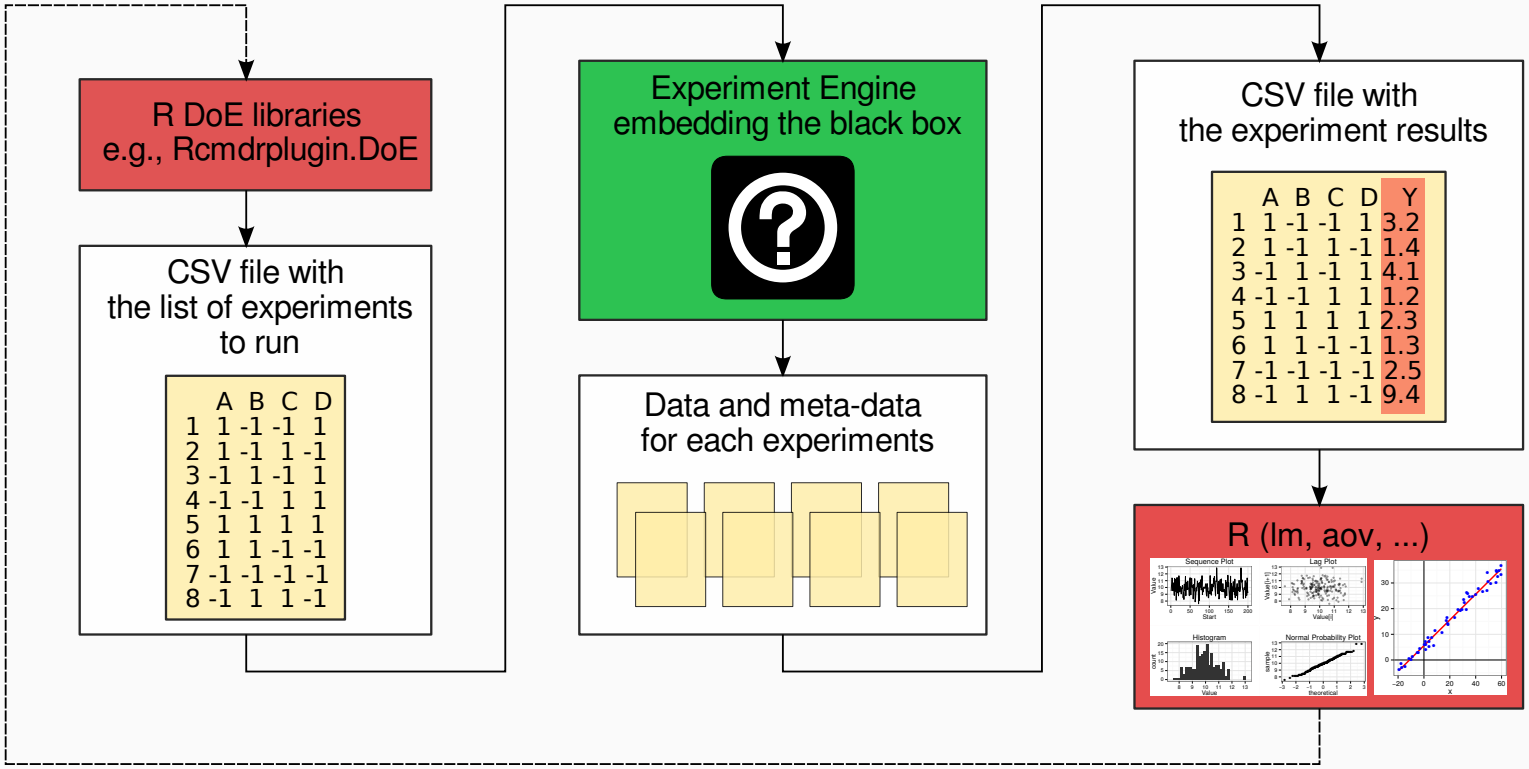
Courtesy of T. Cornebize

Randomize Everything! (Influence of Data)



Courtesy of T. Cornebize

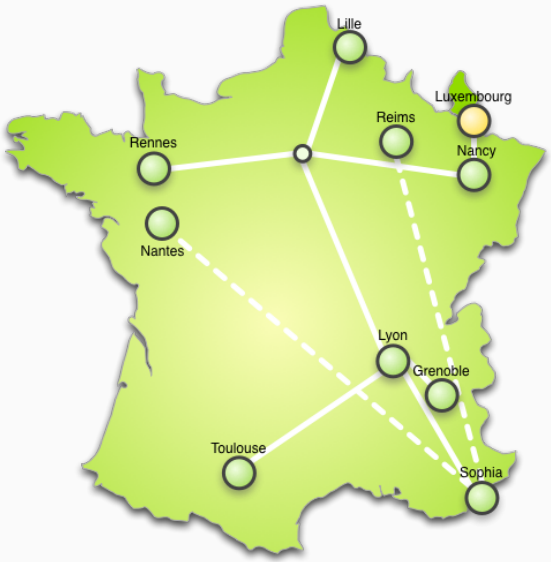
C.S. Experimental Methodology



1. A separation of concerns
   - Transparent Measurement Procedure and Analysis Procedure
2. Randomized and Designed Experiments allowing to both:
   - Check the model and Instanciate it
3. Careful recording of all experimental parameters

Experimental Testbeds

Good experimental practice and platforms FIT IoT-lab, G5K are world leading experimental infrastructures



- These platforms are fully configurable (bare-metal OS deployment, isolation, network reservation, ...)
- Share: the maintenance cost (keeping in pace with technology), practices for prototype platforms, experimental conditions, experimental engines

A few Experiment Management Tools

- Naive way: sh + ssh + ...
- Expo (2007-, G5K)
- XPflow (2012-, G5K)
- Execo (2013-, G5K)

} although nothing specific to G5K

- Plush (2006-, PlanetLab)
- OMF (2009-, Wireless testbeds and Planetlab)
- Splay (2008, distributed algorithm comparison)
- ...

They differ in the underlying paradigms and the platforms for which they have been designed

- A taxonomy of experiment management tools for distributed systems, T. Buchert, C. Ruiz , L. Nussbaum, O. Richard, FGCS, 2014

Expo

Grenoble (B. Videau, C. Ruis, O. Richard) http://expo.gforge.inria.fr/

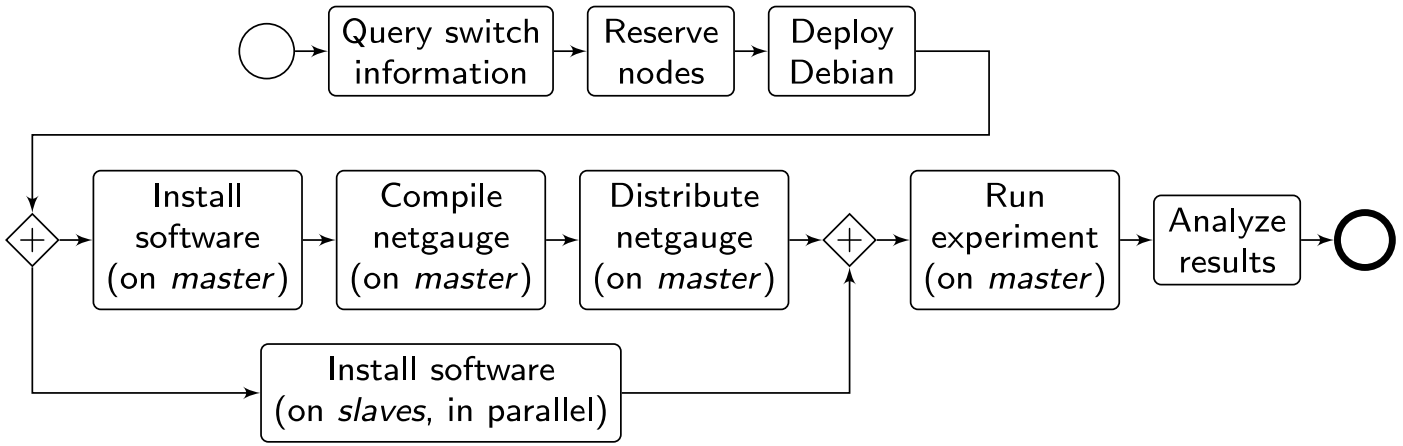
- A Ruby-based DSL for experiment management (based on taktuk, i.e., sh + ssh)
- Expo interacts with Planetlab and Grid5000 testbeds
- Resource and task abstractions, client-server organization, interactive or batch mode
- Native logging and archiving capabilities
  - every action performed on tasks, error flows, dates, ...
  - lets you know what was run, when, where and how

```
1 reserv=ExpoEngine::new(@connection)
2 reserv.site=["bordeaux","lille","luxembourg","nancy","sophia"]
3 reserv.resources=["nodes=50","nodes=10","nodes=4","nodes=4","nodes=30"]
4 reserv.name = "Expo Scalability"
5 reserv.walltime=600
6
7 reserv.run!
8 ptask $all, "hostname"
9 reserv.stop!
```

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy



Courtesy of T. Buchert

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy

```
process :exp do |site, switch|
    s = run g5k.switch, site, switch
    ns = run g5k.nodes, s
    r = run g5k.reserve_nodes,
        :nodes => ns, :time => '2h',
        :site => site, :type => :deploy
    master = (first_of ns)
    rest = (tail_of ns)
    run g5k.deploy,
        r, :env => 'squeeze-x64-nfs'
    checkpoint :deployed
    parallel :retry => true do
        forall rest do |slave|
            run :install_pkgs, slave
        end
        sequence do
            run :install_pkgs, master
            run :build_netgauge, master
            run :dist_netgauge,
                master, rest
        end
    end
    checkpoint :prepared
    output = run :netgauge, master, ns
    checkpoint :finished
    run :analysis, output, switch
end
```

Courtesy of T. Buchert

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy

```
process :exp do |site, switch|
    s = run g5k.switch, site, switch
    ns = run g5k.nodes, s
    r = run g5k.reserve_nodes,
        :nodes => ns, :time => '2h',
        :site => site, :type => :deploy
    master = (first_of ns)
    rest = (tail_of ns)
    run g5k.deploy,
        r, :env => 'squeeze-x64-nfs'
    checkpoint :deployed
    parallel :retry => true do
        forall rest do |slave|
            run :install_pkgs, slave
        end
        sequence do
            run :install_pkgs, master
            run :build_netgauge, master
            run :dist_netgauge,
                master, rest
        end
    end
    checkpoint :prepared
    output = run :netgauge, master, ns
    checkpoint :finished
    run :analysis, output, switch
end
```

Activity `:install_pkgs`

```
activity :install_pkgs do|node|
    log 'Installing packages on ', node
    run 'g5k.bash', node do
        aptget :update
        aptget :upgrade
        aptget :purge, 'mx'
    end
end
```

Courtesy of T. Buchert

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy

```
process :exp do |site, switch|
    s = run g5k.switch, site, switch
    ns = run g5k.nodes, s
    r = run g5k.reserve_nodes,
        :nodes => ns, :time => '2h',
        :site => site, :type => :deploy
    master = (first_of ns)
    rest = (tail_of ns)
    run g5k.deploy,
        r, :env => 'squeeze-x64-nfs'
    checkpoint :deployed
    parallel :retry => true do
        forall rest do |slave|
            run :install_pkgs, slave
        end
        sequence do
            run :install_pkgs, master
            run :build_netgauge, master
            run :dist_netgauge,
                master, rest
        end
    end
    checkpoint :prepared
    output = run :netgauge, master, ns
    checkpoint :finished
    run :analysis, output, switch
end
```

Activity `:build_netgauge`

```
activity :build_netgauge do |master|
    log "Building netgauge on #{master}"
    run 'g5k.copy', NETGAUGE, master, '~'
    run 'g5k.bash', master do
        build_tarball NETGAUGE, PATH
    end
    log "Build finished."
end
```

Courtesy of T. Buchert

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy

```
process :exp do |site, switch|
    s = run g5k.switch, site, switch
    ns = run g5k.nodes, s
    r = run g5k.reserve_nodes,
        :nodes => ns, :time => '2h',
        :site => site, :type => :deploy
    master = (first_of ns)
    rest = (tail_of ns)
    run g5k.deploy,
        r, :env => 'squeeze-x64-nfs'
    checkpoint :deployed
    parallel :retry => true do
        forall rest do |slave|
            run :install_pkgs, slave
        end
        sequence do
            run :install_pkgs, master
            run :build_netgauge, master
            run :dist_netgauge,
                master, rest
        end
    end
    checkpoint :prepared
    output = run :netgauge, master, ns
    checkpoint :finished
    run :analysis, output, switch
end
```

Activity `:dist_netgauge`

```
activity :dist_netgauge do |m, s|
    master, slaves = m, s
    run 'g5k.dist_keys', master, slaves
    run 'g5k.bash', master do
        distribute BINARY,
            DEST, 'localhost', slaves
    end
end
```

Courtesy of T. Buchert

XPflow

Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/

- Another Ruby-based DSL (Domain Specific Language)
  - Resources, process, and activities
- Top-down vs. than bottom-up: business process management
  - Cope with failures through snapshots and retry policy

```
process :exp do |site, switch|
    s = run g5k.switch, site, switch
    ns = run g5k.nodes, s
    r = run g5k.reserve_nodes,
        :nodes => ns, :time => '2h',
        :site => site, :type => :deploy
    master = (first_of ns)
    rest = (tail_of ns)
    run g5k.deploy,
        r, :env => 'squeeze-x64-nfs'
    checkpoint :deployed
    parallel :retry => true do
        forall rest do |slave|
            run :install_pkgs, slave
        end
        sequence do
            run :install_pkgs, master
            run :build_netgauge, master
            run :dist_netgauge,
                master, rest
        end
    end
    checkpoint :prepared
    output = run :netgauge, master, ns
    checkpoint :finished
    run :analysis, output, switch
end
```

Activity :netgauge

```
activity :netgauge do |master, nodes|
    log "Running experiment..."
    out = run 'g5k.bash', master do
        cd PATH
        mpirun nodes, "./netgauge"
    end
    log "Experiment done."
end
```

Courtesy of T. Buchert

Conclusion

Reproducible Research = Rigor and Transparency

To err is human.

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

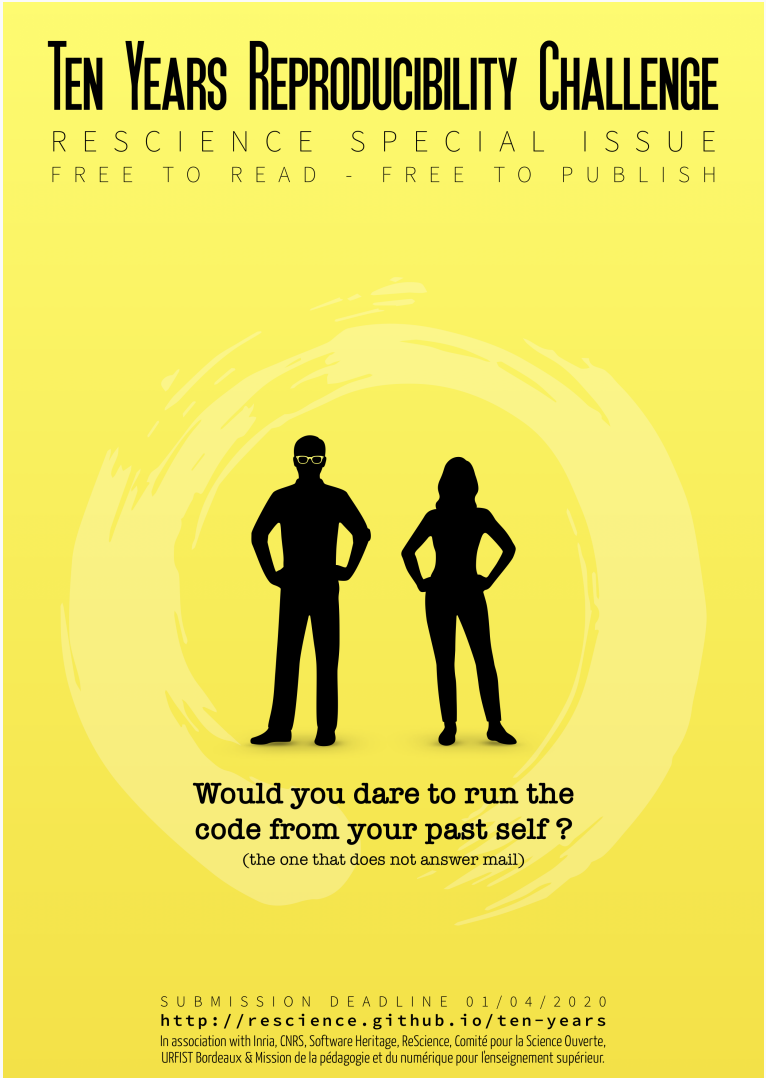
Reproducible Research = Rigor and Transparency

To err is human.

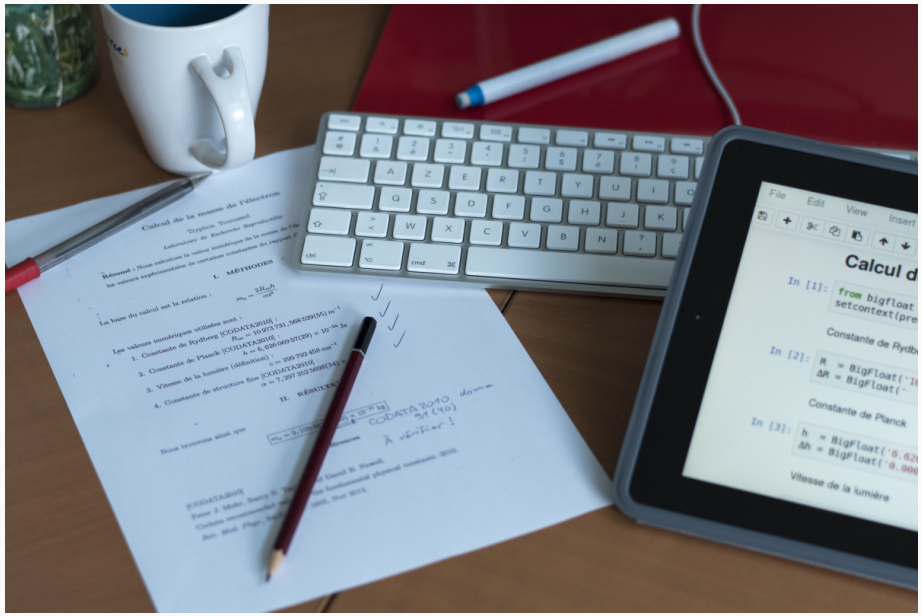
Good research requires time and resources

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
4. **Prepare the Future:** Toward literate experimentation?
   - Reuse, reuse, reuse!
   - Shared and controled testbeds (e.g., Grid'5000/SILECS)
   - How to share Experiments ?

Some Advertising



http://rescience.github.io/ten-years/



3rd Edition: ≈ Feb. 2020

A new MOOC: "Advanced RR"

- Software environment control (Docker)
- Scientific workflow (snakemake)
- Managing data (HDF5, archiving)

October 2020 ?