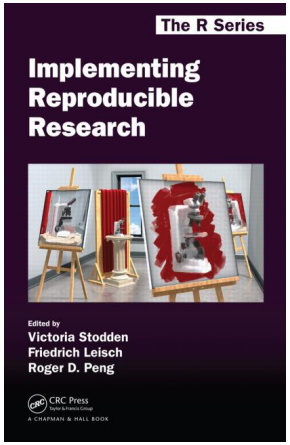
Scientific workflows for computational reproducibility in the life sciences

Action ReproVirtuFlow GDR MaDICS

Sarah Cohen-Boulakia
Université Paris-Sud, Laboratoire de Recherche en Informatique
CNRS UMR 8623, Université Paris-Saclay, Orsay, France

Reproducibility



V. Stodden *et al.*

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

Context, Challenges

Computational reproducibility crisis

Increasing number of irreproducible results
- Even published in high IF venues
- Not (always) deliberately

Various scientific domains
- Consequences may be huge (preclinical studies…)

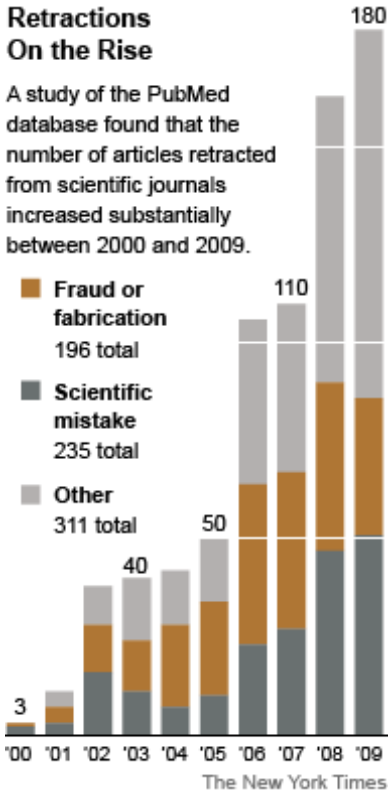
Major challenge
- The cost of irreproducible preclinical studies have been evaluated to >$10 Billions per year (USA)

Becoming mandatory
- NSF projects, editors, ANR…

→ ReproVirtuFlow Action created
GDR CNRS MaDICS (2014)

47/53 "landmark" publications could not be replicated

[Begley, Ellis Nature, 483, 2012]



Aims of our Action

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

Interdisciplinarity
Databases, Knowledge Representation, Semantic Web, Algorithmics, Graphes, Operating Systems, Compilation, Engineering, Langages… Bioinformatics, Molecular Biology, Plant Biology, Biomedical…

Members

IRISA Univ. Rennes

Univ. CHU Nantes

Centre de Biophysique Moléculaire, CNRS Orléans

IRD, CIRAD, INRA, Inria, Univ. Montpellier

Univ. Lyon 1 LIRIS

LRI Univ. Paris Sud

CDS, Center for Data Science Saclay

Institut Francais Bioinformatique

Institut Pasteur, Paris

Lamsade Univ. Paris Dauphine

LIG (Grenoble)

Use cases from the bioinfo domain

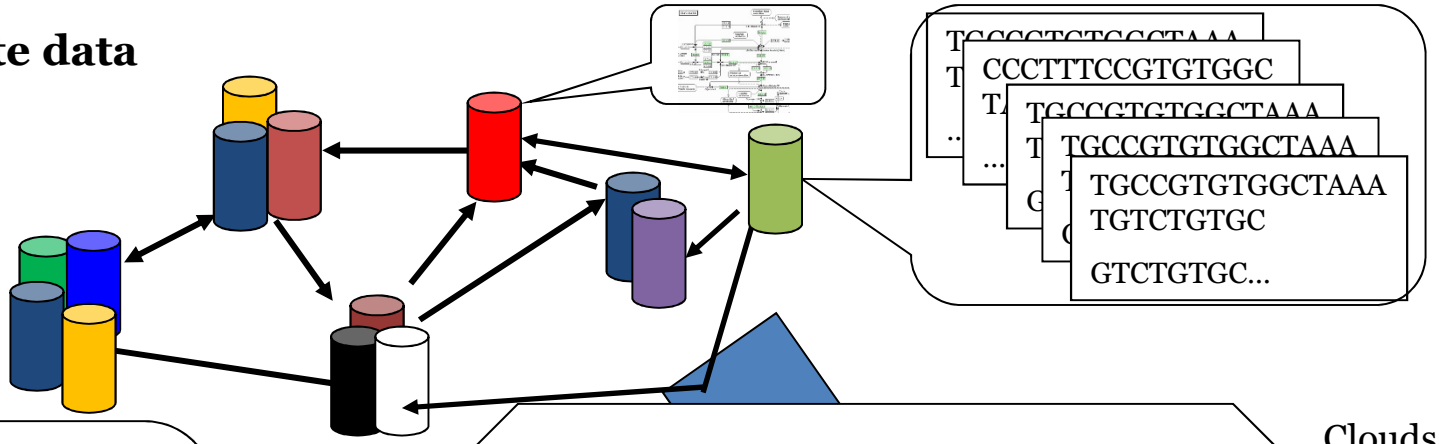
Bioinformatics analysis

Public and private data sources

Distributed

Heterogeneous

> 1,500



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

What is the difference between these experiments?

Binarization Water Use Efficiency
Segmentation
Python Java
Web services

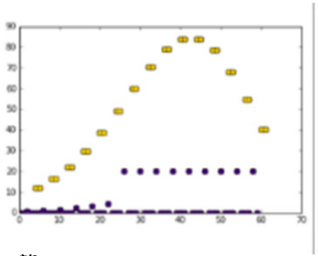
Clouds
Grids
Clusters
Desktop

Tools

Distributed

Heterogeneous

To be chained



Take Home Message

Compared to 20 years ago…

The number and diversity of the data sources has increased a lot

> 1,500 public databases (NAR databases issue)

Need for data provenance to determine data quality

The complexity of the pipelines to be designed has increased a lot

2,000 tools in bio.tools.org (repository of tools)

Need to combine tools to design pipelines

Need for process (workflow, tools) provenance to determine data quality

→Increase in the heterogeneity of data

+ Increase in the complexity of analysis pipelines

+ *Increase in the need to publish…*

= increasing difficulties to reproduce experiments!

Scripts and reproducibility?
Good practices

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

But scripts do not allow to

Distinguish between steps of the analysis
- piece of codes, methods/functions

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

Emphasize the major steps of the analysis
Provide solution for data management
- Naming convention for produced files, storage...

→ Scripts are difficult to share, exchange and reuse (repurpose)

Outline

Context

Scientific workflows

- Scientific workflow systems
- Repositories of scientific workflows
- Companion tools to ensuring properly rerun
- Reprohackathons

Lessons learnt on Scientific workflows and reproducibility

- Levels of reproducibility with scientific workflows
- Reproducibility-friendly features
- Open problems

Conclusion

Scientific workflow systems

SWFS = “Data analysis pipeline”

Data flow driven

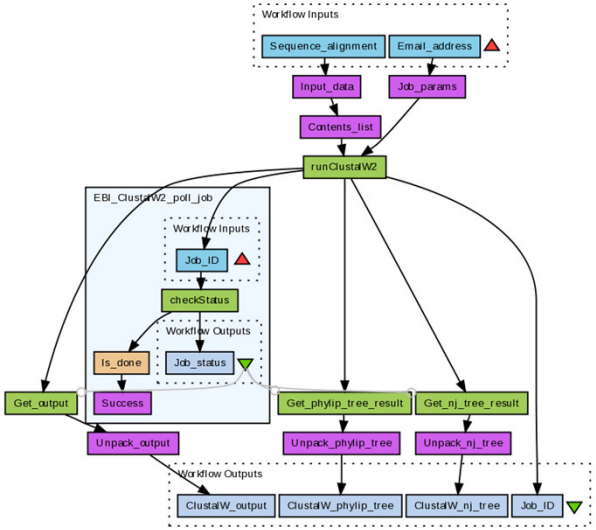
WF specification: connected tools

steps of the analysis

WF execution: data consumed and produced during tools execution tacked

Provenance modules

data management

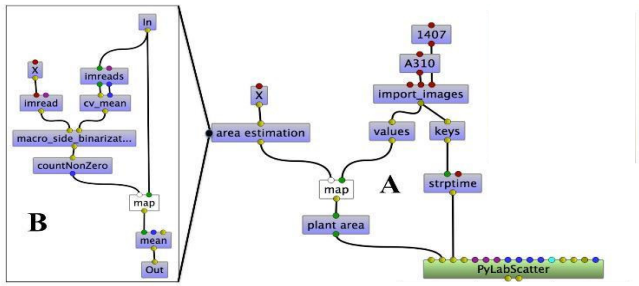


SWFS manages scheduling, logging, recovery …

May be equipped with GUI

Several systems available

- Galaxy, NextFlow, SnakeMake, OpenAlea…



Scientific Workflow Discovery

Pose keyword query

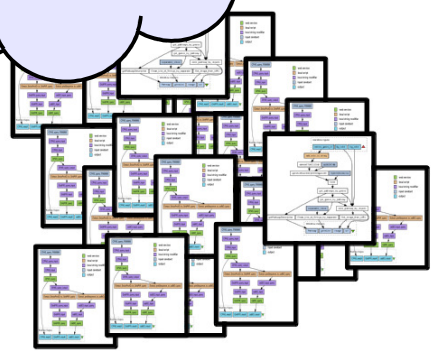
Using workflow repositories

Kepler

Search in textual annotations

myexperiment

?



List of 10s or 100s of workflows

Find appropriate workflows

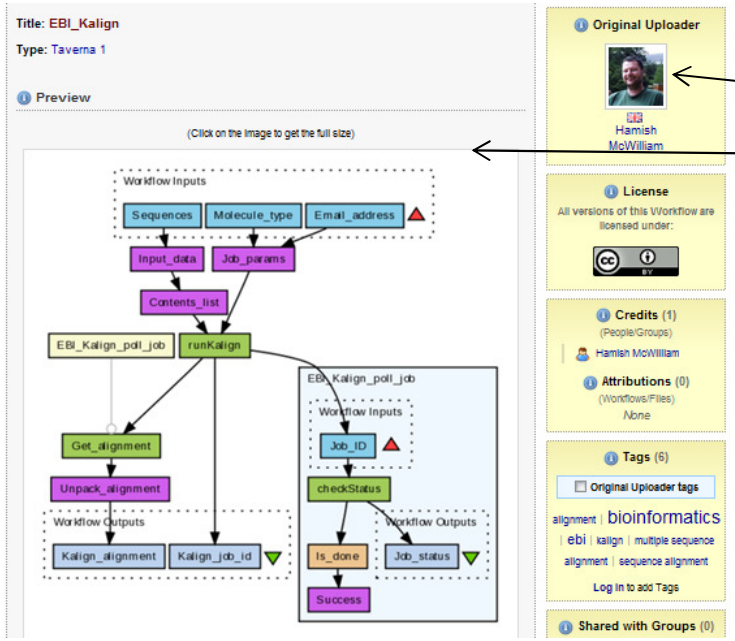
Reuse scientific workflow

myExperiment repository

myExperiment.org

Looking for workflows

- By keywords
  - BioAID… workflow
  - Inspecting meta-data (author, favourited by, history…)
- By authors
- By group
- …



- Conceptor
- Workflow
- Annotations
- …

What else do we need to reach *computationnal* reproducibility?…

We have stored the scripts (or workflows)

We have the exact data sets…

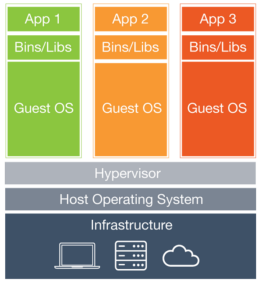
Capturing the programming environment

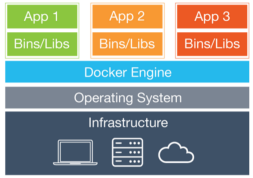
Ensuring your workflow has everything it needs to run

Libraries, dependencies…

Virtual machines capture the programming environment

Containers solutions





- package an application
  - with all of its dependencies
  - into a standardized unit for software development

include the application and all of its dependencies

- but share the kernel with other containers
- They
  - are not tied to any specific infrastructure;
  - run on any computer, on any infrastructure and in any cloud

Lighter solution than classical VM

➔ BioContainers: a registry of containers!

Outline

- Context
- Scientific workflows
  - Scientific workflow systems
  - Repositories of scientific workflows
  - Companion tools to ensuring properly rerun
  - Reprohackathons
- Lessons learnt on Scientific workflows and reproducibility
  - Levels of reproducibility with scientific workflows
  - Reproducibility-friendly features
  - Open problems
- Conclusion

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

The first edition of ReproHackathon

- RNA-Seq data from patients with uveal melanoma: genes involved
- Divergent published results…
- 25 participants (IGRoussy, Curie, Pasteur, Saclay, Paris, Nantes, …)

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

Workflow Systems : SnakeMake, NextFlow, Galaxy…
Executed in the Cloud@IFB

+ Reprohackathon 2 in Lyon, July 2018
Phylogenetics

+ (coming) Reprohackathon 3
Montpellier Nov 2019
Plant phenotyping analysis

Outline

Context

Scientific workflows

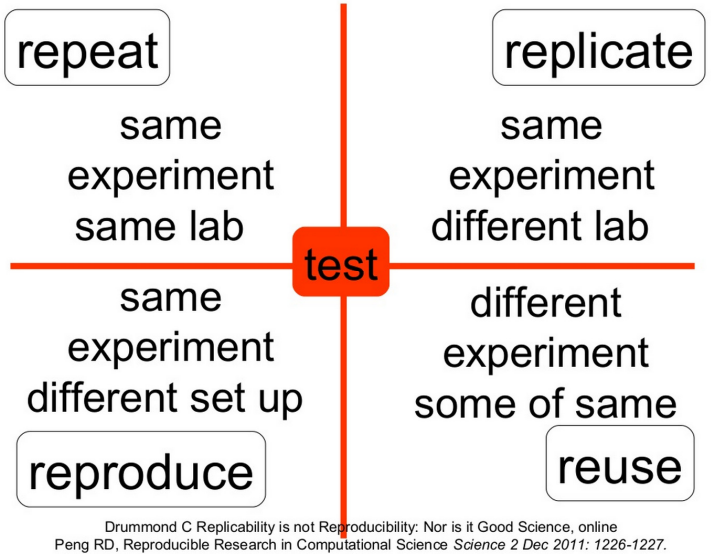
- Scientific workflow systems
- Repositories of scientific workflows
- Companion tools to ensuring properly rerun
- Reprohackathons

Lessons learnt on Scientific workflows and reproducibility

- Levels of reproducibility with scientific workflows
- Reproducibility-friendly features
- Open problems

Conclusion

Levels of computational reproducibility



Repeat

- *Redo*: exact same context
- Same workflow, execution setting, environement
- Identical *output*

→Aim = proof for reviewers ☺

3 ingredients

Workflow Specification

Chained Tools

Workflow Execution

Input data and parameters

Environment

OS/librairies …

Replicate

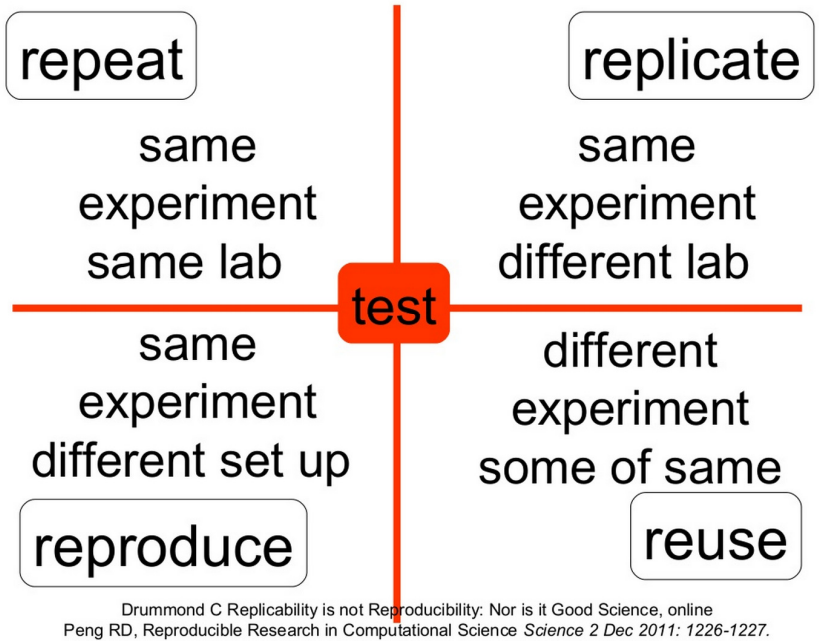
- Variation allowed in the workflows, execution setting, environement
- Similar *output*

→ Aim = robustness

A continuum of possibilities

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

Outline

Context

Scientific workflows

- Scientific workflow systems
- Repositories of scientific workflows
- Companion tools to ensuring properly rerun
- Reprohackathons

Lessons learnt on Scientific workflows and reproducibility

- Levels of reproducibility with scientific workflows
- Reproducibility-friendly features
- Open problems

Conclusion

Reproducibility-friendly features in scientific workflows

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

Workflow 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, myexperiement…) → reuse

Execution

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

Presentation (interactivity with the results/provenance, notebooks) → replicate … reuse

Annotations → reuse

Reproducibility-friendly features in scientific workflows (cont.)

Environment (companion tools)

Ability to run workflows within a given environment → repeat (… reuse)

Virtual machines capture the programming environment

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

Lighter solutions (containers)

- Only capture software dependencies
- Docker, Rocket, OpenVZ, LXC, Conda

Capturing the command-line history, input/output, specification

CDE, ReproZip (NewYork University)

Outline

Context

Scientific workflows

- Scientific workflow systems
- Repositories of scientific workflows
- Companion tools to ensuring properly rerun
- Reprohackathons

Lessons learnt on Scientific workflows and reproducibility

- Levels of reproducibility with scientific workflows
- Reproducibility-friendly features
- Open problems

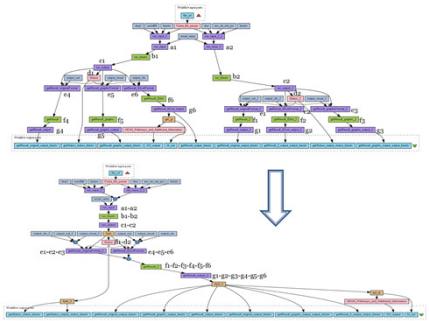
Conclusion

Open Challenges

Querying workflow repositories (IR-style)

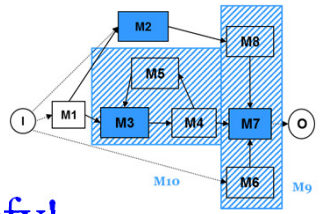
- Open question: Query languages for repositories
- Core of the problem: Workflow similarity [SCB+14]
- Same point with Reproducible papers (Notebooks)

➔ Efficiently reusing (searching for) Notebooks is an open question



Reducing the complexity of workflows

- Graph-based approaches
- Semantics-based approaches
- Software engineering/languages approaches



Finding the right set *of compatible libraries*

- Docker, VM allows to freeze the environment → Need to liquefy!

Bridge the gap between scripts and workflows

Conclusion

Many scientific results are not computationally reproducible

Providing scripts is an excellent start

Scientific workflows are increasingly mature solutions to

- Keep track of the exact connected tools used
- Keep track of the exact data used, produced and tool parameters setting
  - → Provenance modules
- Coarse-grain version of the analysis to better capture the analysis steps
- Exchange and share analysis pipelines (myExperiment)

Conclusion

Many scientific results are not computationally reproducible

Providing scripts is an excellent start

Scientific workflows are increasingly mature solutions to

- Keep track of the exact connected tools used
- Keep track of the exact data used, produced and tool parameters setting
  - → Provenance modules
- Coarse-grain version of the analysis to better capture the analysis steps
- Exchange and share analysis pipelines (myExperiment)

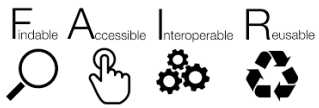
Repeat is (almost) always reachable

- Next levels may be more difficult to reach

Several open challenges are directly related to improvement in research in computer science (graphs, algorithmics…)

Workflows play key role to produce FAIR data

FAIR metrics for workflows have to be defined too!



Results of our Action

(1) Paper @ FGCS
Levels of reproducibility
Criteria of choice
Open Challenges

Future Generation Computer Systems
Volume 75, October 2017, Pages 284–298

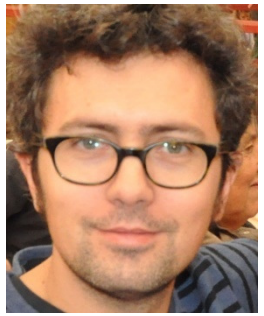
Scientific workflows for computational reproducibility in the life sciences: Status, challenges and opportunities

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]

https://hal.archives-ouvertes.fr/hal-01516082/document

(2) 3 hour Webinar : Tutorial + 2 demos

(3) ReproHackathon
New concept designed
3 editions
- RNA seq 06/2017 Gif, PhiloData 07/2018, Lyon
- Next edition Nov. 2019 Plant phenotyping, Montpellier



Join us!
cohen@lri.fr