Towards reproducibility

And how to design experiment in Computer Science

Motivation

“It’s not really for the benefit of other people. Experience shows the principal beneficiary of reproducible research is you the author yourself.”

Three degrees of reproducibility

- First degree: Author could rerun experiment on his own.

  Potential problems: source codes and scripts are lost or changed, experiment conditions are forgotten, steps in data cleaning are lost.. etc.

- Second degree: Any other can repeat experiment. All information must be publicly available.

- Third degree: Anybody can do a follow-up experiment based on your research.

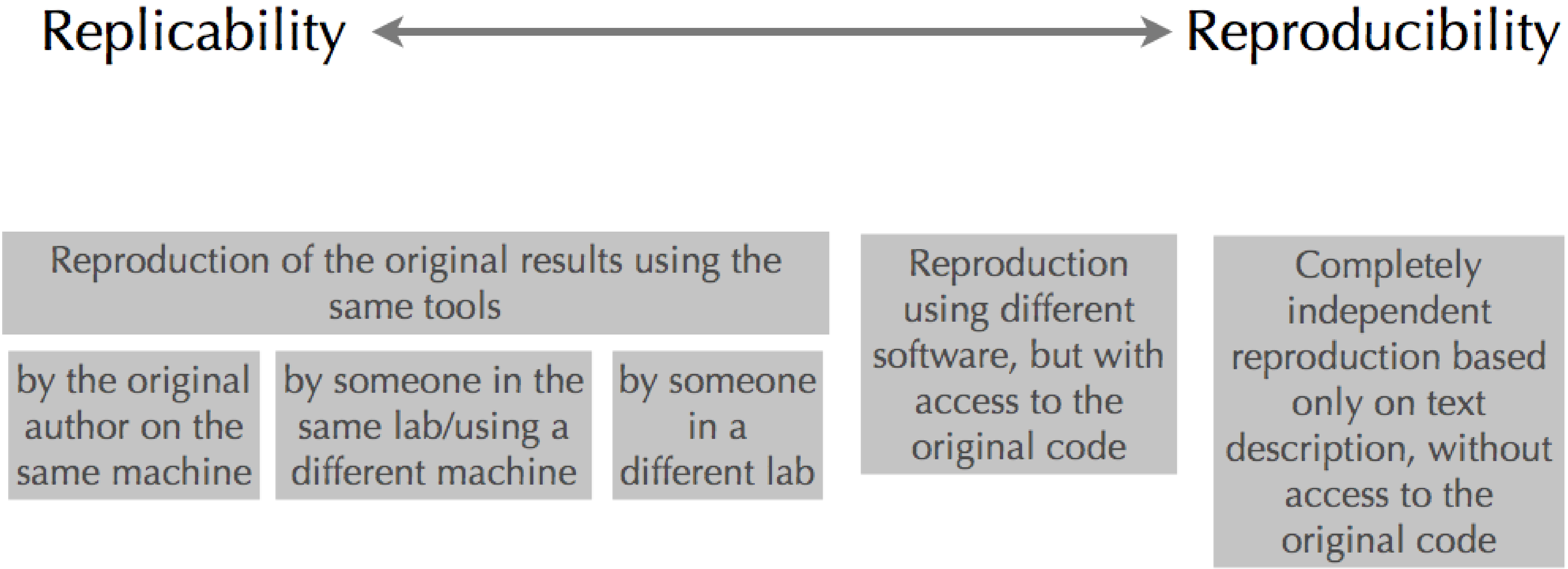
Common terms

- **Repeat (rerun)** – same researchers and experiment. E.g, when additional results from reviewers are required.
- **Replicate (recompute)** – different researchers, same experiment, same conditions. E.g., make sure that first research group did not make a mistake.
- **Reproduce** – same experiment, insignificantly different conditions.

OR making same experiment on data from first experiment.

- **Reuse** – some part of experiment (data, code) is used for different purposes.

Replicability vs. Reproducibility



Replicability vs. Reproducibility

- Replicability is stronger than reproducibility.
- A study is only replicable if you perform the exact same experiment (at least) twice, collect data in the same way both times, perform the same data analysis, and arrive at the same conclusions.
- Replication = independent people going out and collecting new data
- Reproducibility = independent people analyzing the same data.

Reproducibility by Thompson and Burnett [1]

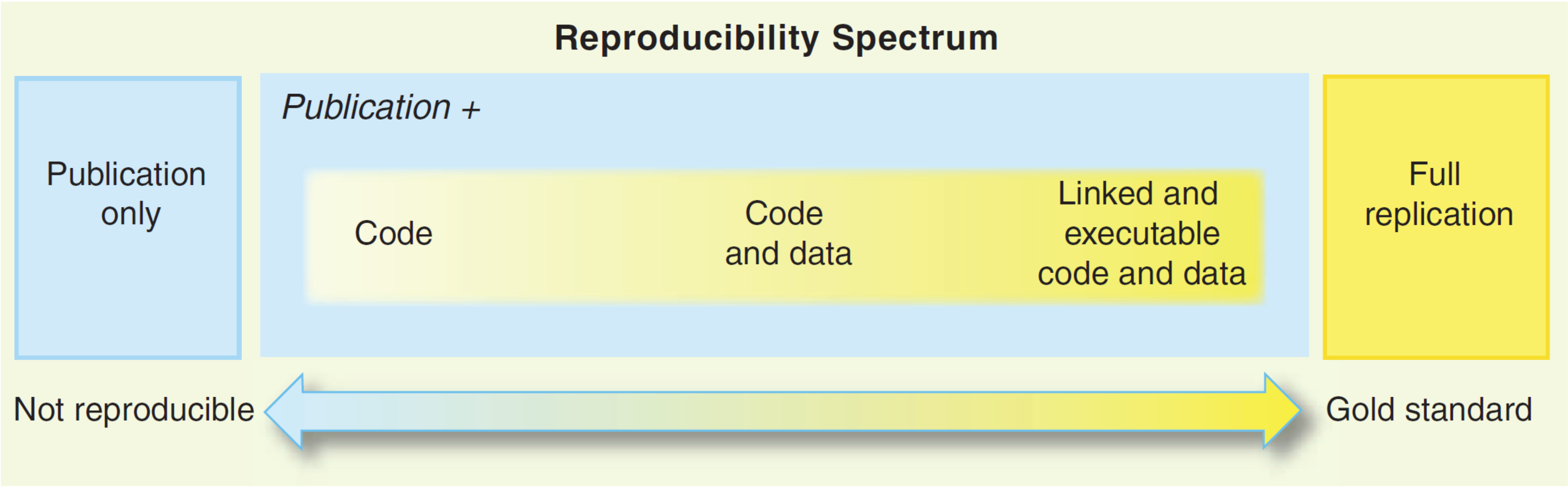
- **Code** – sharing source code, tools and workflow to execute this tools.

- **Methods** – scripts which conduct analyses and produce components of publication such as tables and figures.

- **Data** – access to raw data

[1] Paul A. Thompson and Andrew Burnett: Reproducible research, 2012

Reproducible research in computational science [1]



[1] R. Peng: Reproducible research in computational science, 2011

The recomputation manifesto [1]

1. Computational experiments should be recomputable for all time
2. Recomputation of recomputable experiments should be very easy
3. Tools and repositories can help recomputation become standard
4. It should be easier to make experiments recomputable than not to
5. The only way to ensure recomputability is to provide virtual machines
6. Runtime performance is a secondary issue

[1] Ian P. Gent: The recomputation manifesto, 2013

Ten simple rules for reproducible computational research [1]

- **For every result, keep track of how it was produced.** – Track program name, version, exact parameters and inputs.
- **Avoid manual data manipulation steps.** – Use scripts instead.
- **Archive the exact versions of all external programs used.**
- **Version control all custom scripts.** – Only exact state of script can produce the exact input.
- **Record all intermediate results, when possible in standardized formats**. – It allows parts of process to be rerun.
- **For analyses that include randomness, note underlying random seeds**

Ten simple rules for reproducible computational research

- **Always store raw data behind plots.** – And also store the code to make a plot.
- **Generate hierarchical analysis output, allowing layers of increasing detail to be inspected.** – Provide not only raw data but also data analysis summaries.
- **Connect textual statements to underlying results.** – Data and its interpretation should not be separated.
- **Provide public access to scripts, runs and results.**

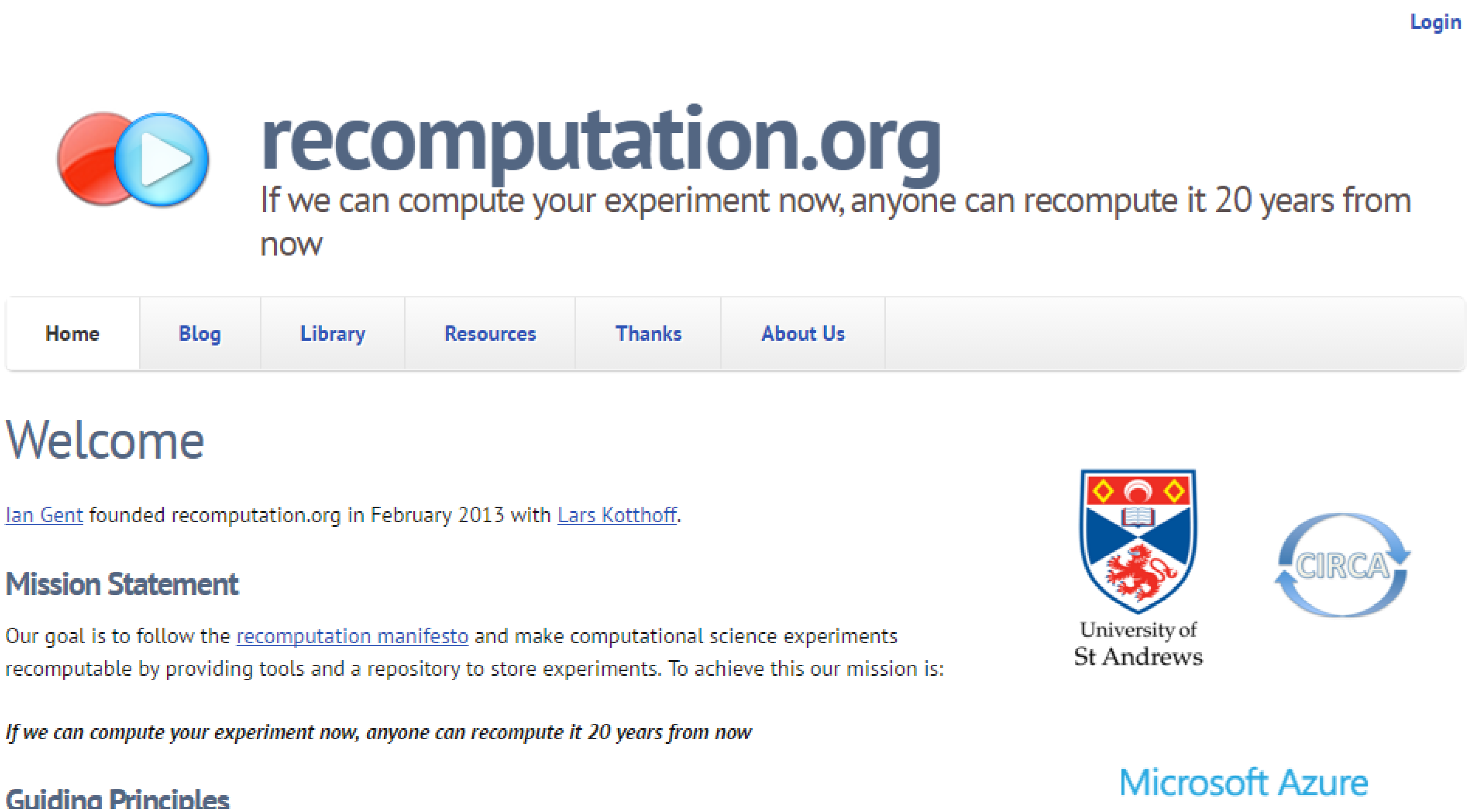
[1] Sandve: Ten simple rules for reproducible computational research, 2013

Tools

- Store not only data, but repository of fully realised experiment.
- GitHub, Subversion,
- Git + DOI: https://guides.github.com/activities/citable-code/
- http://www.runmycode.org
- http://www.myexperiment.org
- http://recomputation.org

Runmycode.org

Recomputation.org


Login

recomputation.org

If we can compute your experiment now, anyone can recompute it 20 years from now

Home | Blog | Library | Resources | Thanks | About Us

Welcome

Ian Gent founded recomputation.org in February 2013 with Lars Kotthoff.

Mission Statement

Our goal is to follow the recomputation manifesto and make computational science experiments recomputable by providing tools and a repository to store experiments. To achieve this our mission is:

If we can compute your experiment now, anyone can recompute it 20 years from now

Guiding Principles

University of St Andrews

CIRCA

Microsoft Azure

Any question about reproducibility?

How to design computer security experiments [1]

- Classical „science“ method:
- 1. Form hypothesis
- 2. Perform experiment and collect data.
- 3. Analyze data.
- 4. Interpret data and draw conclusions.
- 5. Depending on conclusions, return to #1 and iterate.

[1] S. Peisert and M. Bishop: How to design computer security experiments.

3. Basic needs

- 1. Falsifiable. An experiment must be constructed to test a hypothesis that is both testable and falsifiable. Other factors: observability and measurability.
- 2. Controlled. An experiment must have exactly one variable, or if an experiment has multiple variables, then it must be able to be separated into multiple experiments where exactly one variable at a time can be tested.
- 3. Reproducible. An experiment must be reproducible, and results repeatable.

Some „hypothesis“ examples

- Software X is secure.

Some „hypothesis“ examples

- Software X is secure.
- Intrusion detection system X catch 50% more attacks than with our competitor’s product, Y .

Some „hypothesis“ examples

- Software X is secure.
- Intrusion detection system X catch 50% more attacks than with our competitor’s product, Y .
- Only an extraordinarily skilled attacker can break into our firewall.

Some „hypothesis“ examples

- Software X is secure.
- Intrusion detection system X catch 50% more attacks than with our competitor’s product, Y .
- Only an extraordinarily skilled attacker can break into our firewall.
- The firewall accepts all well-formed packets and sessions, and handles malformed packets and sessions as documented in the firewall’s manual.

Experiment design to test the hypothesis.

- Hypothesis: The firewall accepts all well-formed packets and sessions, and handles malformed packets and sessions as documented in the firewall's manual.
- Experiment design: Connect the firewall to a local network and send packets, some malformed and some parts of malformed sessions, through the firewall. Record the packets and the firewall's responses.

What is missing in Experiment design?

Experiment design to test hypothesis.

- Hypothesis: The firewall accepts all well-formed packets and sessions, and handles malformed packets and sessions as documented in the firewall’s manual.
- Experiment 2 : Connect the firewall to a local network and send packets, some malformed and some parts of malformed sessions, though the firewall. Record the network traffic, including timings, and the firewall’s responses.

Think twice about input data!

- Input data set should accurately captures the relevant characteristics of the data that the firewall must handle in practice.

(Input data should represent data that firewall will handle in practise.)

Questions about experiment design?

Further (openLab) research?

- Ethical aspect in research (especially in cybersecurity).
- Doing measurements properly (Hopet talk on DUVOD). You may see: C. McGeoch: **A Guide to Experimental Algorithmics**

Thank you for your attention!