Reproducible Research for Computer Scientists

Arnaud Legrand and Jean-Marc Vincent

Scientific Methodology and Performance Evaluation
M2R MOSIG, Grenoble, September-December 2015

Outline

Frustration as an Author

- I thought I used the same parameters but I'm getting different results!
- The new student wants to compare with the method I proposed last year
- My advisor asked me whether I took care of setting this or this but I can't remember
- The damned fourth reviewer asked for a major revision and wants me to change figure 3 :(
- Which code and which data set did I use to generate this figure?
- It worked yesterday!
- 6 months later: why did I do that?

Frustration as a Reviewer

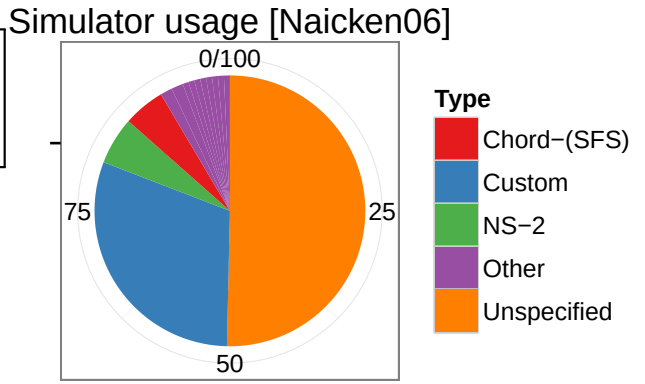
This may be an interesting contribution but:

- This average value must hide something
- As usual, there is no confidence interval, I wonder about the variability and whether the difference is significant or not
- That can't be true, I'm sure they removed some points
- Why is this graph in logscale? How would it look like otherwise?
- The authors decided to show only a subset of the data. I wonder what the rest looks like
- There is no label/legend/... What is the meaning of this graph? If only I could access the generation script

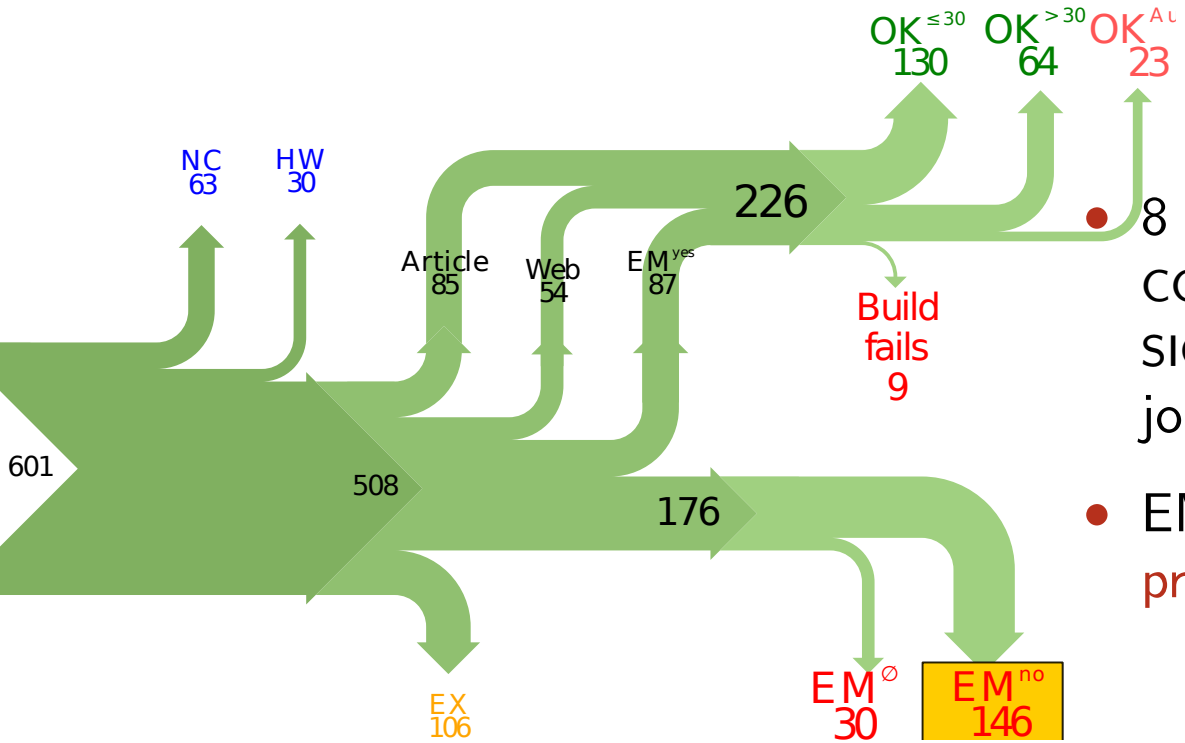
A Few Edifying Examples

Naicken, Stephen *et Al.*, *Towards Yet Another Peer-to-Peer Simulator*, HET-NETs'06.

From 141 P2P sim.papers, 30% use a custom tool,
50% don't report used tool



Collberg, Christian *et Al.*, *Measuring Reproducibility in Computer Systems Research*, http://reproducibility.cs.arizona.edu/



- 8 ACM conferences (ASPLOS'12, CCS'12, OOPSLA'12, OSDI'12, PLDI'12, SIGMOD'12, SOSP'11, VLDB'12) and 5 journals
- EM^{no}= the code cannot be provided

The Dog Ate my Homework !!!

- Versioning Problems

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.

Attached is the ⟨system⟩ source code of our algorithm. I'm not very sure whether it is the final version of the code used in our paper, but it should be at least 99% close. Hope it will help.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices

Unfortunately, the server in which my implementation was stored had a disk crash in April and three disks crashed simultaneously. While the help desk made significant effort to save the data, my entire implementation for this paper was not found.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon

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.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon
- No Intention to Release

I am afraid that the source code was never released. The code was never intended to be released so is not in any shape for general use.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon
- No Intention to Release
- Programmer Left

⟨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.

Unfortunately, the author who has done most of the coding for this paper has passed away and the code is no longer maintained.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon
- No Intention to Release
- Programmer Left
- Commercial Code

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.

The code owned by ⟨COMPANY⟩, and AFAIK the code is not open-source. Your best bet is to reimplement :(Sorry.

The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon
- No Intention to Release
- Programmer Left
- Commercial Code
- Proprietary Academic Code

Unfortunately, the ⟨SYSTEM⟩ sources are not meant to be opensource (the code is partially property of ⟨UNIVERSITY 1⟩, ⟨UNIVERSITY 2⟩ and ⟨UNIVERSITY 3⟩.)

If this will change I will let you know, albeit I do not think there is an intention to make the ⟨SYSTEM⟩ sources opensource in the near future.

If you're interested in obtaining the code, we only ask for a description of the research project that the code will be used in (which may lead to some joint research), and we also have a software license agreement that the University would need to sign.

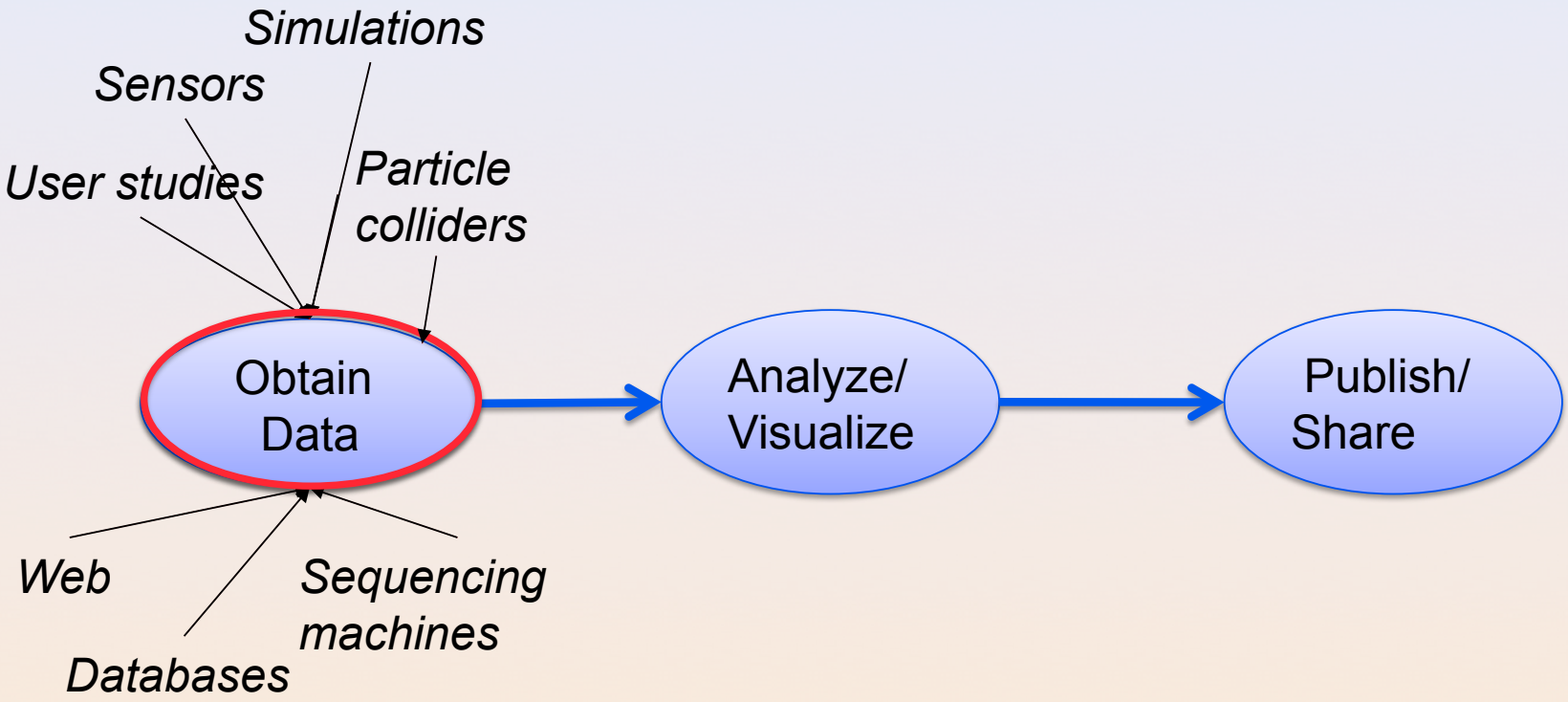
The Dog Ate my Homework !!!

- Versioning Problems
- Bad Backup Practices
- Code Will be Available Soon
- No Intention to Release
- Programmer Left
- Commercial Code
- Proprietary Academic Code
- Research vs. Sharing
- ...
- ...

In the past when we attempted to share it, we found ourselves spending more time getting outsiders up to speed than on our own research. So I finally had to establish the policy that we will not provide the source code outside the group.

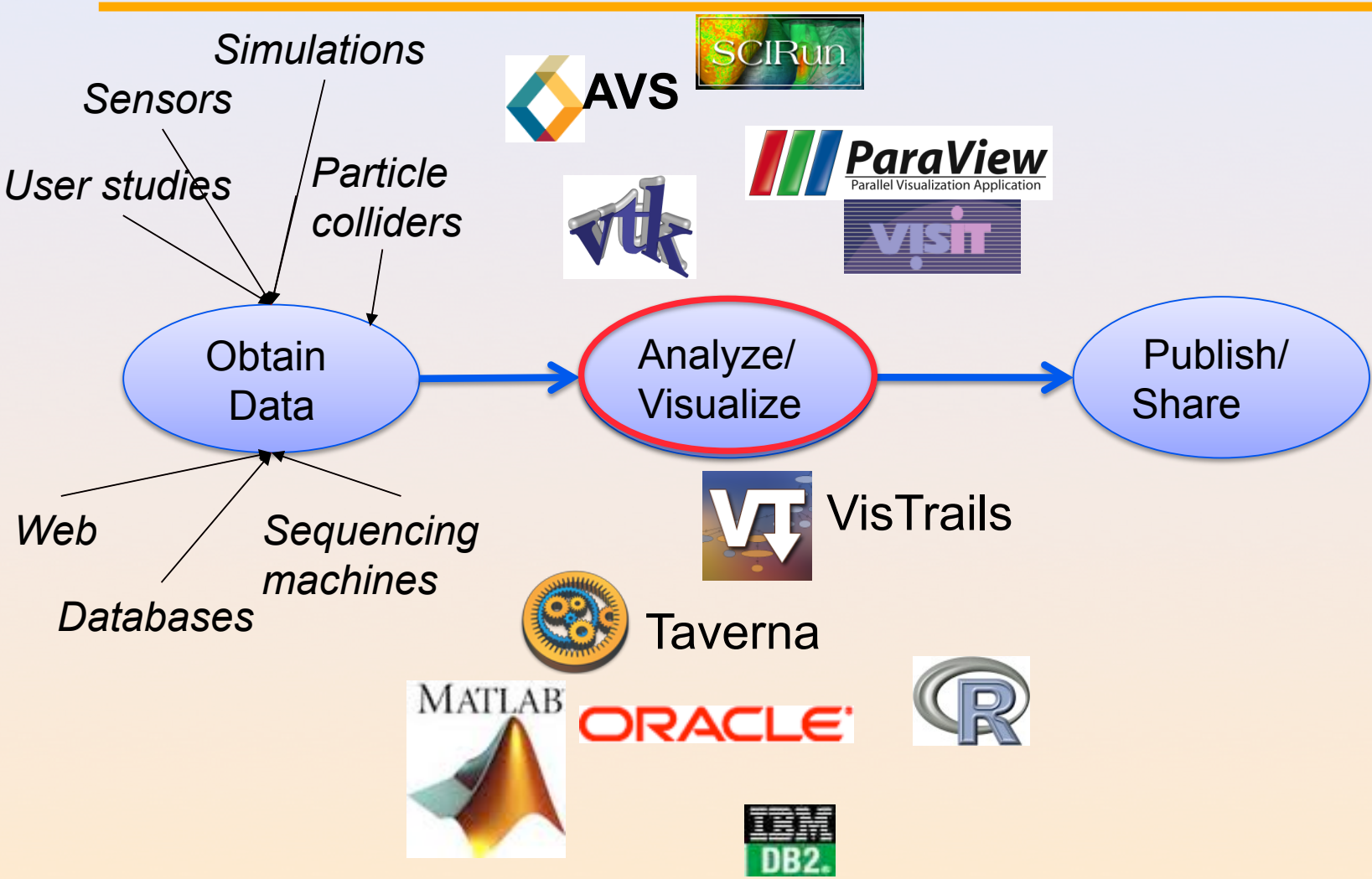
Outline

Science Today: Data Intensive



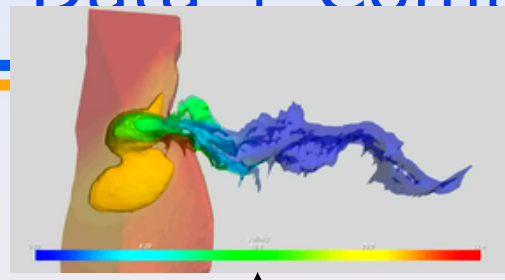
Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

Science Today: Data + Computing Intensive



Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

Science Today: Data + Computing Intensive



Simulations

Sensors

User studies

Particle colliders

Microsoft PowerPoint

Obtain Data

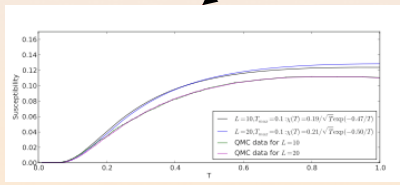
Analyze/ Visualize

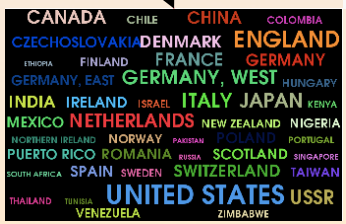
Publish/ Share

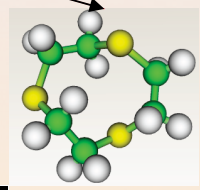
Web

Sequencing machines

Databases

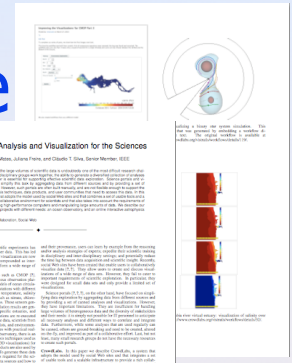


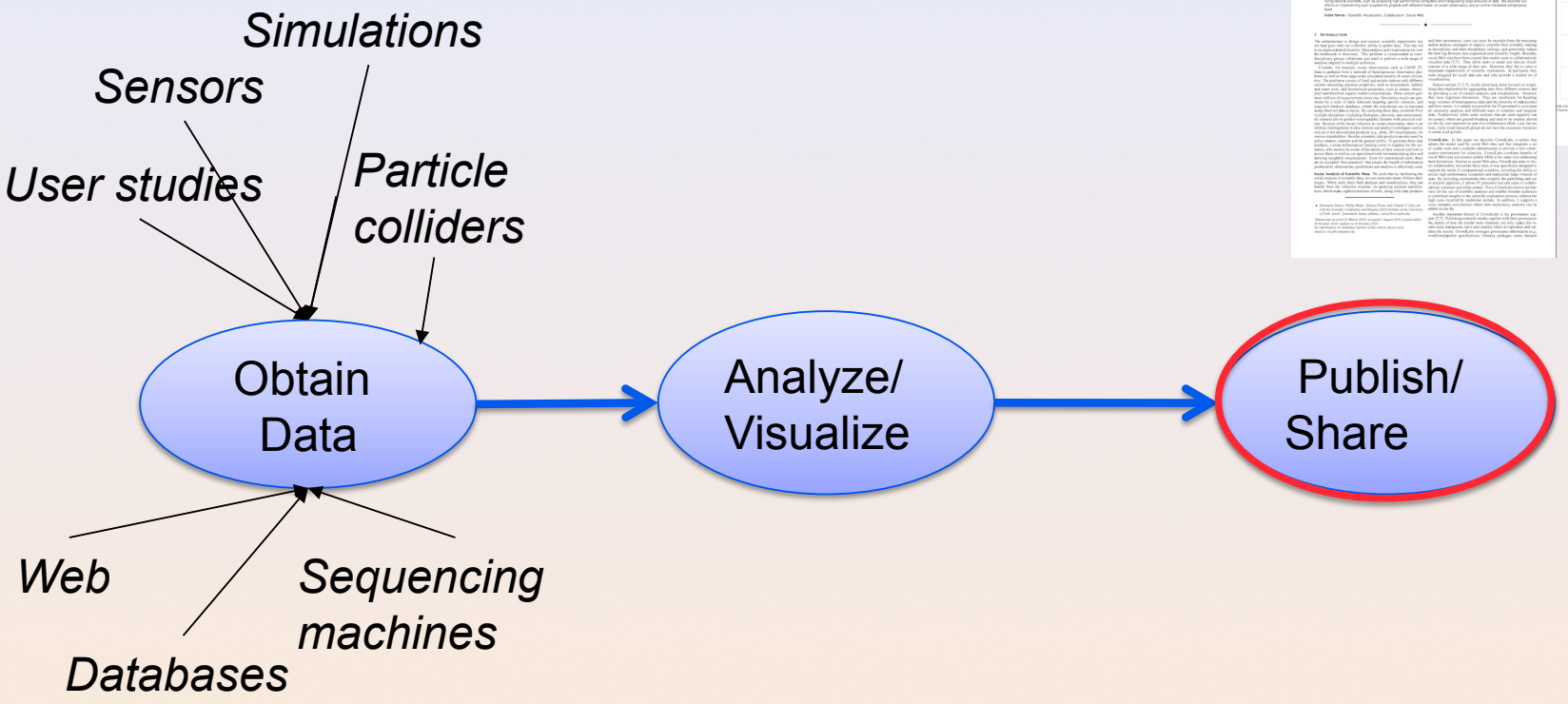




Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

Science Today: Data + Computing Inte





Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

Science Today: Incomplete Publications

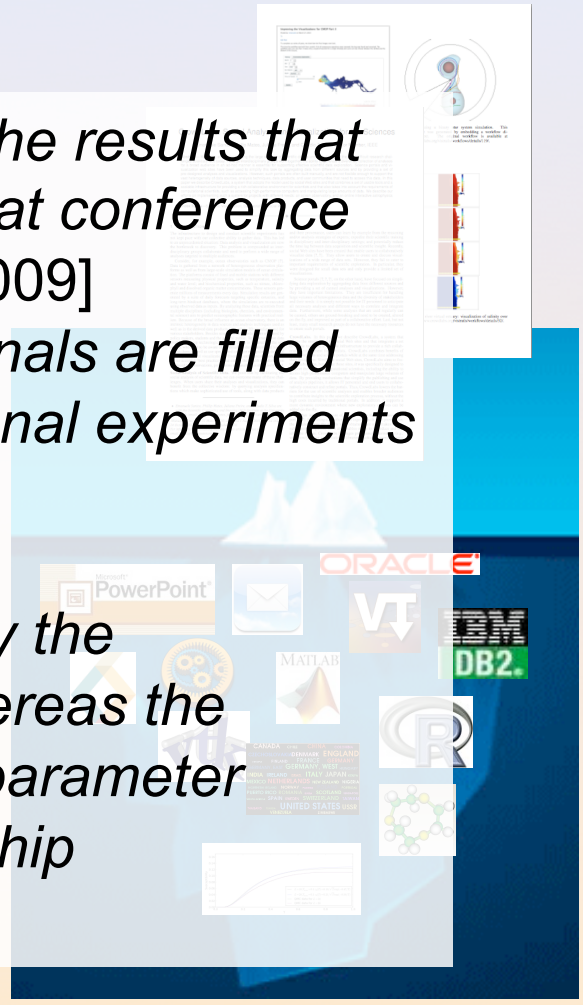
- Publications are just the tip of the iceberg
  - Scientific record is incomplete--- to large to fit in a paper
  - Large volumes of data
  - Complex processes
- Can't (easily) reproduce results

Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

Science Today: Incomplete Publications

- Publications are just the tip of the iceberg
  - Scientific record is incomplete---
  - Large volumes of data
  - Complex processes
- Can't (easily) reproduce results

> "*It's impossible to verify most of the results that computational scientists present at conference and in papers.*" [Donoho et al., 2009]
>
> "*Scientific and mathematical journals are filled with pretty pictures of computational experiments that the reader has no hope of repeating.*" [LeVeque, 2009]
>
> "*Published documents are merely the advertisement of scholarship whereas the computer programs, input data, parameter values, etc. embody the scholarship itself.*" [Schwab et al., 2007]



Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

A few Words on Scientific Foundation

- Falsifiability or refutability of a statement, hypothesis, or theory is an inherent possibility to prove it to be false (not "*commit fraud*" but "*prove to be false*").
- Karl Popper makes falsifiability the demarcation criterion to distinguish the scientific from the unscientific

> *It is not only not right, it is not even wrong!*
> *– Wolfgang Pauli*

- Theories cannot be proved correct but they can be disproved. Only a few stand the test of batteries of critical experiments.
- It is not all black and white. There are many stories where scientists stick with their theories despite evidences and sometimes, they were even right to do so...

Testing and checking is thus one of the basis of science

Further readings: A Summary of Scientific Method, Peter Kosso, Springer

Why Are Scientific Studies so Difficult to Reproduce?

- Copyright/competition issue
- Publication bias (only the idea matters, not the gory details)
- Rewards for positive results
- Experimenter bias
- Programming errors or data manipulation mistakes
- Poorly selected statistical tests
- Multiple testing, multiple looks at the data, multiple statistical analyses
- ~~Lack of easy-to-use tools~~

Courtesy of Adam J. Richards

Evidence for a Lack of Reproducibility

- Studies showing that scientific papers commonly leave out experimental details essential for reproduction and showing difficulties with replicating published experimental results:
  - J.P. Ioannidis. *Why Most Published Research Findings Are False* PLoS Med. 2005 August; 2(8)
- High number of failing clinical trials.
  - *Do We Really Know What Makes Us Healthy?*, New-York Times — September 16, 2007
  - *Lies, Damned Lies, and Medical Science*, The Atlantic. 2010, Nov.
- Increase in retracted papers:
  - Steen RG, *Retractions in the scientific literature: is the incidence of research fraud increasing?* J Med Ethics 37: 249–253.

Courtesy of Adam J. Richards

A Reproducibility Crisis?

The Duke University scandal with scientific misconduct on lung cancer

- *Nature Medicine* - 12, 1294 - 1300 (2006) Genomic signatures to guide the use of chemotherapeutics, by Anil Potti and 16 other researchers from Duke University and University of South Florida
- Major commercial labs licensed it and were about to start using it before two statisticians discovered and publicized its faults

Dr. Baggerly and Dr. Coombes found errors almost immediately. Some seemed careless — moving a row or a column over by one in a giant spreadsheet — while others seemed inexplicable. The Duke team shrugged them off as "clerical errors."

The Duke researchers continued to publish papers on their genomic signatures in prestigious journals. Meanwhile, they started three trials using the work to decide which drugs to give patients.

- Retractions: January 2011. Ten papers that Potti coauthored in prestigious journals were retracted for varying reasons
- Some people die and may be getting worthless information that is based on bad science

Courtesy of Adam J. Richards

Definitely

A recent scandal In 2013, *Dong-Pyou Han*, a former assistant professor of biomedical sciences at Iowa State University was disgraced:

- Falsified blood results to make it appear as though a vaccine he was working on had exhibited anti-HIV activity
- Han and his team received $\approx$ \$19 million from NIH
- Retraction and resignation of university
- Han was sentenced in 2015 to 57 months imprisonment for fabricating and falsifying data in HIV vaccine trials. He was also fined US \$7.2 million!

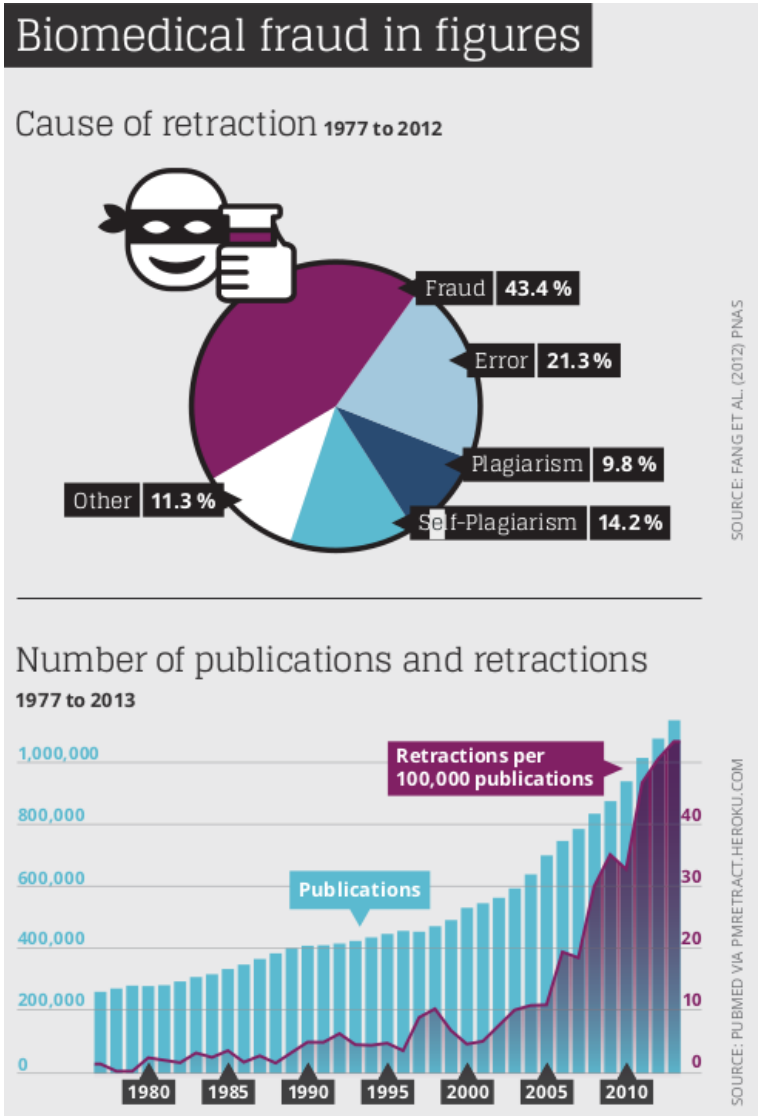
We should avoid witch-hunt

- August 5, 2014, Yoshiki Sasai (stem cell, considered for Nobel Prize) hanged in his laboratory at the RIKEN (Japan). Fraud suspicion...
- In 1986, a young postdoctoral fellow at MIT accused her director, Thereza Imanishi-Kari, of falsifying the results of a study published in Cell and co-signed by the Nobel laureate David Baltimore. [..] Declared guilty, Univ. presidency resignation, and finally cleared. This put the careers of two outstanding researchers on hold for ten years based on unfounded accusations.

Scientific fraud is bad but let's be careful Have a look at the wikipedia *list of academic scandals*. On a totally different aspect, do not forget to also have a look at the *plagiarism* and *paper generation* entries at *having fun with h-index*

The Battle against Scientific Fraud in the CNRS International Magazine

Is Fraud a new phenomenon?



- Galileo (data fabrication), Ptolemy (plagiarism), Mendel (data enhancement), Pasteur (rigorous but hided failures), ...

Outline

My Feeling

Computer scientists have an incredibly poor training in probabilities, statistics, experiment management

Why should we? Computer are deterministic machines after all, right? ;)

Ten years ago, I've started realizing how lame the articles I reviewed (as well as those I wrote) were in term of experimental methodology.

- Yeah, I know, your method/algorithm is better than the others as demonstrated by the figures
- Not enough information to discriminate real effects from noise
- Little information about the workload
- Would the "conclusion" still hold with a slightly different workload?
- I'm tired of awful combination of tools (perl, gnuplot, sql, ...) and bad methodology

Common practice in CS

Computer scientists tend to either:

- vary one factor at a time, use a very fine sampling of the parameter range,
- run millions of experiments for a week varying a lot of parameters and then try to get something of it. Most of the time, they (1) don't know how to analyze the results (2) realize something went wrong...

Interestingly, most other scientists do the exact opposite.

These two flaws come from poor training and from the fact that C.S. experiments are almost free and very fast to conduct

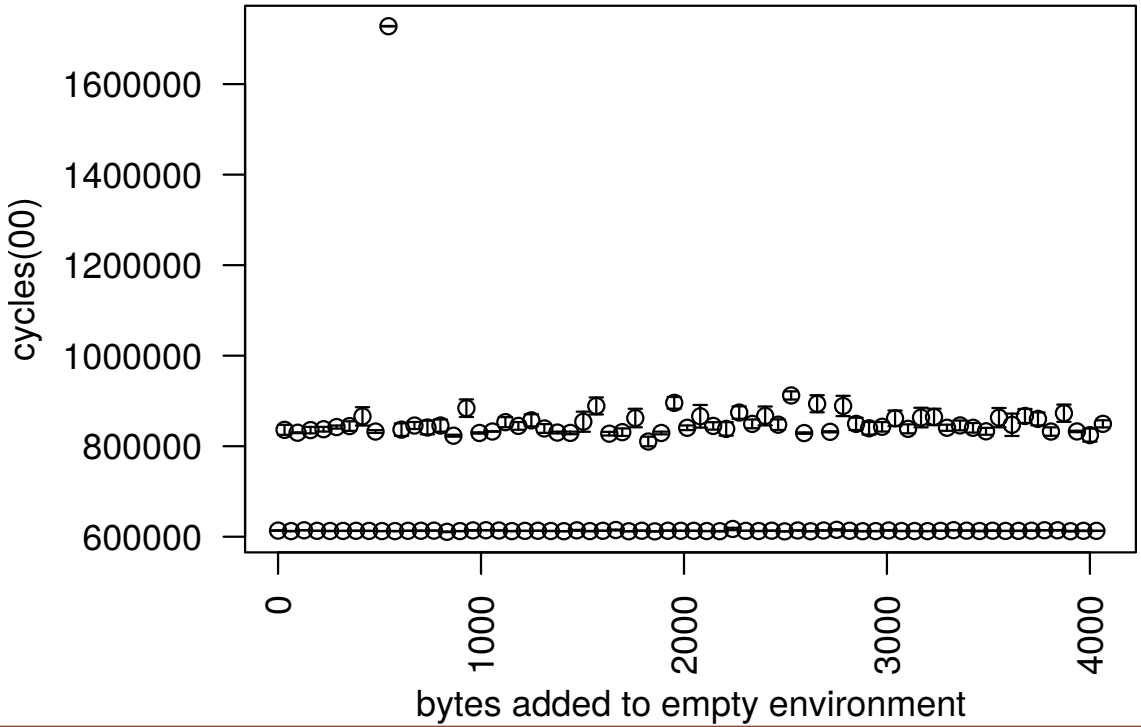
- Most strategies of experimentation (DoE) have been designed to provide sound answers despite all the randomness and uncontrollable factors
- Maximize the amount of information provided by a given set of experiments
- Reduce as much as possible the number of experiments to perform to answer a given question under a given level of confidence

Takes a few lectures on Design of Experiments to improve. But anyone can start by reading Jain's book on The Art of Computer Systems Performance Analysis

But do we **really** have to care?

Yes, although designed and built by human beings, computers are so complex that mistakes are easy to do...

- T. Mytkowicz, A. Diwan, M. Hauswirth, and P. F. Sweeney. Producing wrong data without doing anything obviously wrong!. SIGPLAN Not. 44(3), March 2009



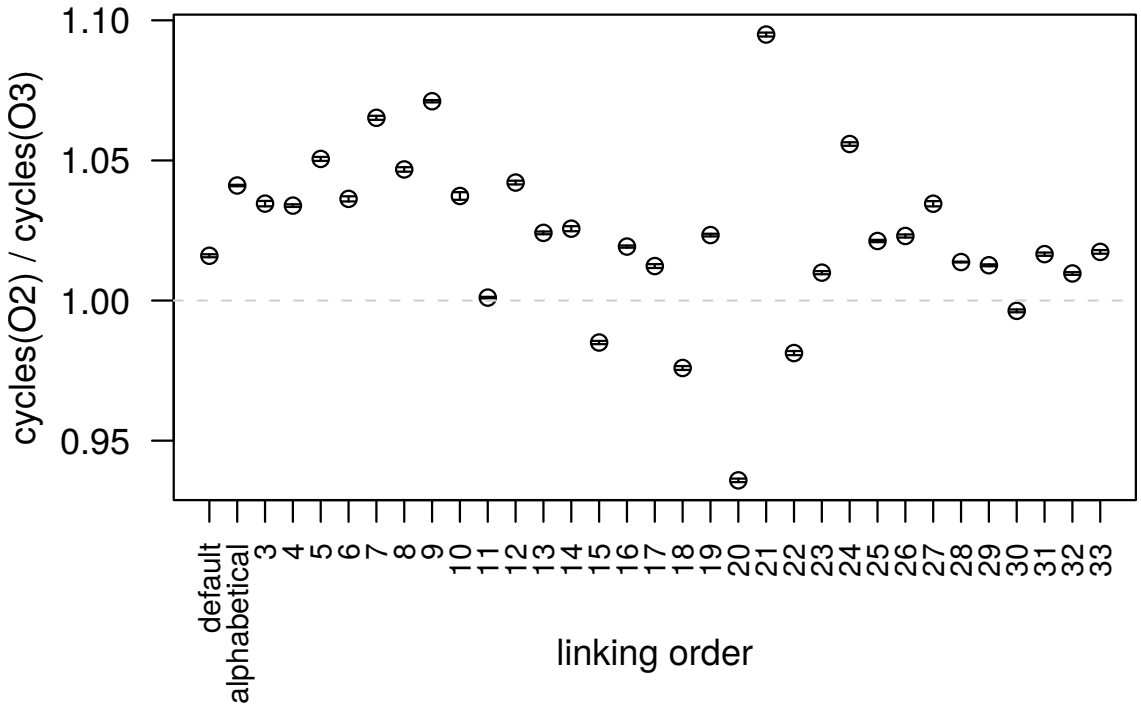
Key principles of experiment design

- Randomize to reduce bias
- Replicate (possibly in a smart way) to increase reliability

But do we **really** have to care?

Yes, although designed and built by human beings, computers are so complex that mistakes are easy to do...

- T. Mytkowicz, A. Diwan, M. Hauswirth, and P. F. Sweeney. Producing wrong data without doing anything obviously wrong!. SIGPLAN Not. 44(3), March 2009



Key principles of experiment design

- Randomize to reduce bias
- Replicate (possibly in a smart way) to increase reliability

Outline

Reproducible Research: the New Buzzword?

H2020-EINFRA-2014-2015

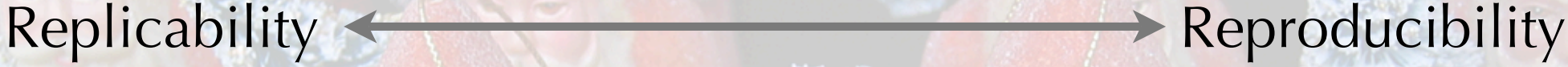
A key element will be capacity building to link literature and data in order to enable a more transparent evaluation of research and reproducibility of results.

More and more workshops

- Workshop on Duplicating, Deconstructing and Debunking (WDDD) (2014 edition)
- Reproducible Research: Tools and Strategies for Scientific Computing (2011)
- Working towards Sustainable Software for Science: Practice and Experiences (2013)
- REPPAR'14: 1st International Workshop on Reproducibility in Parallel Computing
- Reproducibility@XSEDE: An XSEDE14 Workshop
- Reproduce/HPCA 2014
- TRUST 2014

Should be seen as opportunities to share experience.

Reproducibility: What Are We Talking About?



Reproduction of the original results using the same tools

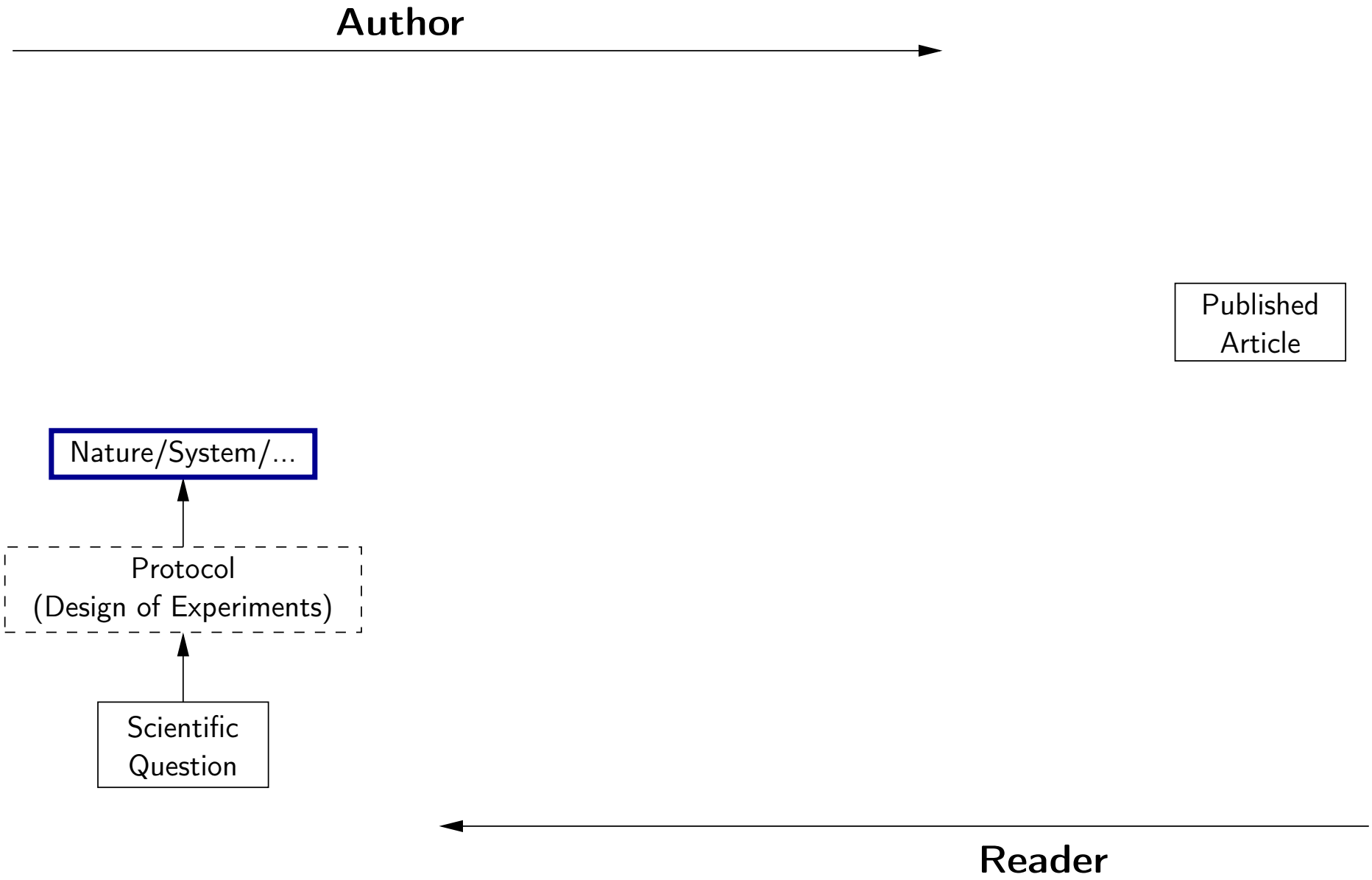
- by the original author on the same machine
- by someone in the same lab/using a different machine
- by someone in a different lab

Reproduction using different software, but with access to the original code

Completely independent reproduction based only on text description, without access to the original code

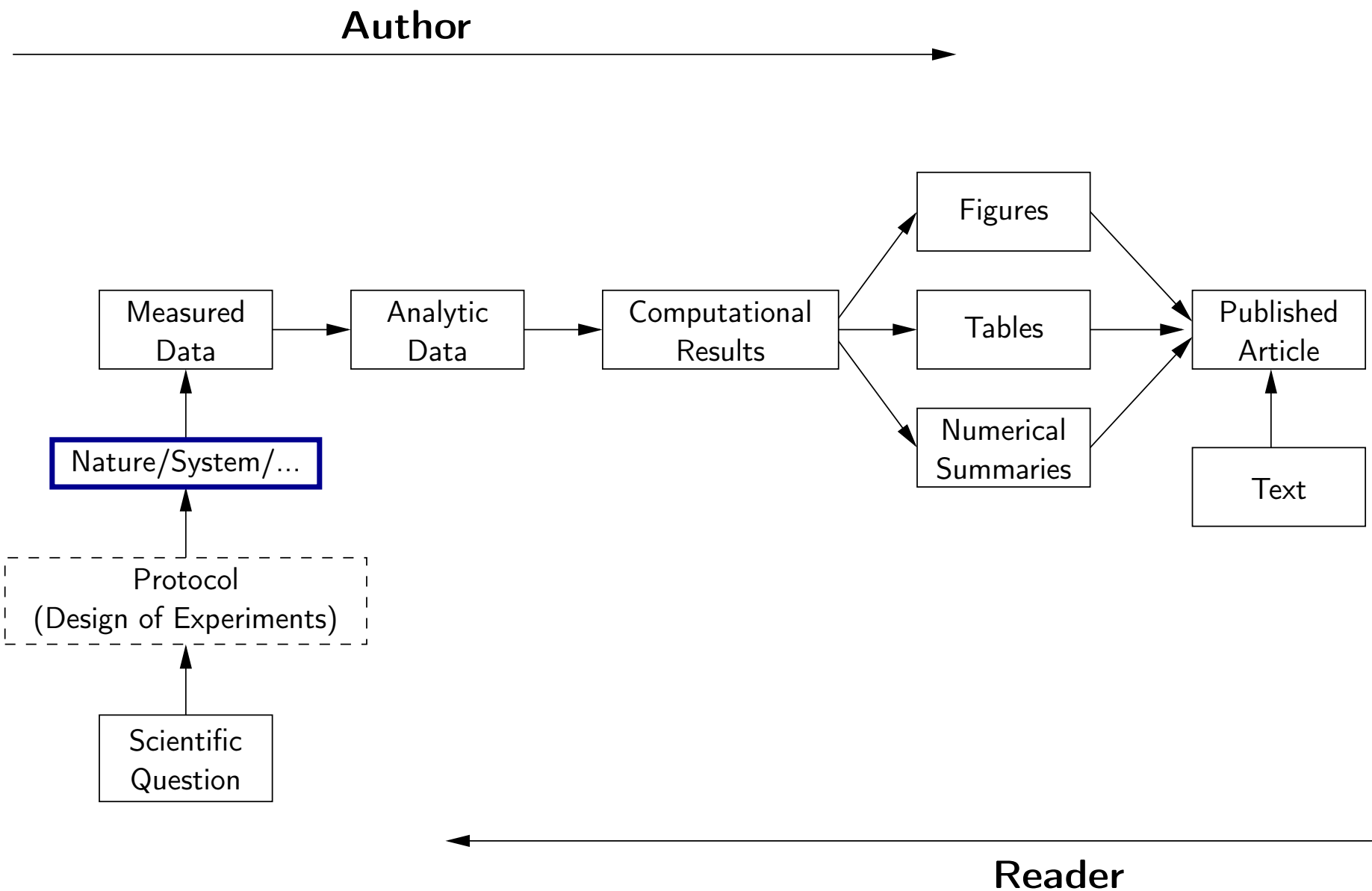
Courtesy of Andrew Davison (AMP Workshop on Reproducible research)

Reproducible Research: Trying to Bridge the Gap



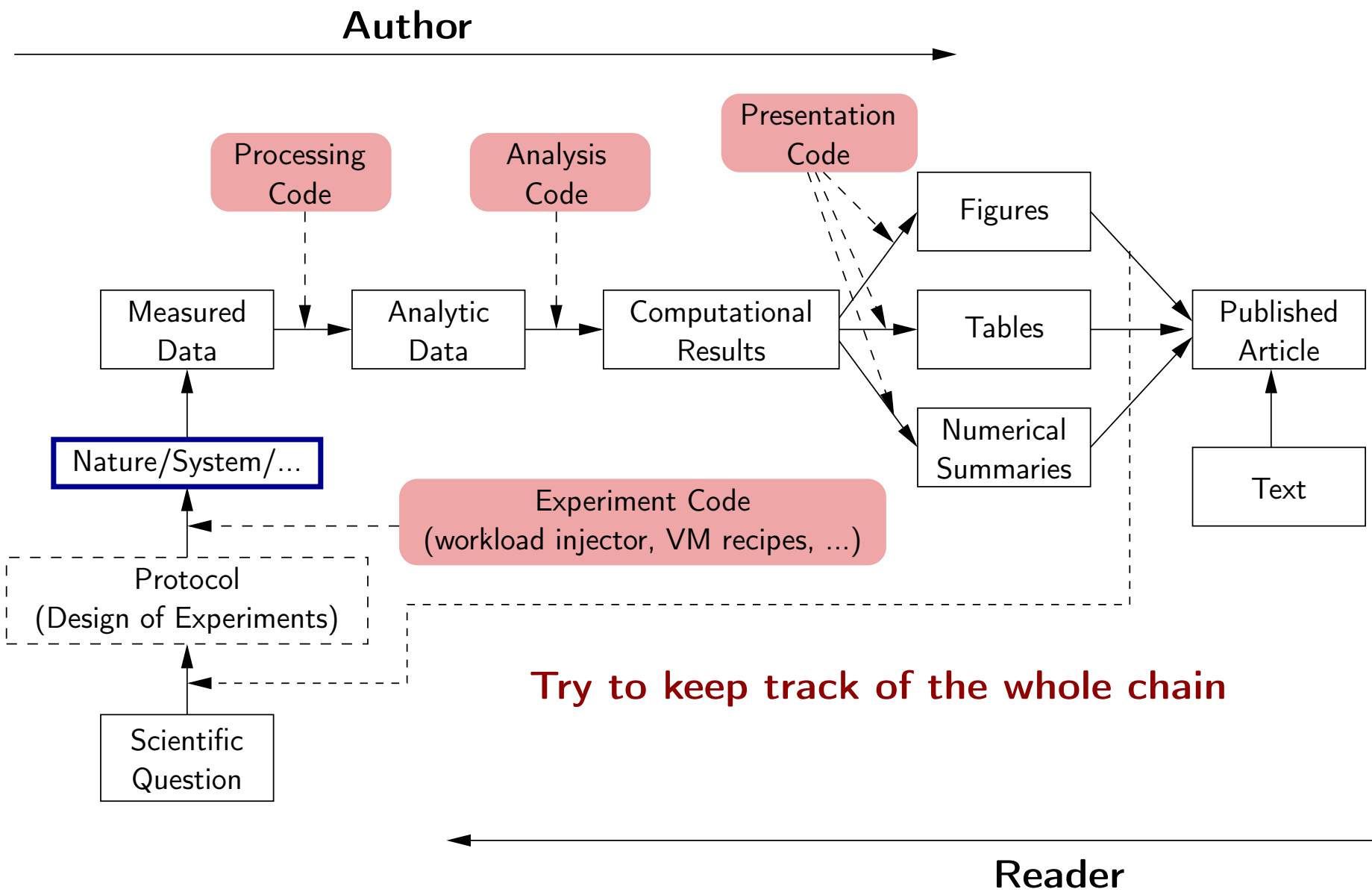
Inspired by Roger D. Peng's lecture on reproducible research, May 2014

Reproducible Research: Trying to Bridge the Gap



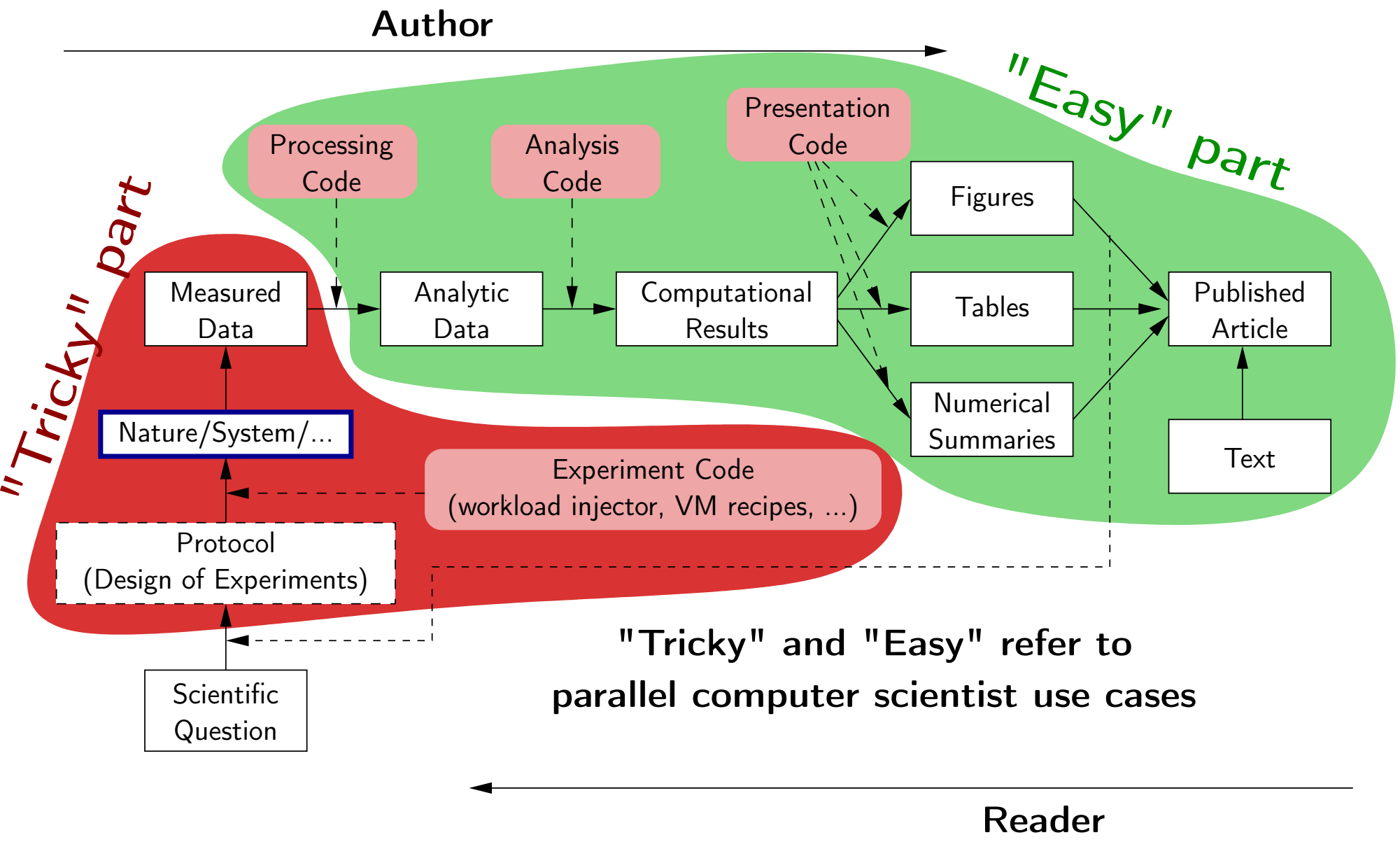
Inspired by Roger D. Peng's lecture on reproducible research, May 2014

Reproducible Research: Trying to Bridge the Gap



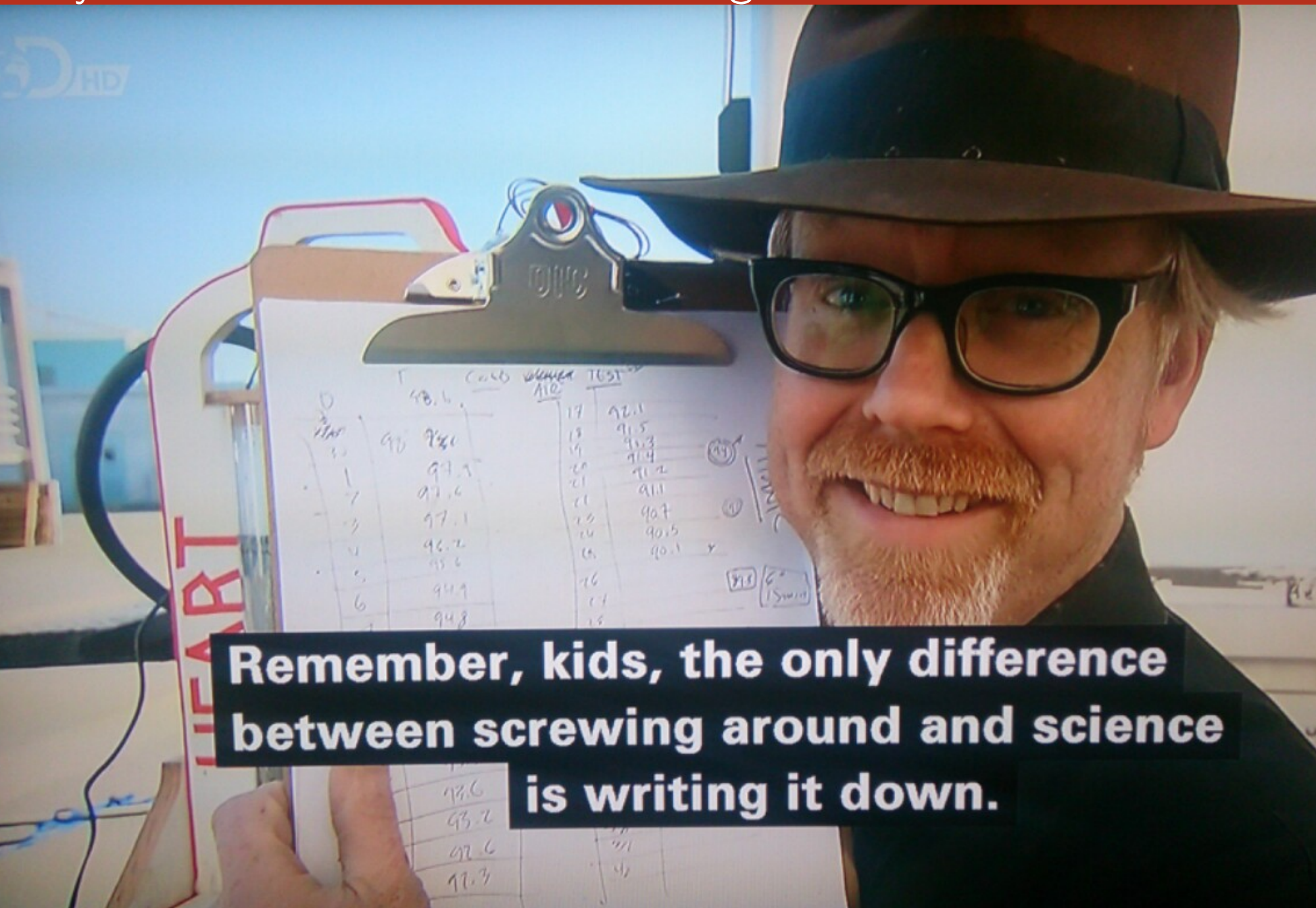
Inspired by Roger D. Peng's lecture on reproducible research, May 2014

Reproducible Research: Trying to Bridge the Gap



Inspired by Roger D. Peng's lecture on reproducible research, May 2014

Mythbusters: Science vs. Screwing Around



A Difficult Trade-off

Many different tools/approaches developed in various communities

But mainly two approaches:

Automatically keeping track of everything

- the code that was run (source code, libraries, compilation procedure)
- processor architecture, OS, machine, date, ...

VM-based solutions and experiment engines

Ensuring others can understand/adapt what was done

- Why did I run this?
- Does it still work when I change this piece of code for this one?

Laboratory notebook and recipes

Outline

A few Experiment Management Tools

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

} although nothing specific to G5K

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

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

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

Expo

- Grenoble (B. Videau, C. Ruis, O. Richard) http://expo.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment (based on taktuk, i.e., sh + ssh)
- At the moment Expo interacts with Planetlab and Grid5000 testbeds
- Resource and task abstractions, client-server organization, interactive or batch mode
- Native logging and archiving capabilities
  - every action performed on tasks, error flows, dates, ...
  - lets you know what was run, when, where and how

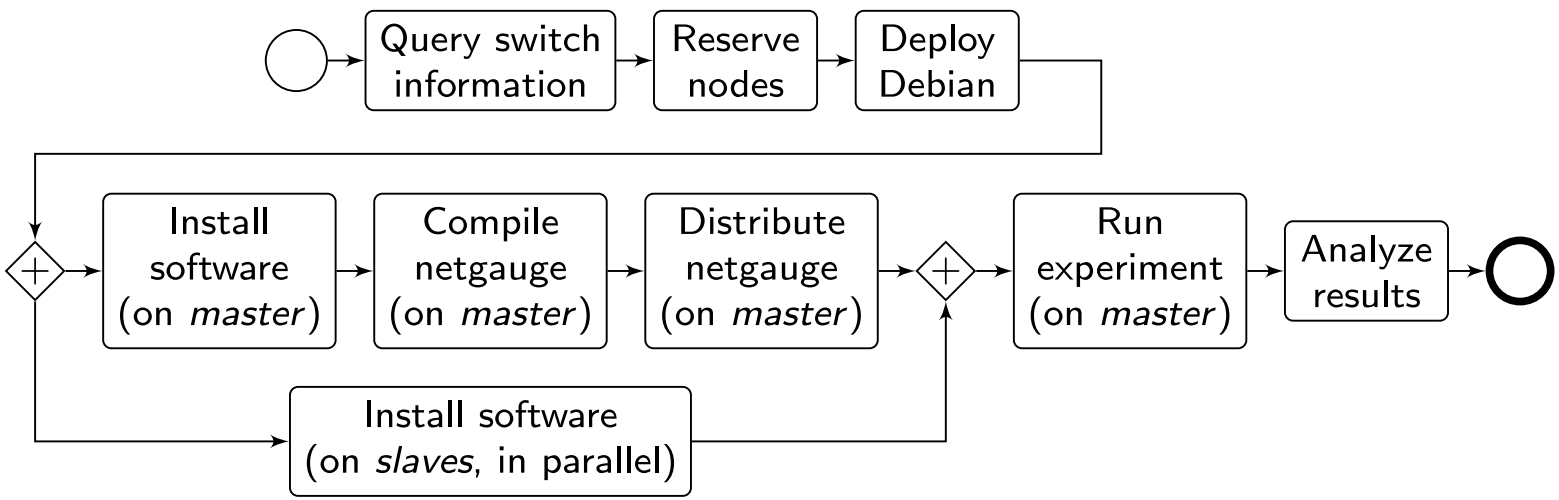
```
reserv=ExpoEngine::new(@connection)
reserv.site=["bordeaux","lille","luxembourg","nancy","sophia"]
reserv.resources=["nodes=50","nodes=10","nodes=4","nodes=4","nodes=30"]
reserv.name = "Expo Scalability"
reserv.walltime=600

reserv.run!
ptask $all, "hostname"
reserv.stop!
```

- Inspired similar tools like *Execo* that provides a Python-based API. Script-oriented, fork+sh+ssh or taktuk

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy



Courtesy of T. Buchert

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy

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

Courtesy of T. Buchert

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy

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

Activity `:install_pkgs`

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

Courtesy of T. Buchert

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy

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

Activity `:build_netgauge`

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

Courtesy of T. Buchert

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy

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

Activity `:dist_netgauge`

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

Courtesy of T. Buchert

XPflow

- Nancy (T. Buchert, L. Nussbaum) http://xpflow.gforge.inria.fr/
- DSL (Domain Specific Language) derived from Ruby and adapted to the management of experiment
- Resources, process, and activities
- Top-down rather than bottom-up: business process management
- Cope with failures through snapshots and retry policy

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

Activity :netgauge

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

Courtesy of T. Buchert

A few Environment Management Tools

CDE automatically tracks and packages up the Code, Data, and Environment

Providing not only VMs or binaries but also recipes is good!

E.g., the Kameleon project

- Univ. Grenoble (C. Ruiz, S. Harrache, M. Mercier, O. Richard, . . .)

  http://kameleon.readthedocs.org/
- Generate customized appliances (kvm, LXC, Virtualbox, iso, . . .)
- Ruby-based, YAML description of recipes with steps and aliases, execution in contexts
- Automatically checkpoints to rebuild only what is required

Outline

Our Approach: An Infrastructure to Support Provenance-Rich Papers [Koop et al., ICCS 2011]

- Tools for *authors* to create reproducible papers
  - Specifications that encode the computational processes
  - Package the results
  - Link from publications

Support different approaches

- Tools for testers to repeat and validate results
  - Explore different parameters, data sets, algorithms
- Interfaces for searching, comparing and analyzing experiments and results
  - Can we discover better approaches to a given problem?
  - Or discover relationships among workflows and the problems?
  - How to describe experiments?

Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

An *Provenance-Rich* Paper: ALPS2.0

The ALPS project release 2.0: Open source software for strongly correlated systems

B. Bauer[1] L. D. Carr[2] H.G. Evertz[3] A. Feiguin[4] J. Freire[5] S. Fuchs[6] L. Gamper[1] J. Gukelberger[1] E. Gull[7] S. Guertler[8] A. Hehn[1] R. Igarashi[9,10] S.V. Isakov[1] D. Koop[5] P.N. Ma[1] P. Mates[1,5] H. Matsuo[11] O. Parcollet[12] G. Pawłowski[13] J.D. Picon[14] L. Pollet[1,15] E. Santos[5] V.W. Scarola[16] U. Schollwöck[17] C. Silva[5] B. Surer[1] S. Todo[10,11] S. Trebst[18] M. Troyer[1]‡ M. L. Wall[2] P. Werner[1] S. Wessel[19,20]

[1]Theoretische Physik, ETH Zurich, 8093 Zurich, Switzerland
[2]Department of Physics, Colorado School of Mines, Golden, CO 80401, USA
[3]Institut für Theoretische Physik, Technische Universität Graz, A-8010 Graz, Austria
[4]Department of Physics and Astronomy, University of Wyoming, Laramie, Wyoming 82071, USA
[5]Scientific Computing and Imaging Institute, University of Utah, Salt Lake City, Utah 84112, USA
[6]Institut für Theoretische Physik, Georg-August-Universität Göttingen, Göttingen, Germany
[7]Columbia University, New York, NY 10027, USA
[8]Bethe Center for Theoretical Physics, Universität Bonn, Nussallee 12, 53115 Bonn, Germany

‡ Correspond

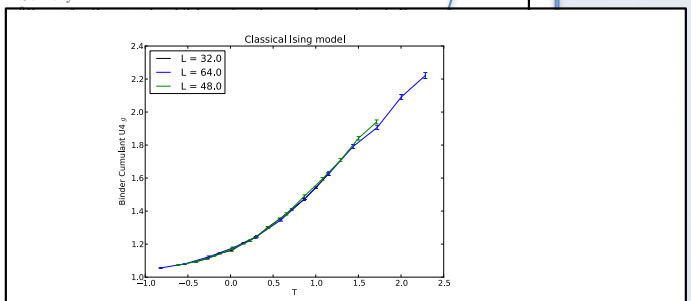
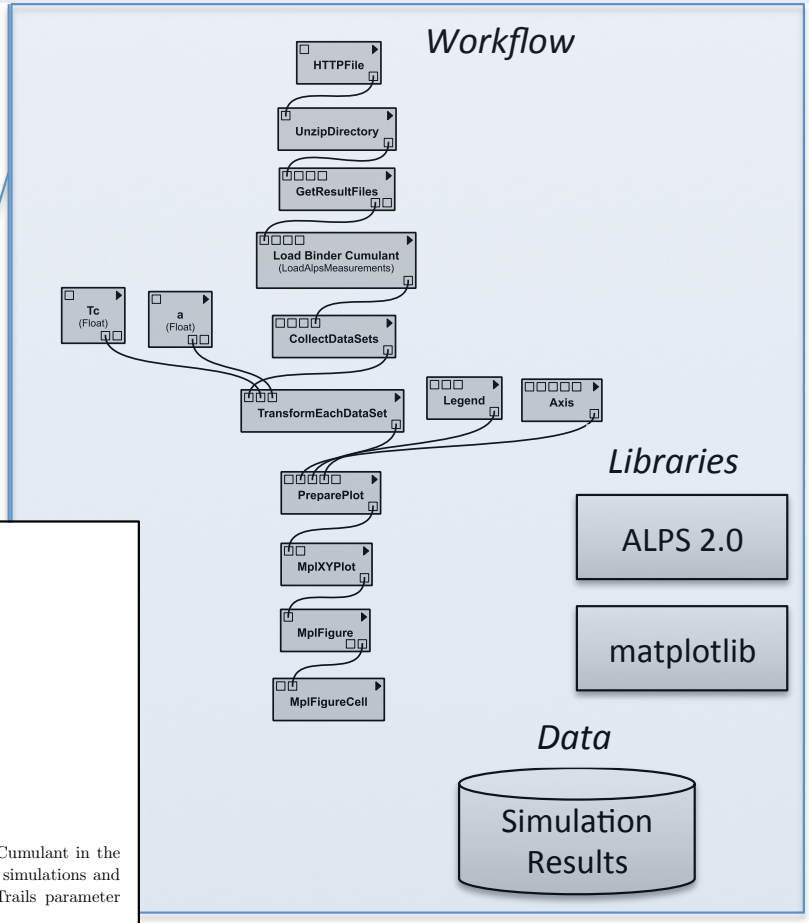


Figure 3. In this example we show a data collapse of the Binder Cumulant in the classical Ising model. The data has been produced by remotely run simulations and the critical exponent has been obtained with the help of the VisTrails parameter exploration functionality.



[Bauer et al., JSTAT 2011]

Courtesy of Juliana Freire (AMP Workshop on Reproducible research)

VCR: A Universal Identifier for Computational Results

Chronicing computations in real-time

VCR computation platform Plugin = Computation recorder

Regular program code

```
figure1 = plot(x)
save(figure1,'figure1.eps')
```

```
> file /home/figure1.eps saved
>
```

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)

VCR: A Universal Identifier for Computational Results

Chronicing computations in real-time

VCR computation platform Plugin = Computation recorder

Program code with VCR plugin

```
repository vcr.nature.com
verifiable figure1 = plot(x)
```

```
> vcr.nature.com approved:
> access figure1 at https://vcr.nature.com/ffaaffb148d7
```

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)

VCR: A Universal Identifier for Computational Results

Word-processor plugin App

LaTeX source

```
\includegraphics{figure1.eps}
```

LaTeX source with VCR package

```
\includeresult{vcr.thelancet.com/ffaaffb148d7}
```

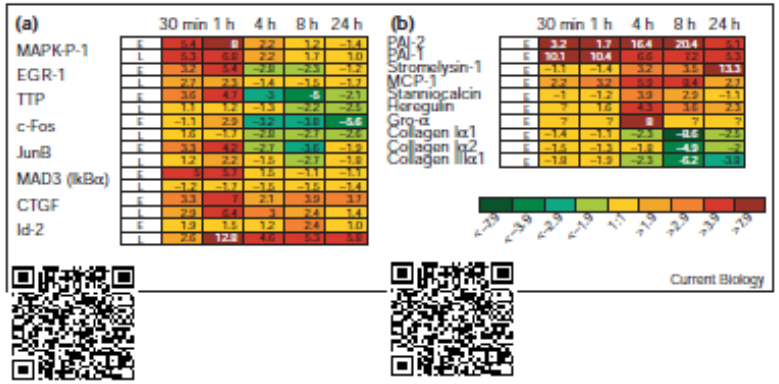
Permanently bind printed graphics to underlying result content

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)

VCR: A Universal Identifier for Computational Results

Figure 3

Time course of serum stimulation. **(a)** Early passage (E: PD30) or late passage (L: PD89) BJ cultures were held in 0.5% serum for 2 days, then stimulated with 10% FBS. RNA levels from cultures at the indicated time points (Cy5 channel) were compared with the uninduced starting culture (Cy3 channel). Positive values indicate higher expression in induced cells; negative values indicate lower expression in induced cells. Question marks indicate that there was insufficient signal for detection. A complete listing of serum-responsive genes from this analysis is provided in Supplementary material. **(b)** The serum-responsiveness of select senescence-regulated genes in early passage (PD30) BJ fibroblasts.



senescence response appears to overlap substantially with gene expression patterns observed in activated fibroblasts during wound healing [24–26]. MCP-1, Gro-α, IL-1β and IL-15 are strong effectors of macrophage and neutrophil recruitment and activation [27,28]. The upregulation of Toll (Tlr-4) in senescent fibroblasts confirms the overall immune response behavior at senescence. Tlr-4 is an IL-1 receptor homolog and is implicated in the activation of the gene regulatory protein NF-κB, a function proposed to be part of the innate immune response [29]. The induction of IL-15 at senescence is also consistent with an innate immune response, as IL-15 can be induced by NF-κB-dependent transcription [30] and also participates in inflammatory disease processes [28].

Deficiencies in the response of senescent cells to serum stimulation have been reported, and include an inability to induce the expression of c-*fos* mRNA [31] and markers of late G1 and S phase [32]. In response to serum, expression of inflammatory chemokines, matrix-degrading proteases and their modulators is induced in early-passage dermal fibroblasts, and expression of matrix collagens is reduced. This transient burst of activity may represent the natural response of these cells in wound repair [24]. Id-2 transcripts were hyper-induced in serum-stimulated senescent

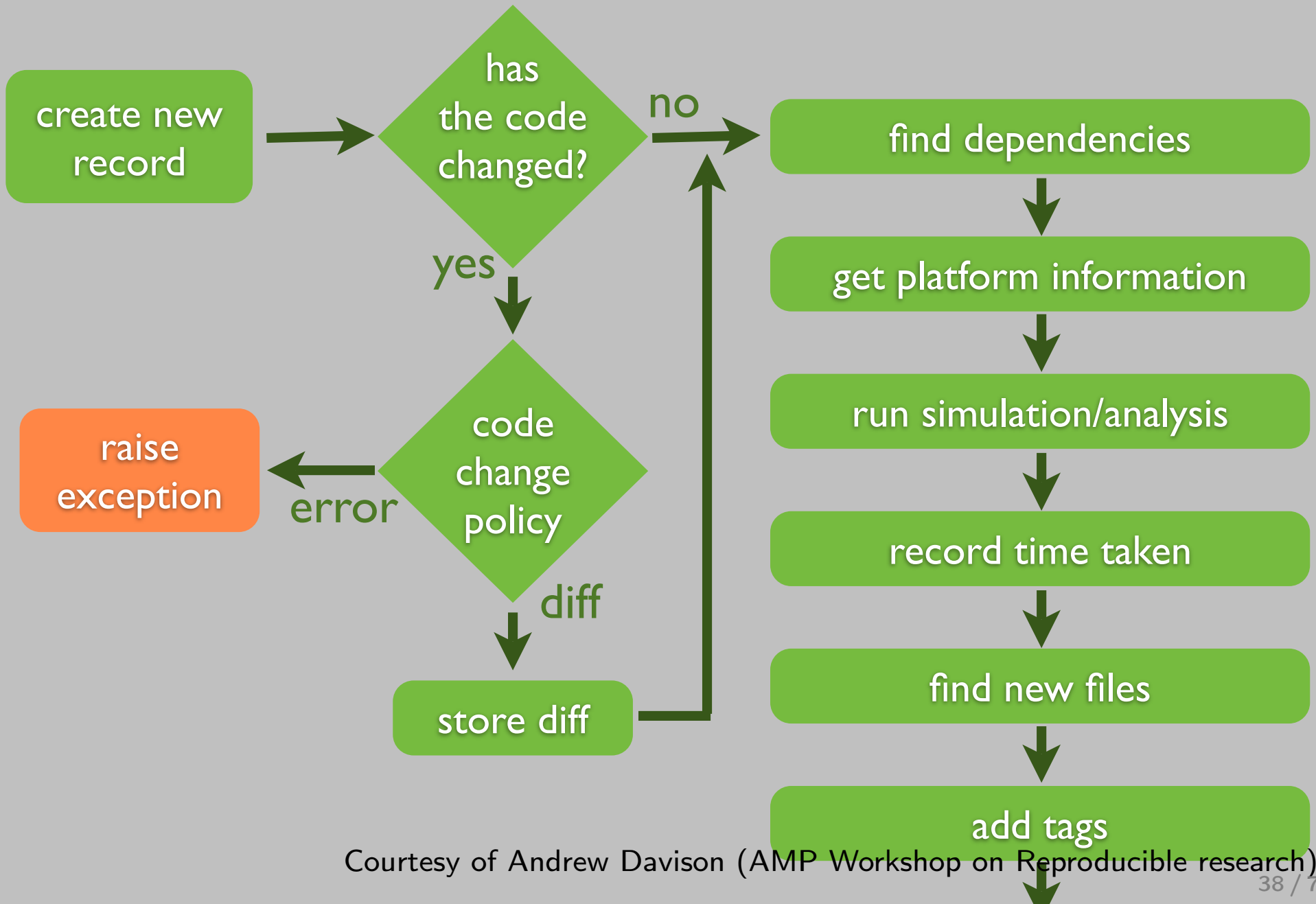
states overlap substantially with those in telomere-induced senescence (W.F., D.N.S., R. Allsopp, S. Lowe, and G. Ferbeyre, unpublished observations) and thus are likely to use many of the same activation processes.

The pattern of gene expression at senescence varies substantially in different cell types. Although the expression of matrix and structural proteins, such as the collagens, keratins and auxiliary factors, is repressed in RPE cells, inflammatory regulators are not induced, in contrast to dermal fibroblasts. Physiologically, this would make sense, as an acute inflammatory response in a tissue critical for normal vision would be likely to have deleterious consequences. However, as the RPE layer has a central role in the deposition and maintenance of extracellular matrix in the retina, decrements in the ability of senescent RPE cells to maintain appropriate expression patterns, as evidenced by decreased expression of collagens, keratins, aggrecan, transglutaminase and so on, would be predicted to have adverse affects on retinal architecture. Dysfunction of the RPE cell layer is considered to be a substantial factor in the development of age-related macular degeneration [36].

Surprisingly, early-passage HUVECs overexpress many of the markers associated with senescence in dermal fibro-

Courtesy of Matan Gavish and David Donoho (AMP Workshop on Reproducible research)

Sumatra: an "experiment engine" that helps taking notes



Courtesy of Andrew Davison (AMP Workshop on Reproducible research)

Sumatra: an "experiment engine" that helps taking notes

```
$ smt comment 20110713-174949 "Eureka! Nobel prize
here we come."
```

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)

Sumatra: an "experiment engine" that helps taking notes

```
$ smt tag “Figure 6”
```

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)

Sumatra: an "experiment engine" that helps taking notes

Sumatra: TestProject: List of records

http://127.0.0.1:8002/

Google

TestProject: List of records

Delete / include data	Label	Reason	Outcome	Duration	Processes	Simulator		Script			Date	Time	Tags
						Name	Version	Repository	Main file	Version			
☐	20100709-154255		'Eureka! Nobel prize here we come.'	0.59 s		Python	2.5.2	/Users/andrew/tmp/SumatraTest	main.py	396c2020ca50	09/07/2010	15:42:55	
☐	20100709-154309			0.59 s		Python	2.5.2	/Users/andrew/tmp/SumatraTest	main.py	396c2020ca50	09/07/2010	15:43:09	
☐	haggling	'determine whether the gourd is worth 3 or 4 shekels'	'apparently, it is worth NaN shekels.'	0.59 s		Python	2.5.2	/Users/andrew/tmp/SumatraTest	main.py	396c2020ca50	09/07/2010	15:43:20	foobar
☐	20100709-154338	'test effect of a smaller time constant'		0.59 s		Python	2.5.2	/Users/andrew/tmp/SumatraTest	main.py	396c2020ca50	09/07/2010	15:43:38	
☐	haggling_repeat	Repeat experiment haggling	The new record exactly matches the original.	0.58 s		Python	2.5.2	/Users/andrew/tmp/SumatraTest	main.py	396c2020ca50	09/07/2010	15:43:47	

Courtesy of Andrew Davison (AMP Workshop on Reproducible research)

New Tools for Computational Reproducibility

- Dissemination Platforms:
  - ResearchCompendia.org
  - IPOL
  - Madagascar
  - MLOSS.org
  - thedatahub.org
  - nanoHUB.org
  - Open Science Framework
  - The DataVerse Network
  - RunMyCode.org
- Workflow Tracking and Research Environments:
  - VisTrails
  - Kepler
  - CDE
  - Galaxy
  - GenePattern
  - Synapse
  - Sumatra
  - Taverna
  - Pegasus
- Embedded Publishing:
  - Verifiable Computational Research
  - Sweave
  - knitR
  - Collage Authoring Environment
  - SHARE

Courtesy of Victoria Stodden (UC Davis, Feb 13, 2014)

And also: **Figshare**, **ActivePapers**, **Elsevier executable paper**, ...

Outline

Structure

Research articles are often structured in this basic order:

Introduction Why was the study undertaken? What was the research question, the tested hypothesis or the purpose of the research?

Methods When, where, and how was the study done? What materials/hardware were used? How was it configured?

Results What answer was found to the research question; what did the study find? Was the tested hypothesis true? Present useful results in a synthetic way with a logical order.

Discussion What might the answer imply and why does it matter? How does it fit in with what other researchers have found? What are the possible bias and points to improve? What are the perspectives for future research?

Such structure facilitates literature review and is a very effective way to convey information.
If the report is a few pages long then an abstract is required.

Outline

Step 0: Taking Notes

Document your:

- Hypotheses: keep track of your ideas/line of thoughts
- Experiments: details on how and why an experiment was run, including failed or ambiguous attempts.
- Initial analysis or interpretation of these experiments: was the outcome conform to the expectation or not? does it (in)validate the hypothesis?
- Organization: keep track of things to do/fix/test/improve

Structure:

1. General information about the document and organization conventions (e.g., directory structure, notebook structure, experimental result storing mechanism, . . .)
2. Documentation of commonly used commands and of how to set up experiments (e.g., git cloning, environment deployment, connection to machines, compiling scripts)
3. Experiment results can be either structured by dates ($\rightsquigarrow$ add tags) or by experiment campaigns ($\rightsquigarrow$ add date/time)

Which format should I use ?

- Wikis are encouraged to favor collaboration but I do not find them really effective
- Blogging systems are also a way of managing such notebook but they should rather be considered as an effective way to share information with others
- I recommend to use basic plain-text format and to structure it hierarchically

  Here is a link to an excerpt of the journal of one of my PhD student, managed with git/org-mode. More detailed links are given in slide 69.

Last but not least:

Provide links to Raw Data!!!

When/How Often Should I Use it?

I have a very intense usage (demo to general journal and specific BOINC journal) and I tend to capture a lot of information but you do not have to be as extreme as I am. Here are a few advices:

- Spending more than an hour without at least writing what you're working on is not right...
  - Take a 5 minutes break and ask yourself what you're doing, what is keeping you busy and where all this is leading you
- While working on something, you will often notice/think about something you should fix/improve but you just don't want to do it now. Take 20 seconds to write a TODO entry.
- There are moments where you have to wait for something (compiling, deployment, ...). It is generally the perfect time for improving your notes (e.g., detail the steps to accomplish a TODO entry).
- By the end of the day: daily (and weekly) review!
  - Update your lists, write what the next steps are
  - Summarize in a 2-4 lines (for your advisor) what you did, what was difficult, what you learnt.

Step 1: Sharing Code and Data

What kinds of systems are available?

- "Good" - The cloud (Dropbox, Google Drive, Figshare)
- Better - Version control systems (SVN, Git and Mercurial)
- "Best" - Version control systems on the cloud (GitHub, Bitbucket)

Depends on the level of privacy you expect but you probably already know these tools. **Few handle GB files...**

Is this enough?

1. Use a workflow that documents both data and process
2. Use the machine readable CSV format
3. Provide raw data and meta data, not just statistical outputs
4. Never do data manipulation and statistical tests by hand
5. Use R, Python or another free software to read and process raw data (ideally to produce complete reports with code, results and prose)

Courtesy of Adam J. Richards

Step 2: Literate Programming

Donald Knuth: explanation of the program logic in a natural language interspersed with snippets of macros and traditional source code.

I'm way too 3133t to program this way but that's
exactly what we need for writing a reproducible article/analysis!

Org-mode (requires emacs)

My favorite tool.

- plain text, very smooth, works both for html, pdf, ...
- allows to combine all my favorite languages even with sessions

Ipython notebook

If you are a python user, go for it! Web app, easy to use/setup...

KnitR (a.k.a. Sweave)

For non-emacs users and as a first step toward *reproducible papers*:

- Click and play with a modern IDE (e.g., Rstudio)

Outline

Why R?

R is a great language for data analysis and statistics

- Open-source and multi-platform
- Very expressive with high-level constructs
- Excellent graphics
- Widely used in academia and business
- Very active community
  - Documentation, FAQ on http://stackoverflow.com/questions/tagged/r
- Great integration with other tools

Why is such R a pain for computer scientists?

- R is not really a programming language
- Documentation is for statisticians
- Default plots are ~~cumbersome~~ (meaningful)
- Summaries are ~~cryptic~~ (precise)
- Steep learning curve even for us, computer scientists whereas we generally switch seamlessly from a language to another! That's frustrating! ;)

Do's and dont's

~~R is high level, I'll do everything myself~~

- CTAN comprises 4,334 TEX, LATEX, and related packages and tools. Most of you do not use plain TEX.
- Currently, the CRAN package repository features 4,030 available packages.
- How do you know which one to use??? Many of them are highly exotic (not to say useless to you).

  I learnt with http://www.r-bloggers.com/
- Lots of introductions but not necessarily what you're looking for so I'll give you a short tour.
  You should quickly realize though that you need proper training in statistics and data analysis if you do not want tell nonsense.
- Again, you should read Jain's book on The Art of Computer Systems Performance Analysis
- You may want to follow online courses:
  - https://www.coursera.org/course/compdata
  - https://www.coursera.org/course/repdata

Install and run R on debian

```
apt-cache search r
```

Err, that's not very useful :) It's the same when searching on google but once the filter bubble is set up, it gets better...

```
sudo apt-get install r-base
```

```
R
```

```
R version 3.2.0 (2015-04-16) -- "Full of Ingredients"
Copyright (C) 2015 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.
```

Install a few cool packages

R has it's own package management mechanism so just run R and type the following commands:

- `ddply`, `reshape` and `ggplot2` by Hadley Wickham (http://had.co.nz/)

```
install.packages("plyr")
  # or better: install.packages("dplyr")
install.packages("reshape")
  # or better; install.packages("tidyr")
install.packages("ggplot2")
```

- `knitR` by (Yihui Xie) http://yihui.name/knitr/

```
install.packages("knitr")
```

IDE

Using R interactively is nice but quickly becomes painful so at some point, you'll want an IDE.

Emacs is great but you'll need *Emacs Speaks Statistics*

```
sudo apt-get install ess
```

In this tutorial, I will briefly show you rstudio (https://www.rstudio.com/) and later how to use `org-mode`

Outline

Rstudio screenshot

RStudio

File Edit Code View Project Workspace Plots Tools Help

markdown-introduction.rmd | example-r-markdown.rmd

Knit HTML | Run | Chunks

```
28
29 ```{r basicconsole}
30 x <- 1:10
31 y <- round(rnorm(10, x, 1), 2)
32 df <- data.frame(x, y)
33 df
34 ```
35
36 ## Plots
37 Images generated by `knitr` are saved in a figures folder. However,
   they also appear to be represented in the HTML output using a [data
   URI scheme]( http://en.wikipedia.org/wiki/Data_URI_scheme). This
   means that you can paste the HTML into a blog post or discussion
   forum and you don't have to worry about finding a place to store the
   images; they're embedded in the HTML.
38
39 ### Simple plot
40 Here is a basic plot using base graphics:
41
42     ```{r simpleplot}
43     plot(x)
44     ```
45
46 ```{r simpleplot}
47 plot(x)
```

47:1 | Chunk 3: simpleplot | R Markdown

Workspace | History

Load | Save | Import Dataset | Clear All

Data	
df	10 obs. of 2 variables
Values	
x	integer[10]
y	numeric[10]

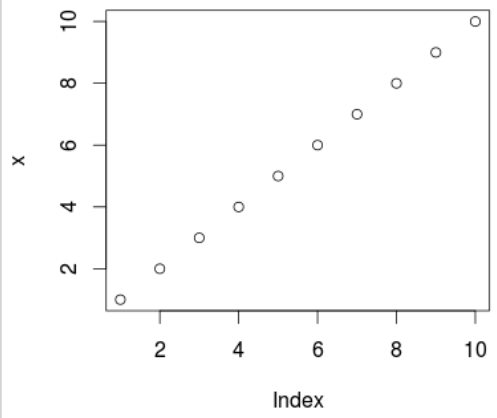
Console ~/research/statistics/rmarkdown-rmeetup-2012/

```
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> set.seed(1234)
> library(ggplot2)
> library(lattice)
> x <- 1:10
> y <- round(rnorm(10, x, 1), 2)
> df <- data.frame(x, y)
> df
    x    y
1   1 1.31
2   2 2.31
3   3 3.36
4   4 3.27
5   5 5.04
6   6 6.11
7   7 8.43
8   8 8.98
9   9 8.38
10 10 9.27
> plot(x)
```

Files | Plots | Packages | Help

Zoom | Export | Clear All



Reproducible analysis in Markdown + R

- Create a new R Markdown document (Rmd) in rstudio
- R chunks are interspersed with ```` ```{r} ```` and ```` ``` ````
- Inline R code: `` `r sin(2+2)` ``
- You can knit the document and share it via rpubs
- R chunks can be sent to the top-level with `Alt-Ctrl-c`
- I usually work mostly with the current environment and only knit in the end
- Other engines can be used (use rstudio completion)

  ```
  ```{r engine='sh'}
  ls /tmp/
  ```
  ```

- Makes reproducible analysis as simple as one click
- Great tool for quick analysis for self and colleagues, homeworks, ...

Reproducible articles with LaTeX + R

- Create a new R Sweave document (Rnw) in rstudio
- R chunks are interspersed with <<>>= and @
- You can knit the document to produce a pdf
- You'll probably quickly want to change default behavior (activate the cache, hide code, ...). In the preembule:

```
<<echo=FALSE>>=
opts_chunk$set(cache=TRUE,dpi=300,echo=FALSE,fig.width=7,
                    warning=FALSE,message=FALSE)
@
```

- Great for journal articles, theses, books, ...

Outline

Data frames

A data frame is a data tables (with columns and rows). `mtcars` is a built-in data frame that we will use in the sequel

```
head(mtcars);
```

```
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

You can also load a data frame from a CSV file:

```
df <- read.csv("http://foo.org/mydata.csv", header=T,
       strip.white=TRUE);
```

You will get help by using `?`:

```
?data.frame
?rbind
?cbind
```

Exploring Content (1)

```
names(mtcars);

 [1] "mpg"  "cyl"  "disp" "hp"   "drat" "wt"   "qsec" "vs"   "am"
[11] "carb"

str(mtcars);

'data.frame': 32 obs. of  11 variables:
 $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
 $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
 $ disp: num  160 160 108 258 360 ...
 $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
 $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
 $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
 $ qsec: num  16.5 17 18.6 19.4 17 ...
 $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
 $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
 $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
 $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

Exploring Content (2)

```
dim(mtcars);
length(mtcars);
```

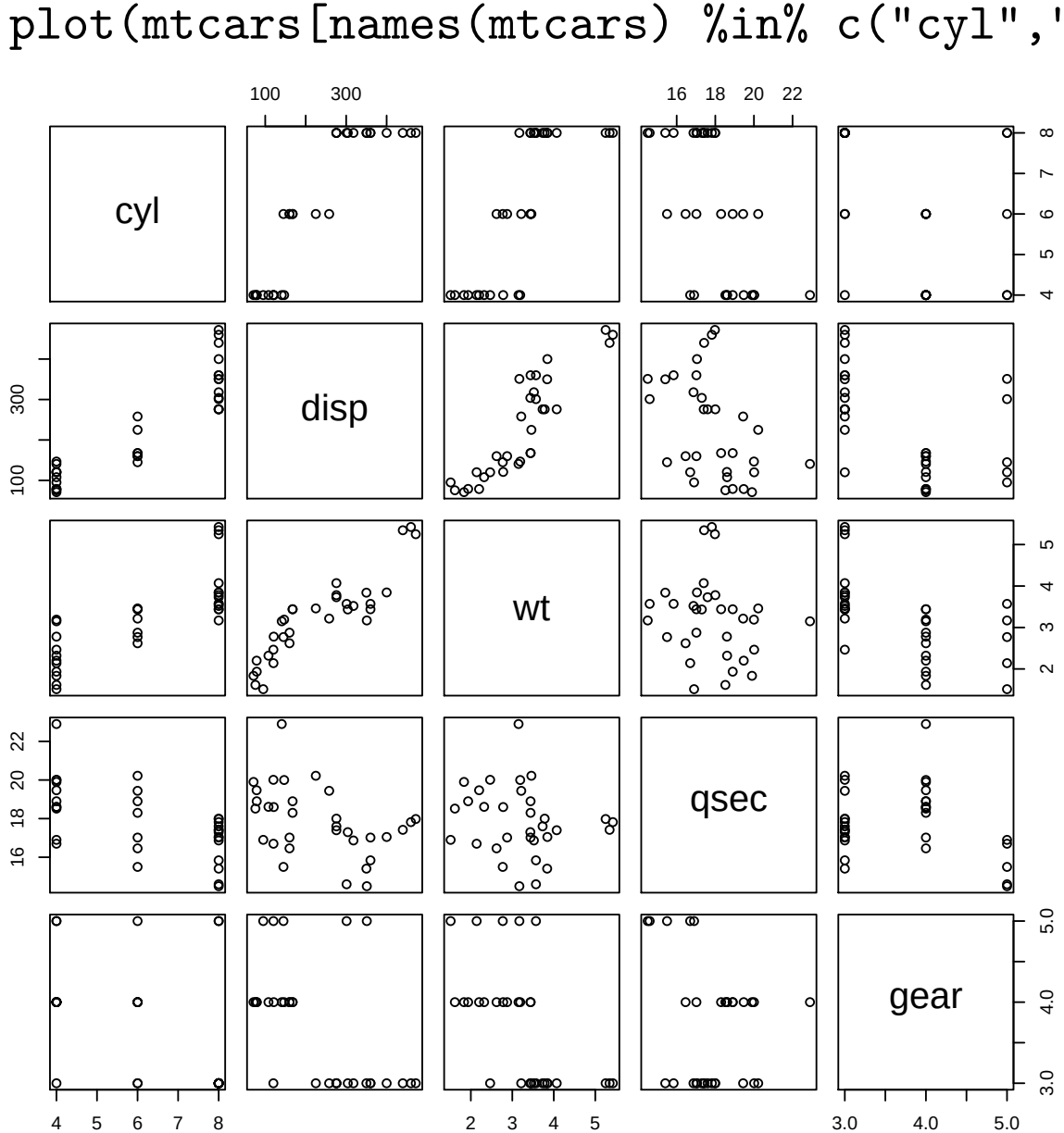
```
[1] 32 11
[1] 11
```

```
summary(mtcars);
```

```
      mpg             cyl             disp             hp
 Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
 1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
 Median :19.20   Median :6.000   Median :196.3   Median :123.0
 Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
 3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
 Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
      drat             wt             qsec             vs
 Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
 1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
 Median :3.695   Median :3.325   Median :17.71   Median :0.0000
 Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
 3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
```

Exploring Content (3)

```
plot(mtcars[names(mtcars) %in% c("cyl","wt","disp","qsec","gear")]);
```



Accessing Content

```
mtcars$mpg
```

```
 [1] 21.0 21.0 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 17.8 16.4 17.
[16] 10.4 14.7 32.4 30.4 33.9 21.5 15.5 15.2 13.3 19.2 27.3 26.0 30.
[31] 15.0 21.4
```

```
mtcars[2:5,]$mpg
```

```
[1] 21.0 22.8 21.4 18.7
```

```
mtcars[mtcars$mpg == 21.0,]
```

```
               mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4       21   6  160 110  3.9 2.620 16.46  0  1    4    4
Mazda RX4 Wag   21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

```
mtcars[mtcars$mpg == 21.0 & mtcars$wt > 2.7,]
```

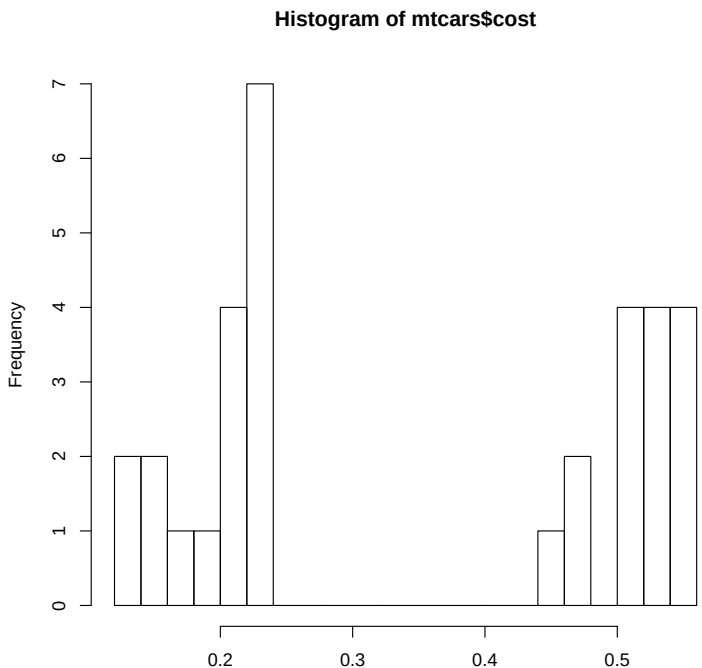
```
               mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4 Wag   21   6  160 110  3.9 2.875 17.02  0  1    4    4
```

Extending Content

```
mtcars$cost = log(mtcars$hp)*atan(mtcars$disp)/
 sqrt(mtcars$gear**5);
mean(mtcars$cost);
summary(mtcars$cost);
```

```
[1] 0.345994
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
 0.1261  0.2038  0.2353  0.3460  0.5202  0.5534
```

```
hist(mtcars$cost,breaks=20);
```



Take away Message

- R is a great tool but is only a tool. There is no magic. You need to understand what you are doing and get a minimal training in statistics
- It is one of the building block of reproducible research (the *reproducible analysis* block) and will save you a lot of time
- It provides you an access to any statistical method you ever dreamt of
- Read at least Jain's book: The Art of Computer Systems Performance Analysis
- There are introductory online courses (from John Hopkins university) on coursera which you may want to follow

Outline

Literate Programming on a Daily Basis

Mastering Emacs

- `C-g`: get me out of here!
- `C-_`: undo
- Activate CUA keys in the Options menu

Mastering Org-mode

- `Tab` will fold/unfold stuff
- `C-c C-c`: do something (context-sensitive) where you are
- `<s + Tab`, `<b`, `<l`, `<r`, `<h`, `...` for creating code blocks
- `C-c C-e`: export
- `C-c c`: capture content
- `C-c C-o` / `C-c l` / `C-c C-l`: open/store/insert links
- `C-c C-a`: attach a file
- `C-c C-d`: set deadline, `C-c C-t`: TODO/DONE

Emacs/Org-mode Recap

Key features

- Plain text makes it very robust and human readable
- Allow to mix any language and has a notion of session that makes its use very effective
- Allow to produce both `html documents`, classical LaTeXarticles, beamer slides, `odt` documents, ... Native pretty printing on Github

A Few Links to Learn More

- *Org for beginners*, *my emacs configuration* and *tricks for Mac OS X users*
- A *script* capturing and gathering many information into a *single result document*
- A *laboratory notebook* with notes about all the experiments performed since the beginning of the project
- *Litterately conducting experiments* using org-mode

This was way too much information...

... but keep these slides in mind and re-read them later. You will follow many links when you will realize what they can bring to you.

- We need to put all this in practice.
- During this semester, you will learn how to improve your methodology
- You will apply analysis and reporting techniques to a simple use case:

  One of your colleague just implemented a multi-threaded version of the quicksort algorithm for multi-core machines. He's convinced his code can save significant time saving but unfortunately, he did not follow the performance evaluation lecture and he would like your help to promote his code.

- After you have tried, we will debrief on what you did and discuss how it could be improved