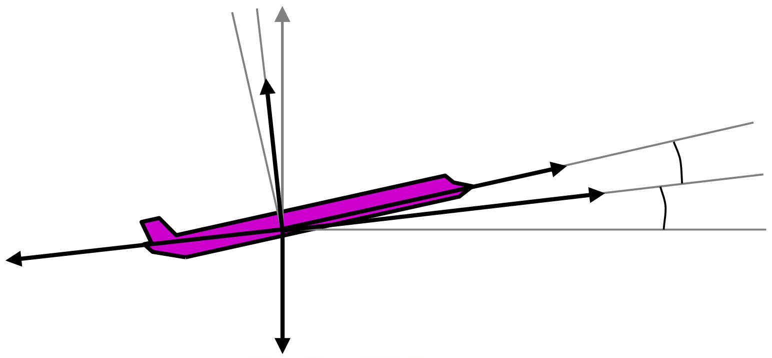
Reproducible Research: Failures, Successes, Challenges and (Re)Setting the Bar

Ian M. Mitchell
Department of Computer Science
University of British Columbia

research supported by
the Natural Science and Engineering Research Council of Canada

Outline

- Motivation: Some lessons in Irreproducible Research
- Computation and Data Science
- Reproducible Research: Changing the Culture
- The HSCC Experience
- Code != Data



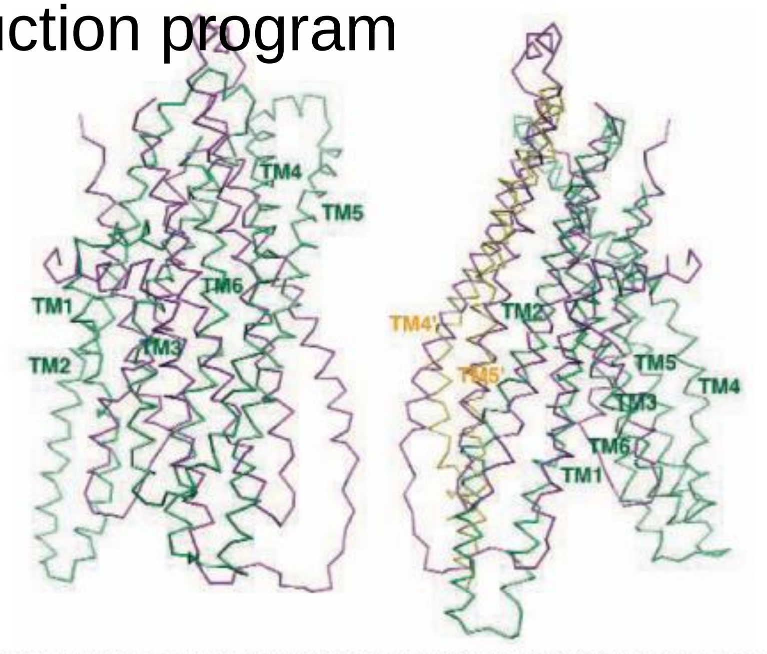
"Your recent Amazon purchases, Tweet score and location history makes you 23.5% welcome here."

Thierry Gregorius

Accurate to within ±One (Hundred Percent)

- 2001–2005: Geoffrey Chang and colleagues published a number of high profile protein structures
  - 2001 paper on MsbA cited 360+ times by 2006
- September 2006: A dramatically different structure for a related protein is published
- December 2006: Chang et al retract five papers because “An in-house data reduction program introduced a change in sign…”

Image from:
Miller, "A Scientist's Nightmare: Software Problem leads to Five Retractions" in *Science* 314(5807): 1856-1857 (22 December 2006)



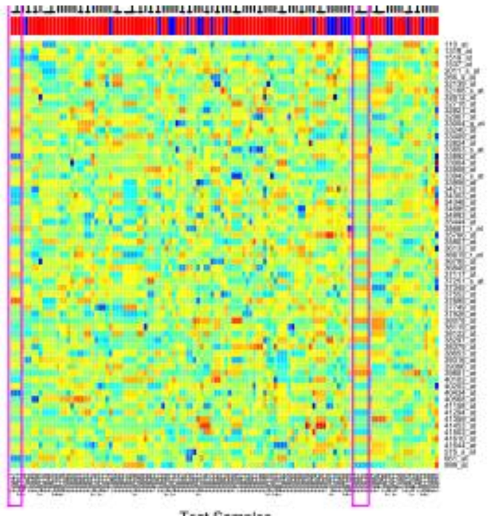
Flipping fiasco. The structures of MsbA (purple) and Sav1866 (green) overlap little (*left*) until MsbA is inverted (*right*).

A Simple Labelling Mistake?

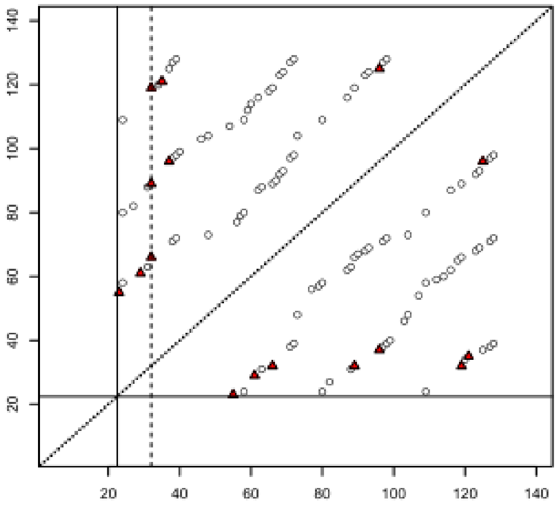
- 2006: Anil Potti and colleagues announce method for predicting patient response to chemotherapy drugs based on gene microarray data
  - 200+ citations by 2009
- 2007: Clinical trials begin
- 2007–2009: Baggerly, Coombes and colleagues try to reproduce results, but find frequent inconsistencies
- 2010–2011: Trials stopped, Potti resigns, 7+ retractions

Images from:
Baggerly & Coombes, "Deriving chemosensitivity from cell lines: Forensic bioinformatics and reproducible research in high-throughput biology" in *Annals of Applied Statistics* 3(4): 1309-1334 (2009)

Response Labelling +
Gene Expression Heatmap



Repeated Columns
(Δ: inconsistent labels)



It's Only the Global Economy

- 2010: Reinhart & Rogoff:

"...whereas the link between growth and debt seems relatively weak at 'normal' debt levels, median growth rates for countries with public debt over roughly 90% of GDP are about one percent lower than otherwise; average (mean) growth rates are several percent lower."

  - Common justification for austerity measures
- 2013: Herndon, Ash & Pollin, unable to recreate results from raw data receive original spreadsheet from RR
  - Discover several discrepancies including that the first five "advanced economies" (alphabetically) were omitted from first calculation

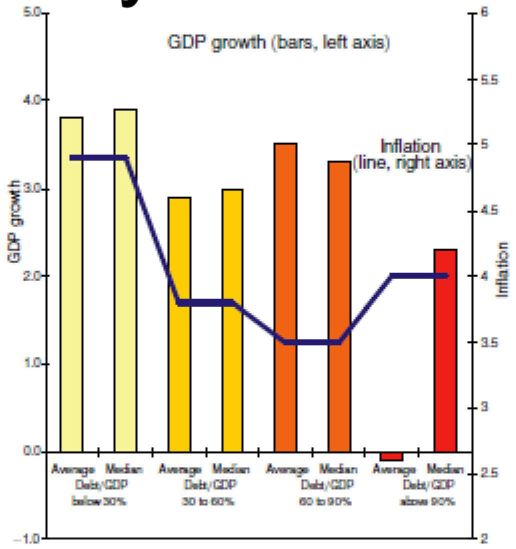
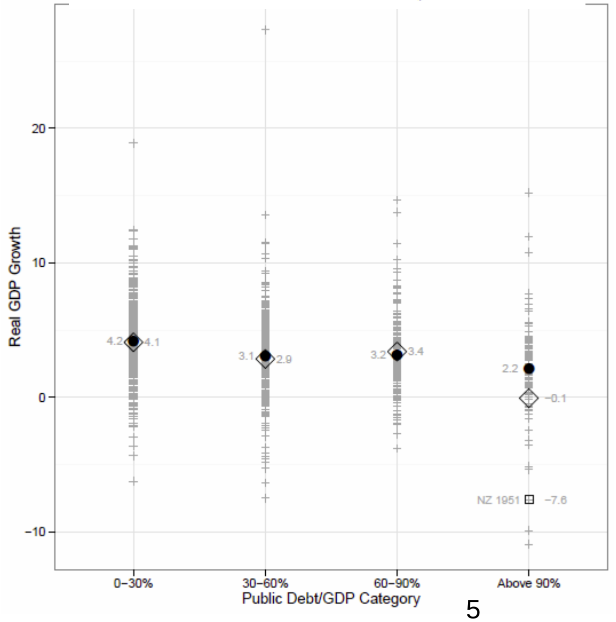


Figure 2. Government Debt, Growth, and Inflation: Selected Advanced Economies, 1946–2009



Images from:
Reinhart & Rogoff, "Growth in a Time of Debt," in *American Economic Review* 100:573-578 (2010) and Herndon, Ash & Pollin, "Does High Public Debt Consistently Stifle Economic Growth? A Critique of Reinhart and Rogoff" Political Economy Research Institute Working Paper (April 2013)

Why so Secretive?

- 2005: Wicherts and colleagues requested data from 49 papers recently published in two highly ranked American Psychological Association journals (part of a larger study)
  - Corresponding authors had signed publication form agreeing to share data
  - 21 shared some data, 3 refused (lost or inaccessible data), 12 promised to later but did not, and 13 never responded
- 2011: Wicherts and colleagues analyze internal consistency of p-values reported from null hypothesis tests
  - Willingness to share is correlated with fewer reporting errors and relatively stronger evidence against NH

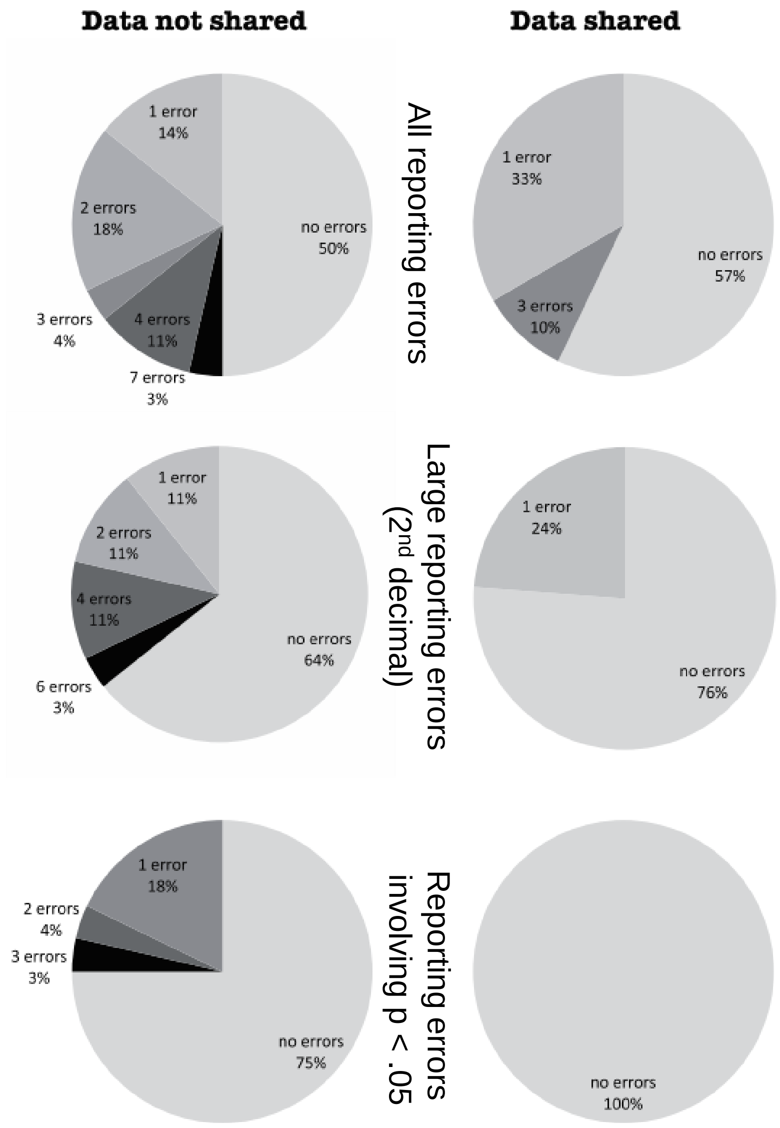
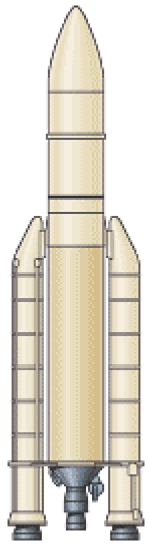


Image from:
Wicherts, Bakker & Molenaar, "Willingness to share research data is related to the strength of the evidence and the quality of reporting of statistical results" in *PLoS ONE* 6(11), Nov. 2011.

Disasters in Numerical Computing

- Feb 25, 1991: Patriot missile battery fails to track a incoming Scud missile
  - Error caused by rounding error in 24 bit timer
- August 23, 1991, Sleipner A oil platform collapses and sinks when first submerged
  - Error in finite element analysis of the strength of key concrete support structures
- June 4, 1996: maiden Ariane 5 rocket's guidance fails leading to self-distruct
  - Error caused by overflow stemming from sloppy software reuse and parameter modification

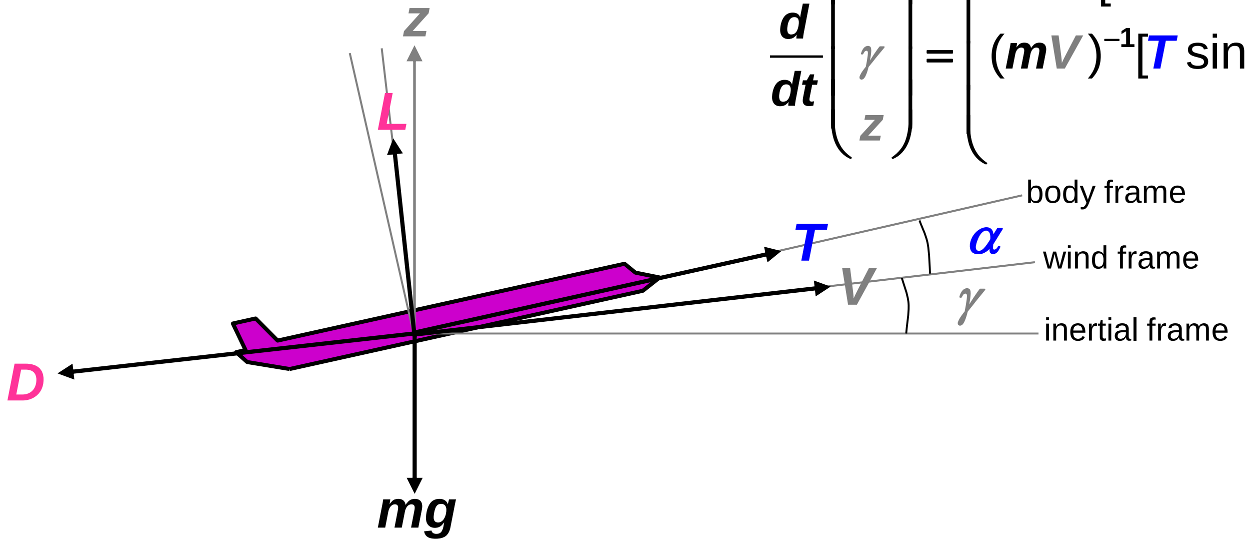


Examples from Douglas N. Arnold
http://www.ima.umn.edu/~arnold/disasters

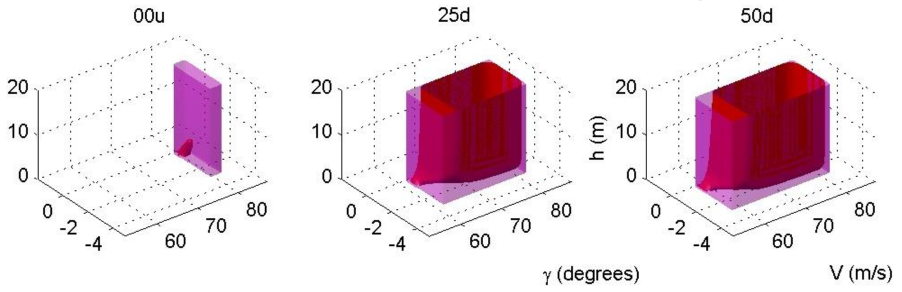
A Personal Example

- Study of safe flap settings during aircraft final approach to runway
  - Publication: Bayen, Mitchell, Oishi & Tomlin, "Aircraft Autolander Safety Analysis Through Optimal Control-Based Reach Set Computation" in *AIAA Journal of Guidance, Control & Dynamics*, 30(1): 68–77 (2007).

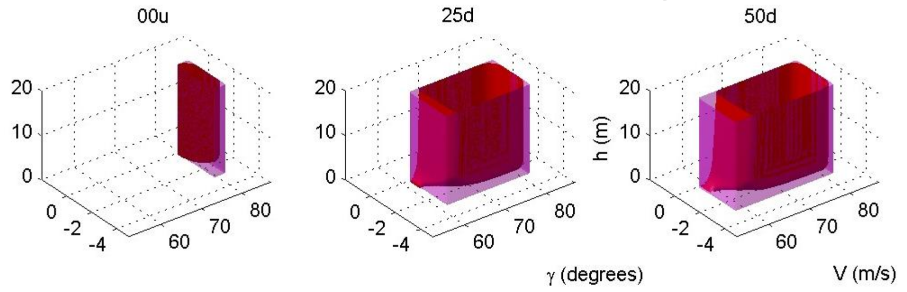
$$\frac{d}{dt}\begin{pmatrix} V \\ \gamma \\ z \end{pmatrix} = \begin{pmatrix} m^{-1}[T\cos\alpha - D(\alpha,V) - mg\sin\gamma] \\ (mV)^{-1}[T\sin\alpha + L(\alpha,V) - mg\cos\gamma] \\ V\sin\gamma \end{pmatrix}$$



without mode switching



with mode switching



Could You Send Me the Code?

- Which directory was that in?

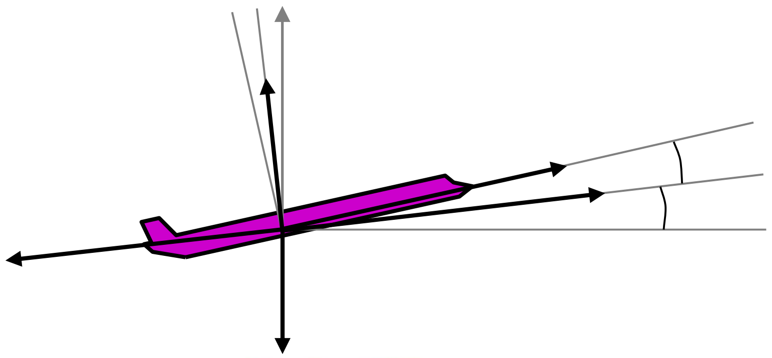
```
~/OldStanfordGagarin/Cyghome/Source/HS01/Landing/
~/OldStanfordGagarin/Cyghome/Source/Projection/Working/
~/OldStanfordGagarin/Cyghome/Source/JCP/
~/OldStanfordGagarin/Cyghome/Papers/AIAA02/Source/
~/OldStanfordGagarin/Winhome/VisualStudioProjects/LandingHighD/
~/OldVonBraun/CygHome/Papers/AIAA03/Landing/Source
~/OldVonBraun/CygHome/Papers/AIAA03/Landing/Shriram
```

- Which parameters did I use?

```
// 70% of 160e3 is 112e3
//  assumes fixed thrust at minT (see Flow::hamiltonian() function)
//const GradValue ModeMinT =   0e3;
//const GradValue ModeMaxT = 160e3;
//const GradValue ModeMinT = 32e3;
//const GradValue ModeMaxT = 32e3;
```

Outline

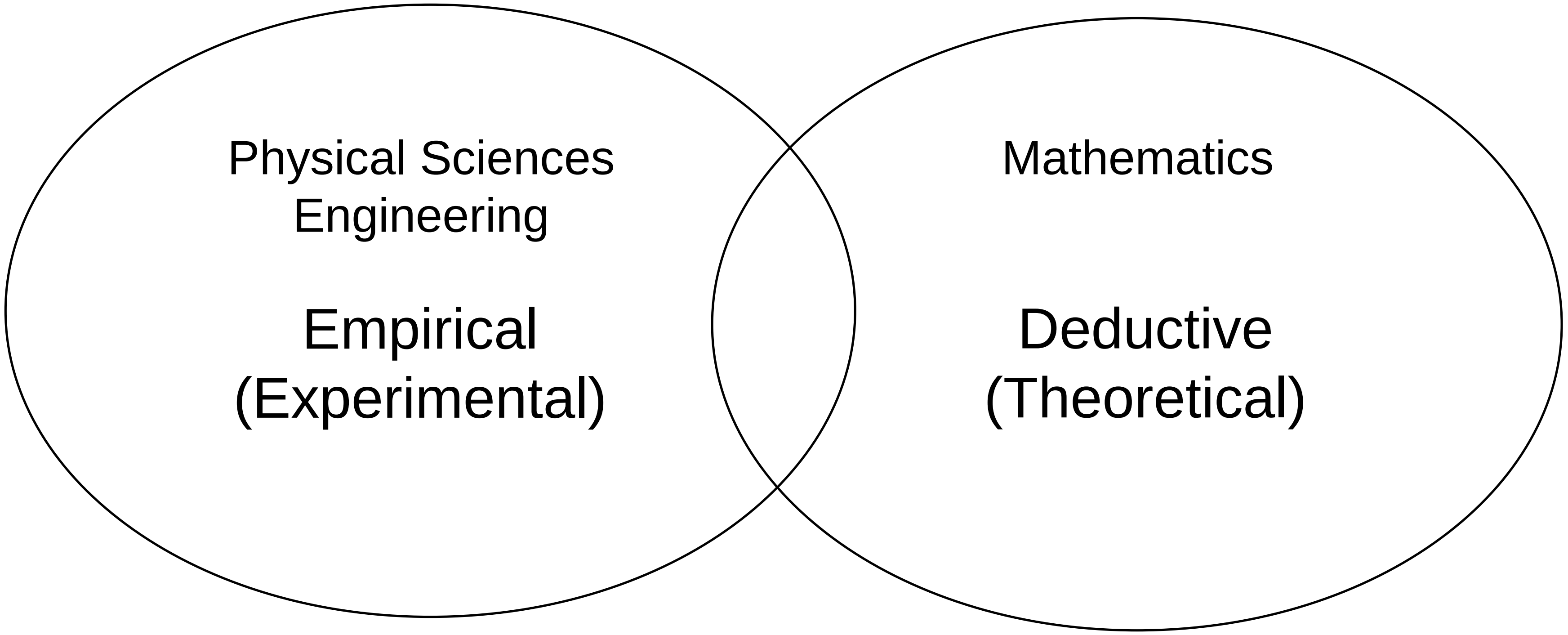
- Motivation: Some lessons in Irreproducible Research
- Computation and Data Science
- Reproducible Research: Changing the Culture
- The HSCC Experience
- Code != Data



"Your recent Amazon purchases, Tweet score and location history makes you 23.5% welcome here."

Thierry Gregorius

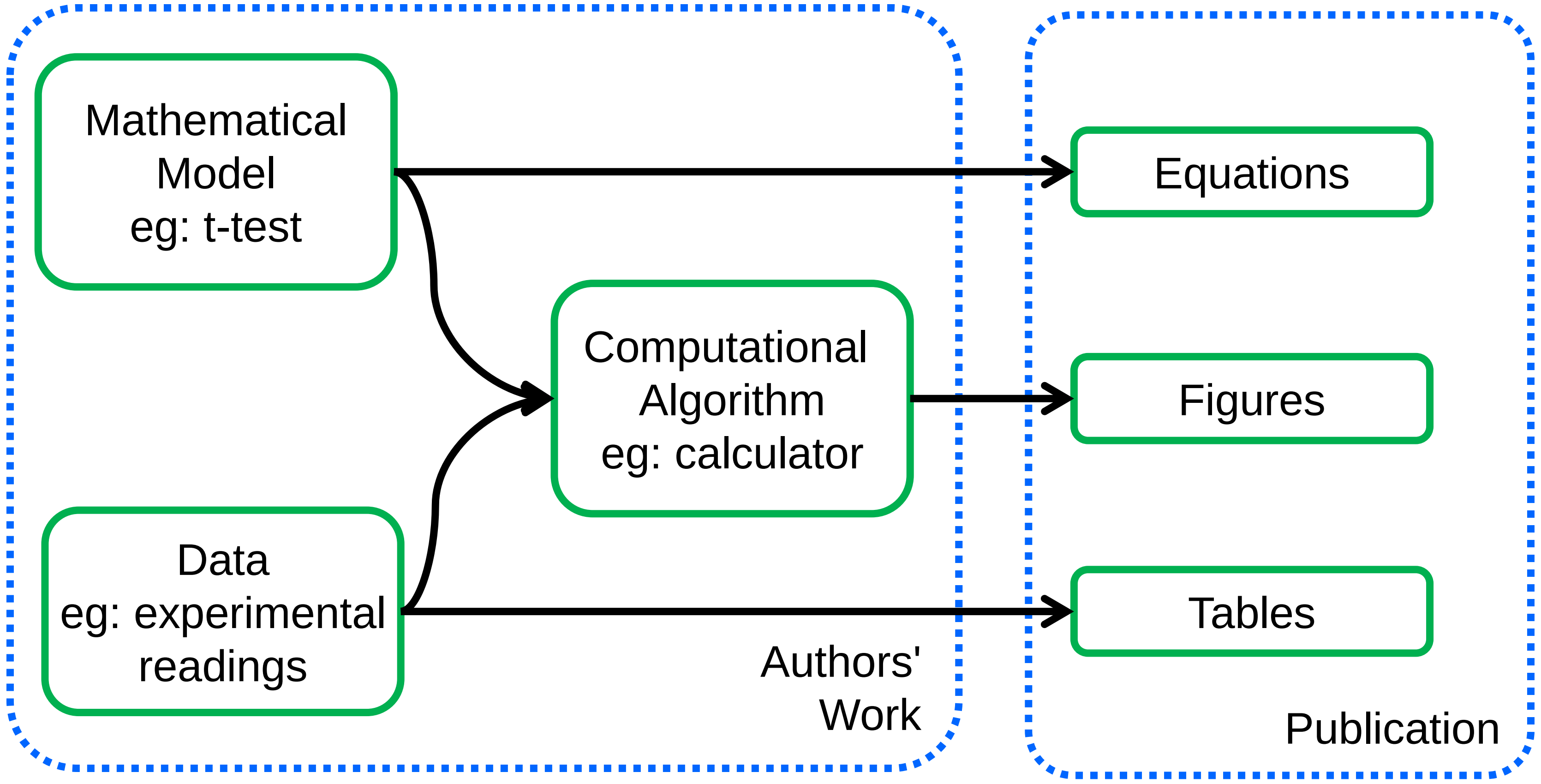
Exploring the World



- Traditionally, scientists used two approaches to build knowledge about the world
  - Data was gathered and processed by hand through simple procedures (eg: statistical summaries)

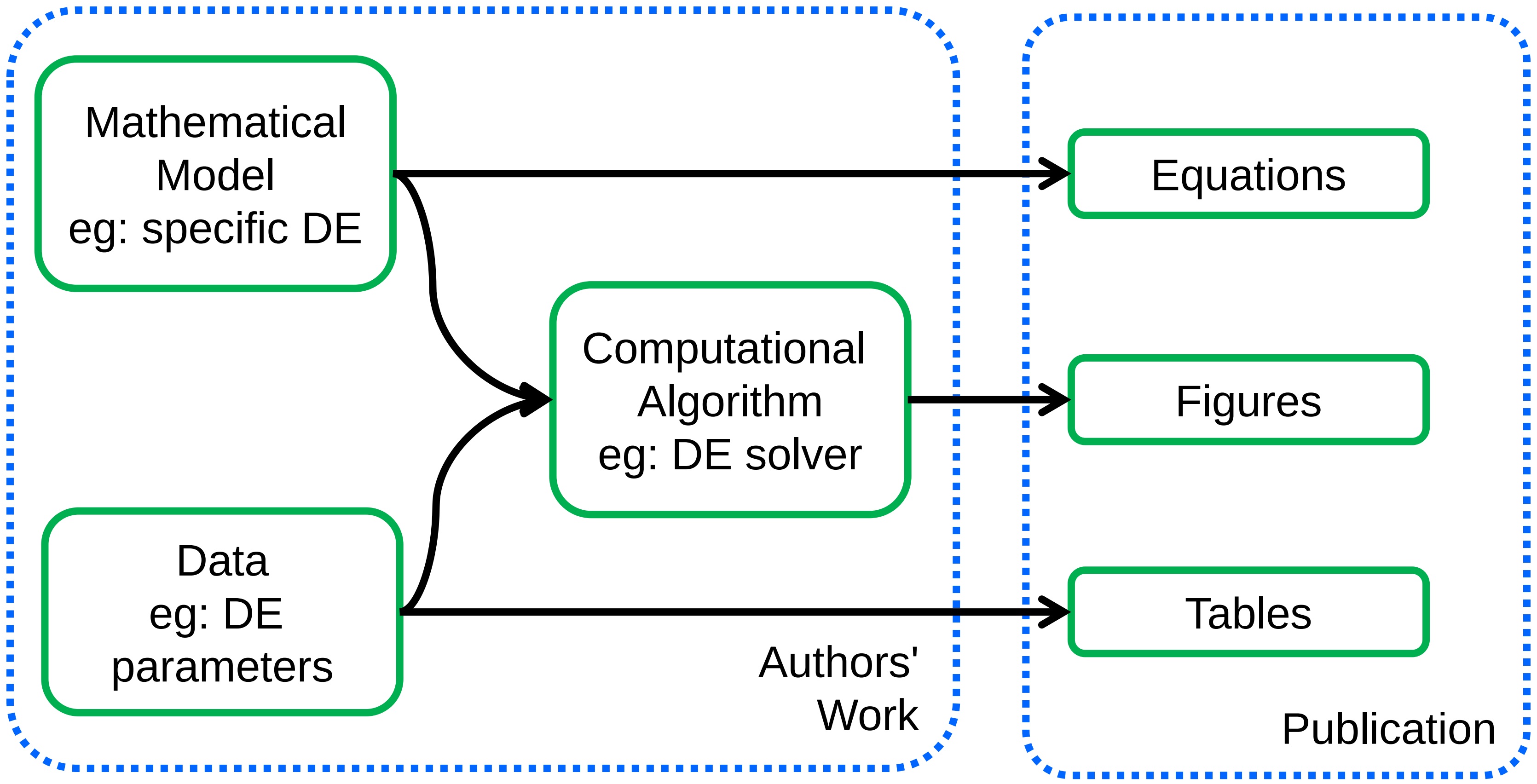
What Came Before

- Computational support for experimental analysis
  - Example: Is my hypothesis valid?


Mathematical Model eg: t-test
Computational Algorithm eg: calculator
Data eg: experimental readings
Authors' Work
Equations
Figures
Tables
Publication

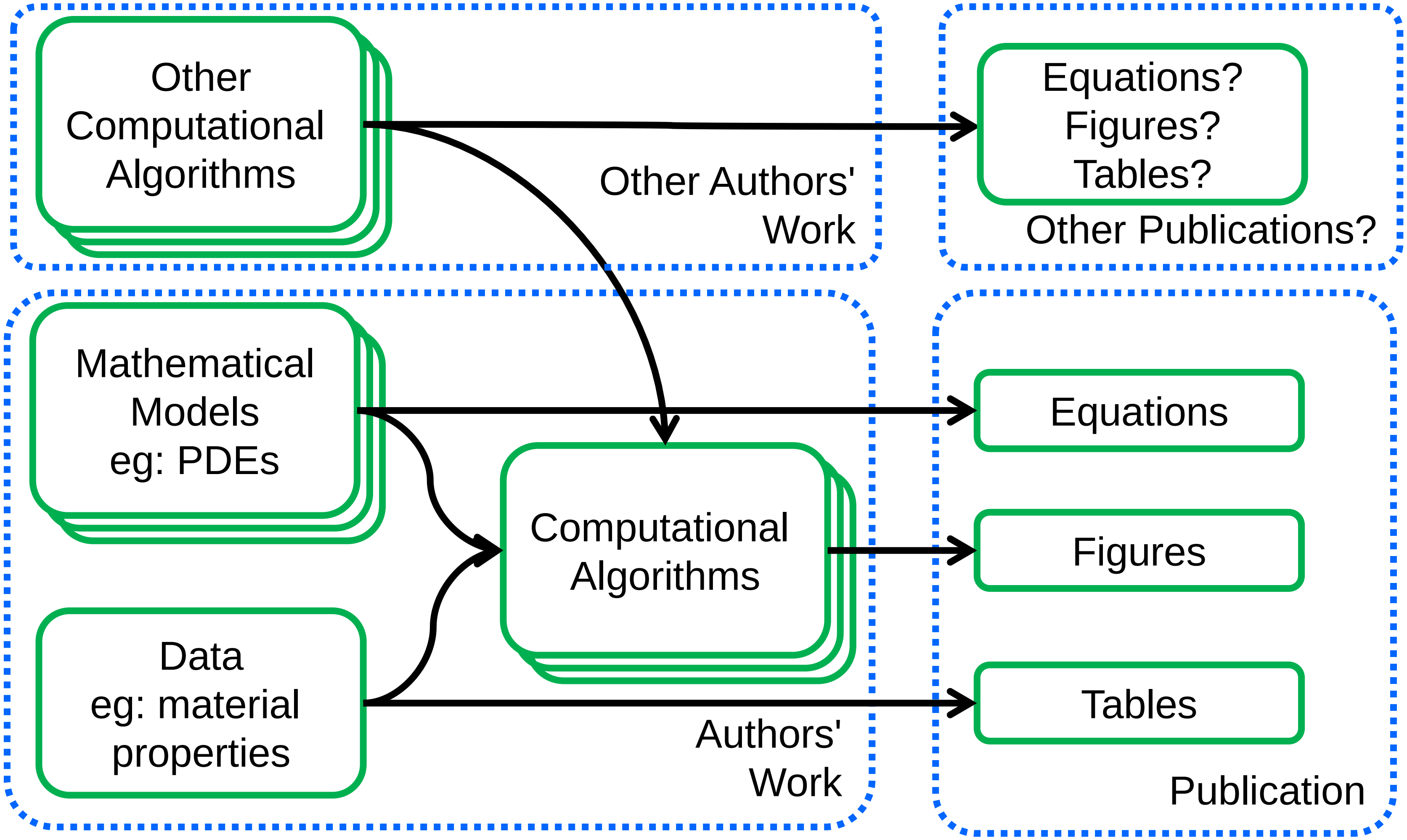
What Came Before

- Computational support for theoretical analysis
  - Example: Is my differential equation (DE) solver stable?


Mathematical Model eg: specific DE
Computational Algorithm eg: DE solver
Data eg: DE parameters
Authors' Work
Equations
Figures
Tables
Publication

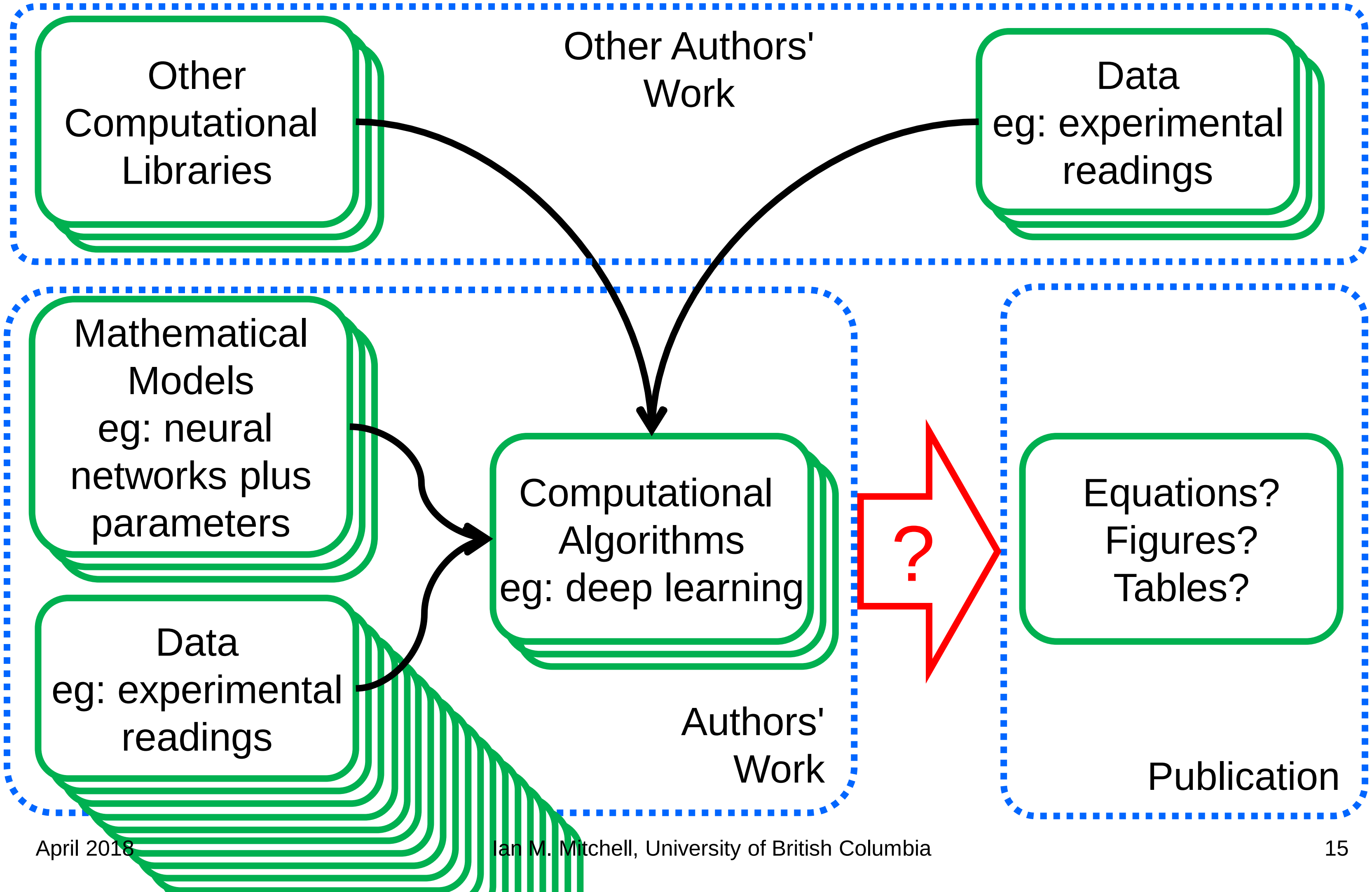
Computational Science & Engineering

- Simulation beyond the bounds of traditional theoretical or experiment analysis


Other Computational Algorithms
Other Authors' Work
Equations? Figures? Tables?
Other Publications?
Mathematical Models eg: PDEs
Data eg: material properties
Computational Algorithms
Authors' Work
Equations
Figures
Tables
Publication

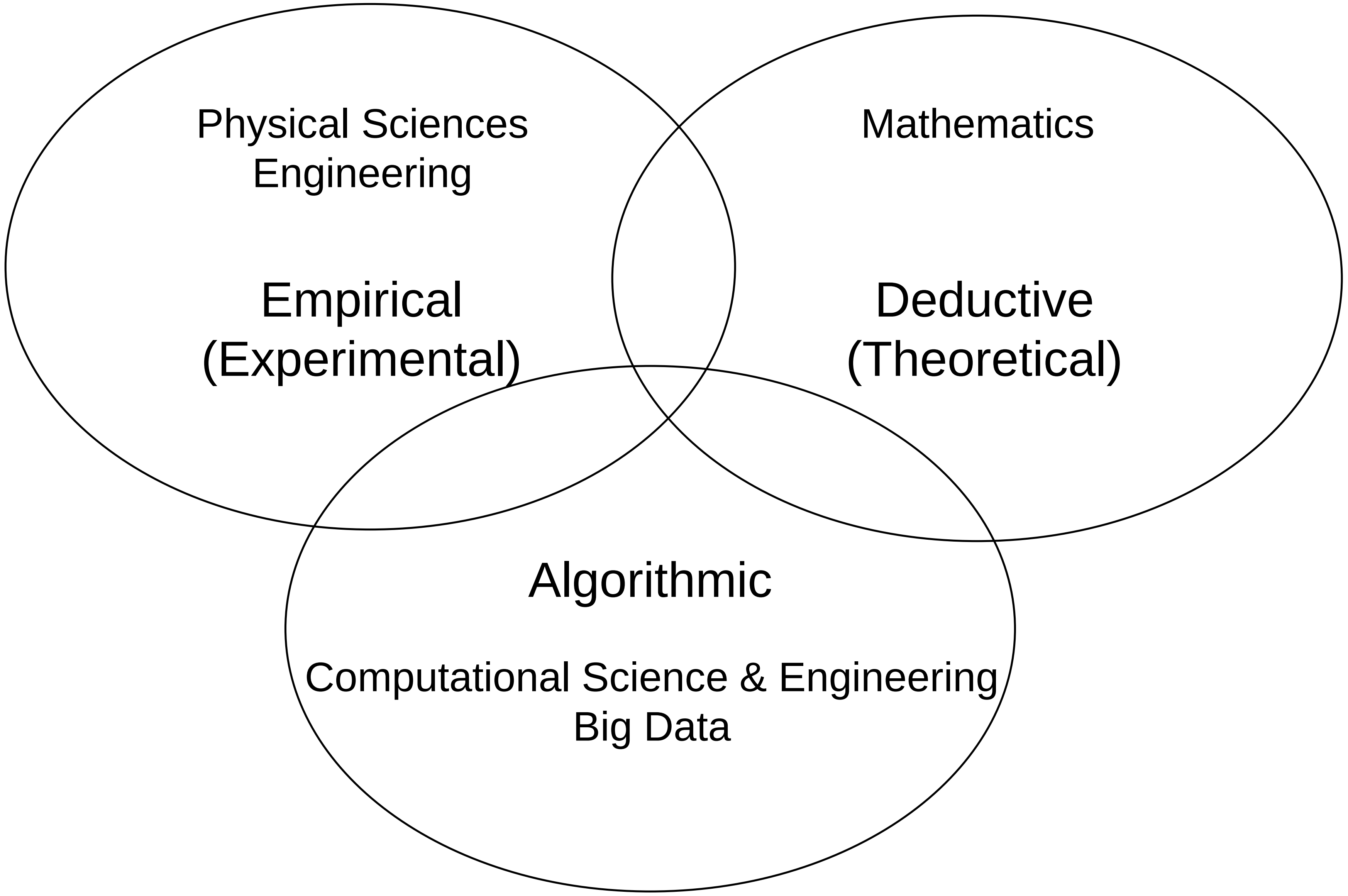
Big Data

- Algorithmically identifying and characterizing features, correlations, etc. from very large data sets



Other Computational Libraries
Other Authors' Work
Data eg: experimental readings
Mathematical Models eg: neural networks plus parameters
Computational Algorithms eg: deep learning
?
Equations? Figures? Tables?
Data eg: experimental readings
Authors' Work
Publication

Exploring the World



What's the Big Deal?

- We must face the ubiquity of error
  - Logic (eg: in proofs)
  - Resolution (eg: accuracy, precision, sensitivity)
  - Observation (eg: calibration, misalignment, noise)
  - Transcription (eg: recording / copying the data)
  - Modeling (eg: one vs two sided t-tests)
  - Tuning (eg: choosing parameters)
  - Implementation (eg: coding the algorithm)
  - Provenance (eg: getting the right data / software)
  - Execution (eg: different hardware / software platforms)
  - Analysis (eg: drawing conclusions)
- These sources of error have always existed
- The scientific method seeks to root out such error
  - Open publication of peer reviewed manuscripts
  - Expectation of reproducibility / repeatability

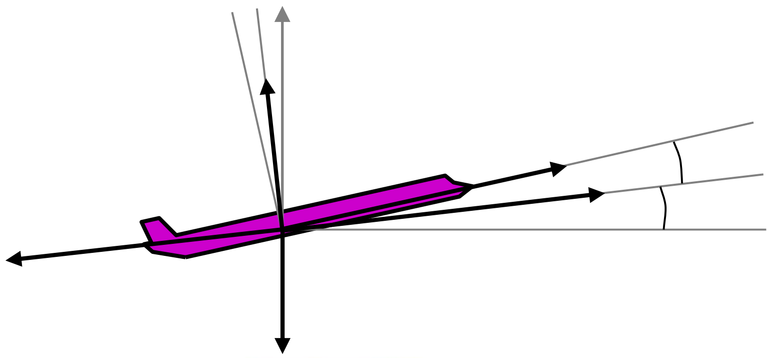
The Goal: Reproducible Research

- Our current approach evolved in an age when
  - All critical details could be recorded in a manuscript
  - A single person could reasonably vet them for correctness
- As automation grows, this is no longer true
  - We can work with data at scales, speeds and efficiencies far beyond manual human oversight
  - Even the details which drive the automation (eg: code and parameters) are often more than a peer reviewer can handle
- The *reproducible research* community seeks to overcome these challenges:

“[a]n 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.”
[Jon Claerbout, as quoted by Buckheit & Donoho, 1995]

Outline

- Motivation: Some lessons in Irreproducible Research
- Computation and Data Science
- Reproducible Research: Changing the Culture
- The HSCC Experience
- Code != Data



"Your recent Amazon purchases, Tweet score and location history makes you 23.5% welcome here."

Thierry Gregorius

Changing the Culture

- Special Issue on Reproducible Research
  - Computing in Science & Engineering (July/August 2012)
  - Articles drawn from workshop and community forum held at UBC in July 2011
  - Co-organized with Victoria Stodden & Randall J. LeVeque

Cover Image of *Computing in Science & Engineering*, 14:4

Three Themes from the Workshop

- Reproducibility of computational and data-driven science must be improved
- Challenges of encouraging reproducibility
  - How can we define, interpret, review, reduce barriers to, improve incentives for and provide examples of reproducible research?
- Development of tools & strategies to enhance and simplify reproducibility
  - Need to capture the computational environment, the provenance and the scientific narrative

Two Discussions at a Community Forum

- Journals & Publishers
  - Unclear whether computational and data science artifacts need traditional journal services (eg: managing peer review, formatting, dissemination, archiving)
  - Not clear to what extent code peer review is feasible
  - Policies can be used to encourage reproducibility, both directly (requiring code and data submission) and indirectly (eg: enforcing consistent citation)
- Funding Agencies
  - NSF data management plan requirements depend on research community
  - Short-term grant funding at odds with archival requirements
  - Include code and data sharing in CVs to provide credit
  - Computational scientists must become involved with discussions around open science

A Call to Arms

- "Next Steps" from the special issue:
  - All computational scientists should practice reproducibility, even if only privately and for the benefit of current and future research efforts
  - All interested computational scientists should tackle institutional and community challenges: train students, publish examples, request code during reviews, audit data management plans, etc.
  - All stakeholders must "consider code a vital part of the digitization of science"

World War I recruiting poster
US Library of Congress Collection

Setting the Default to Reproducible

- Workshop at ICERM in December 2012 produced three recommendations:
  1. It is important to promote a culture change that will integrate computational reproducibility into the research process.
  2. Journals, funding agencies, and employers should support this culture change.
  3. Reproducible research practices and the use of appropriate tools should be taught as standard operating procedure in relation to computational aspects of research.

Making Progress

- Some top journals and conferences are allowing / encouraging / requiring elements of reproducibility
  - Nature (April 2013): Key features of data collection and statistical analysis must be specified plus data deposition mandatory for some data types, strongly recommended for many others, availability of code must be specified
  - Science (Jan 2014): Key features of data collection must be specified
  - Computer science conferences (SIGMOD, OOPSLA, ESEC/FSE, SAS, ECOOP, CAV, HSCC) have begun to optionally accept and evaluate supplemental "artifacts"
  - ACM Digital Library supports linking of both reviewed and unreviewed supplemental material to papers
  - Software Carpentry project is teaching dozens of "bootcamps" on code and data management around the world

IEEE Workshop on Future of Research Curation and Research Reproducibility

- First workshop November 2016
  - Report published March 2017 (where is it?)
- Similar findings
  - Limited funding and pool of reviewers
  - Need to reform publication models and measurements of productivity to encourage reproducibility
  - Excessive levels of review can be counter-productive
  - Scientific progress and commercialization would accelerate
  - Tension between open access and commercial interests
  - Need to improve software quality and evolution
  - Definitions are still not standardized
- Call for bottom-up pilot projects

ACM Workshop on Reproducibility

- Third workshop December 2017
  - “Good, better, best” practices guide under revision
- Targets five stakeholder groups
  - Authors: Increase visibility and impact of research through reproducibility
  - Reviewers: Assist authors to attain reproducibility
  - Editors & chairs: Establish consistent artifact evaluation as part of normal editorial workflow
  - Publishers: Support artifact review and publication with policies, guidelines, tools and scalable infrastructure
  - Repositories, vendors and infrastructure providers: Integrate artifact development, submission, deposit, registration, review (and evolution?) into platforms
- Provides suggested actions for implementing and improving practices

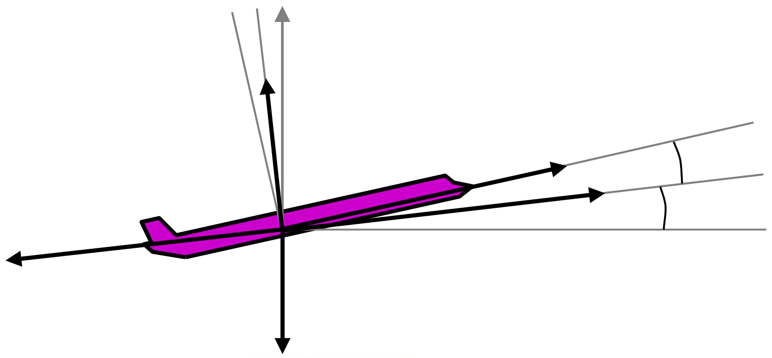
ACM Artifact Review and Badging

- Announced late 2016
- Defines terminology:
  - Repeatable: Same team, same experimental setup
  - Replicable: Different team, same experimental setup
  - Reproducible: Different team, different experimental setup
- Badges for several orthogonal concepts
  - Artifacts evaluated: functional or reusable
  - Artifacts available
  - Results validated: replicated or reproduced
- Review process left for community definition
- Accompanying effort to add badges to ACM DL
  - Allow for searching and linking
  - Badges may be awarded post-publication

Artifacts Evaluated
acm
Functional
Artifacts Evaluated
acm
Reusable
Artifacts Available
acm
Results Replicated
acm
Results Reproduced
acm

Outline

- Motivation: Some lessons in Irreproducible Research
- Computation and Data Science
- Reproducible Research: Changing the Culture
- The HSCC Experience
- Code != Data



"Your recent Amazon purchases, Tweet score and location history makes you 23.5% welcome here."

Thierry Gregorius

HSCC Context

- Sponsored by SIGBED
- Part of Cyber-Physical Systems Week
  - with RTAS, IPSN, ICCPS
- Publishes computational results
  - Most (appear to) require standard HW/SW environments
- Historically, few papers provided software

	2011	2012	2013
Full papers	31	28	30
Significant computation	21	20	20
Stated Matlab	6	10	7
Unstated (mostly Matlab)	8	6	6
HW / data (difficult to repeat)	2	1	2
Links provided in paper	3	4	3

The Goal: What's in it for me?

- Raise the profile of papers containing repeatable computational results by highlighting them at the conference and online
- Raise the profile of HSCC as a whole by making it easier to build upon the published results
- Provide authors an incentive to adopt best-practices for code and data management that are known to improve the quality and extendability of computational results
- Provide authors an opportunity to receive feedback from independent reviewers about whether their computational results can be repeated
- Be able to recreate a previous student's (or your own) results two years later

Built on Established Examples

- SIGMOD 2008 “experimental reproducibility effort”
- Artifact evalution
  - ESEC/FSE 2011 & 2013,
  - SAS 2013,
  - ECOOP 2013 & 2014
- OOPSLA 2013 “Artifact Evaluation Artifact

Followed Common Procedures

- No perturbation to traditional paper review and acceptance process
- Optional repeatability evaluation for accepted papers
- Repeatability evaluation committee of postdocs and senior grad students
- Single-blind reviews
  - Chair manages inevitable communication about installation issues
- Submitted material is confidential
  - Authors encouraged but not required to release software
- Non-competitive, threshold-based acceptance
  - Only successful submissions are publicized

Evaluation Rubric

- Provide authors and reviewers guidance
  - Concrete and objective descriptions of insufficient, sufficient and aspirational levels in each criterion
- Criteria
  - Coverage: Fraction of computational figures / tables that can be repeated
  - Instructions: Details of how to repeat and possibly extend results
  - Quality: Documentation and testing
- Each criterion scored 0–4
- Repeatable if
  - No score of 0
  - Average score of 2

Coverage

- Consider only figures / tables whose data is generated computationally
  - “Repeatable” if same results can be generated
  - “Extensible” if variations can be run
- Scoring categories
  - None repeatable (0 / missing)
  - Some repeatable (1 / falls below expectations)
  - Most repeatable (2 / meets expectations)
  - All repeatable / most extensible (3 / exceeds expectations)
  - All extensible (4 / significantly exceeds expectations)

Instructions

- Authors submit a document explaining how to install, run and possibly extend the software
- Scoring categories
  - None (0 / missing)
  - Rudimentary (1 / falls below expectations): script or command but nothing else
  - Complete (2 / meets expectations): for every computational element a procedure is described
  - Comprehensive (3 / exceeds expectations): for every computational element a single command recreates that element almost exactly
  - Outstanding (4 / significantly exceeds expectations): description of design decisions, major components / modules, how to modify / extend

Quality

- Can the software be trusted and deciphered
- Scoring categories
  - None (0 / missing): No evidence of documentation or testing
  - Rudimentary documentation (1 / falls below expectations): purpose of almost all files explained
  - Comprehensive documentation (2 / meets expectations): within source files classes, methods, functions, attributes and variables given clear names and/or documentation; within data files format and structure of data is documented
  - Documentation and rudimentary testing (3 / exceeds expectations): identified test cases with known solutions validate some components
  - Documentation and comprehensive testing (4 / significantly exceeds expectations): identified unit and system level testing of a significant fraction of the code

Results

	2014	2016	2017	2018
Full papers (not incl. tool papers)	29	28	29	25
Submitted RPs (incl. tool papers)	5	18	14	18
Successful RPs (incl. tool papers)	5	14	13	14

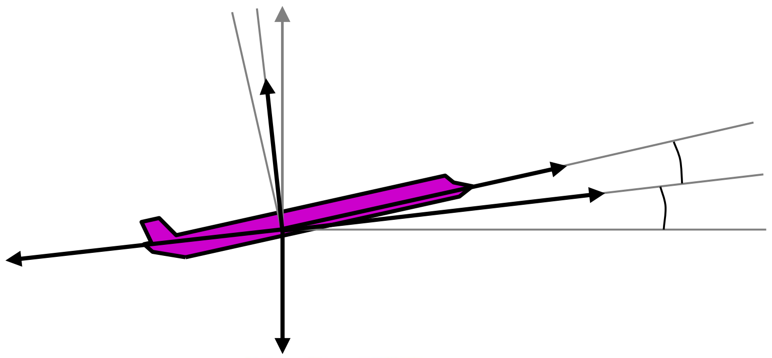
- 2016: Repeatability prize introduced
- 2017: Sergiy Bogomolov (The Australian National University) assumed RE chair
- 2018: Tool papers required to pass RE at initial submission

Challenges

- Installation headaches
- Conflicting reviews
- Increasing use of specialized hardware / software environments
- Timing: RE submission, final paper submission, publisher deadlines (and term holidays)
- Long-term access to released software
- Coordination with ACM policies
- Broader uptake in CPS and SIGBED community

Outline

- Motivation: Some lessons in Irreproducible Research
- Computation and Data Science
- Reproducible Research: Changing the Culture
- The HSCC Experience
- Code != Data



"Your recent Amazon purchases, Tweet score and location history makes you 23.5% welcome here."

Thierry Gregorius

Code vs Data in Reproducible Research

Treating code as a form of supplementary data ignores important features of code as an information storage artifact

- Relating to the practice of science
- Relating to the management of code
- Relating to the interaction of code with society

Code and the Practice of Science

- Code is a mechanism for generating data and hence a source of error
  - Digital data formats introduce no error (except when they do)
  - Errors are not smooth: The size of the mistake has little relationship to the size of the resultant error
  - Errors are not well characterized

- Scientists at all levels are not trained to manage code (and its errors)
  - At UBC: Physical science undergraduates take two courses in programming, life science undergraduates take none
  - Little incentive for giving or receiving instruction

Management of Code

- Almost always evolving
  - Bug fixes, refactoring, new features
  - Application programming interface (API) attempts to hide internal details from users
- Inverted data to metadata ratio
  - The code written to support a particular analysis may be short, but it draws upon libraries, compilers, operating systems, drivers, etc.
- Readable by both machine and people
- Many practices and tools have been developed to manage code
  - Version control systems and ecosystems (eg: github)
  - Virtual machines
  - Lints, automated testing, debuggers, profilers, ...
  - Extensive opportunities for training

Software Carpentry

- www.software-carpentry.org
- Dedicated to teaching basic software and data management skills to scientists
- Bootcamps: Intensive two-day, hands-on session covers:
  - Programming basics (Python or R)
  - Version control (git or subversion)
  - Unit testing
  - Using shell to automate tasks
  - Optional topics: databases & SQL, regular expressions, debugging, numerical packages, ...
- Screencasts covering many more topics are available from the website

software carpentry

Code and Society

- Privacy is not an issue

- Intellectual property rights are a huge issue
  - Afforded strong copyright protection, possibly also patents
  - Most companies and some universities restrict researchers' ability to release code
  - Proprietary platforms / libraries restrict ability to capture metadata and reproduce results
  - Broad legal consensus that open code should be treated differently than open data or open creative works
  - Huge open source community provides examples of and demonstrates benefits of open code, although size is critical to success

Conclusions

- Increasing dependence on poorly shared code and data threatens the credibility of research throughout the sciences
- Reproducible research is a broad and diffuse effort to counteract this threat
  - Overlaps with but is distinct from open access, open science, open source, etc.
  - Many ~~exploratory~~ efforts underway to change the culture
  - Expectations and aspirational goals will evolve with community pilot projects, technology and generational turnover
- The "big data revolution" cannot ignore the code
  - Automation is critical to managing the data glut
  - Code can and must be managed differently than other types of data

Reproducible Research Citations

- Reproducible research
  - Stodden, Leisch & Peng (eds.), *Implementing Reproducible Research*, CRC Press (2014)
  - Stodden, Borwein & Bailey, "Setting the Default to Reproducible in Computational Science Research" in *SIAM News*, June 2013
  - Leveque, "Top Ten Reasons to Not Share Your Code (and why you should anyway)" in *SIAM News*, April 2013
  - LeVeque, Mitchell & Stodden, "Reproducible Research for Scientific Computing: Tools and Strategies for Changing the Culture" in *Computing in Science and Engineering* 14(4): 13–17 (2012)
  - Stodden, "Enabling Reproducible Research: Licensing for Scientific Innovation" in *Int. J. Communications Law & Policy* 13 (winter 2009)
- ACM badging: Boisvert, "Incentivizing Reproducibility" in Comm. ACM 59(10): 10 (Oct 2016).

Reproducible Research: Failures, Successes, Challenges and (Re)Setting the Bar

For more information contact

Ian M. Mitchell

Department of Computer Science

The University of British Columbia

`mitchell@cs.ubc.ca`

`http://www.cs.ubc.ca/~mitchell`

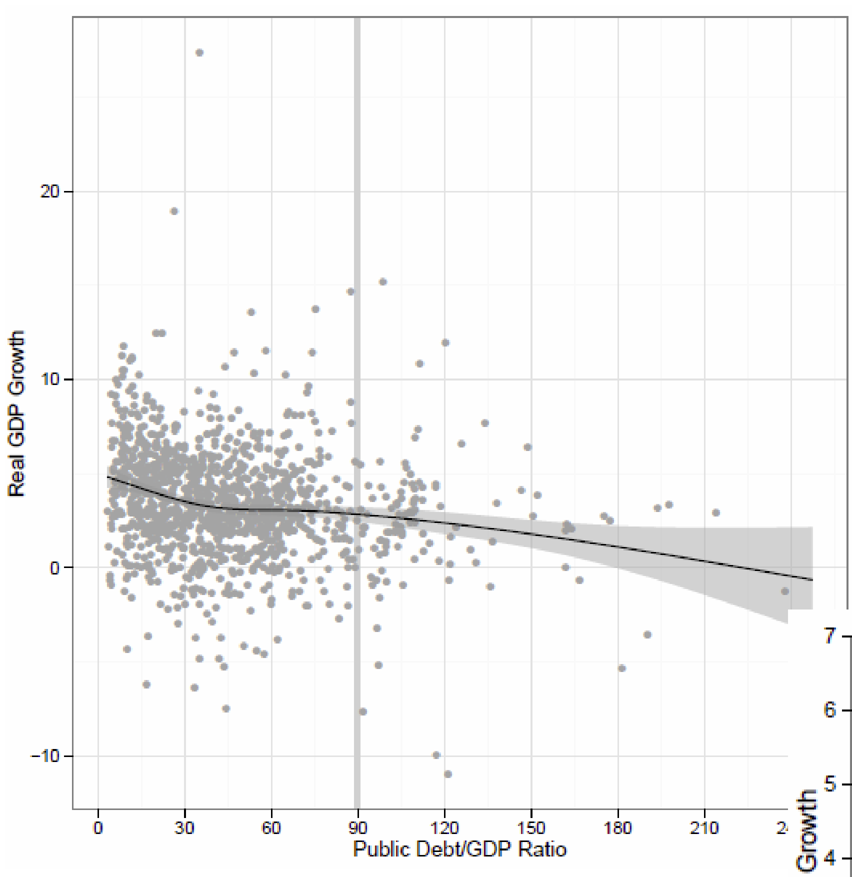
Some Suggestions for Doing It Better

- Use a (modern) version control system
  - Online repositories (eg: bitbucket, github, google code, sourceforge) include wikis and issue trackers
- Document in the data (and code is data)
  - You will forget how and why you did things
  - Files and directories will get separated and lost
- Write tests first and run them often
  - Bugs are inevitable and “static” code isn’t
- If you do it twice, automate it
  - Computers are better at repetition, you can automate a person with a checklist, and automation is documentation
- Look at the code together
  - Code review and pair programming lead to demonstrable improvements in code quality
- Plan to release your code
- **Improve your process gradually but continually**
  - Every little bit helps

Citations and Links

- Ideas from software engineering
  - Wilson et. al., "Best Practices for Scientific Computing" PLoS Biol 12(1): e1001745. doi:10.1371/journal.pbio.1001745
  - Heroux & Willenbring, "Barely Sufficient Software Engineering: 10 Practices to Improve Your CSE Software" in *ICSE Workshop on Software Engineering for Computational Science & Engineering*, pp.15-21 (2009)
  - Sink, *Version Control by Example*, 2011
- Testing differential equation codes
  - Oberkampf & Roy, *Verification and Validation in Scientific Computing*, 2010
  - Roy, "Review of Code and Solution Verification Procedures for Computational Simulation", *J. Comp. Physics* 205:131-156 (2005)
  - Knupp & Salari, *Verification of Computing Codes in Computational Science and Engineering*, 2003

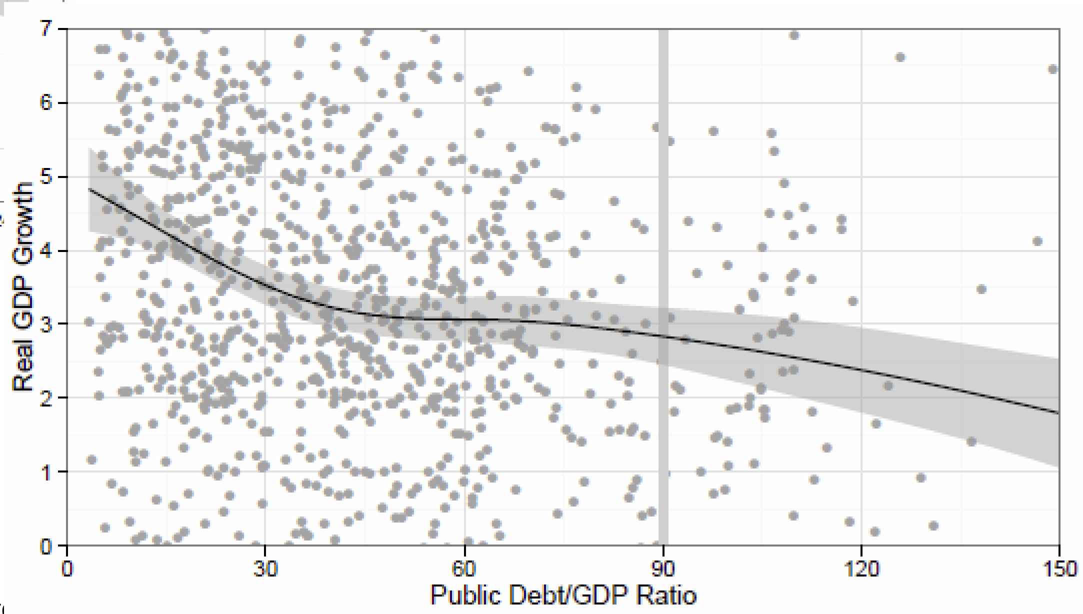
Lies, Damn Lies & Statistics



Full Advanced Economy Postwar Data Set

Images from:
Herndon, Ash & Pollin, "Does High Public Debt Consistently Stifle Economic Growth? A Critique of Reinhart and Rogoff" Political Economy Research Institute Working Paper (April 2013)

Zoomed View


Real GDP Growth
Public Debt/GDP Ratio