Reproducibility Minisymposium

K. Jarrod Millman
University of California, Berkeley

Vincent J. Carey
Harvard University

SIAM Conference on Computational Science and Engineering (CSE13)

Reproducibility and computationally intensive, data-driven research
February 28, 2011 ♦ 14:00-16:00 (Part I) ♦ 16:30-18:00 (Part II)

Outline

- Why is there a problem?
- How can we tackle the problem?
- What we are going to talk about today?

Computational Lifecycle

- Individual exploration
- Collaboration
- Production-scale execution
- Publication
- Education

What Percent of Publications are Reproducible?

“Take nobody’s word for it”

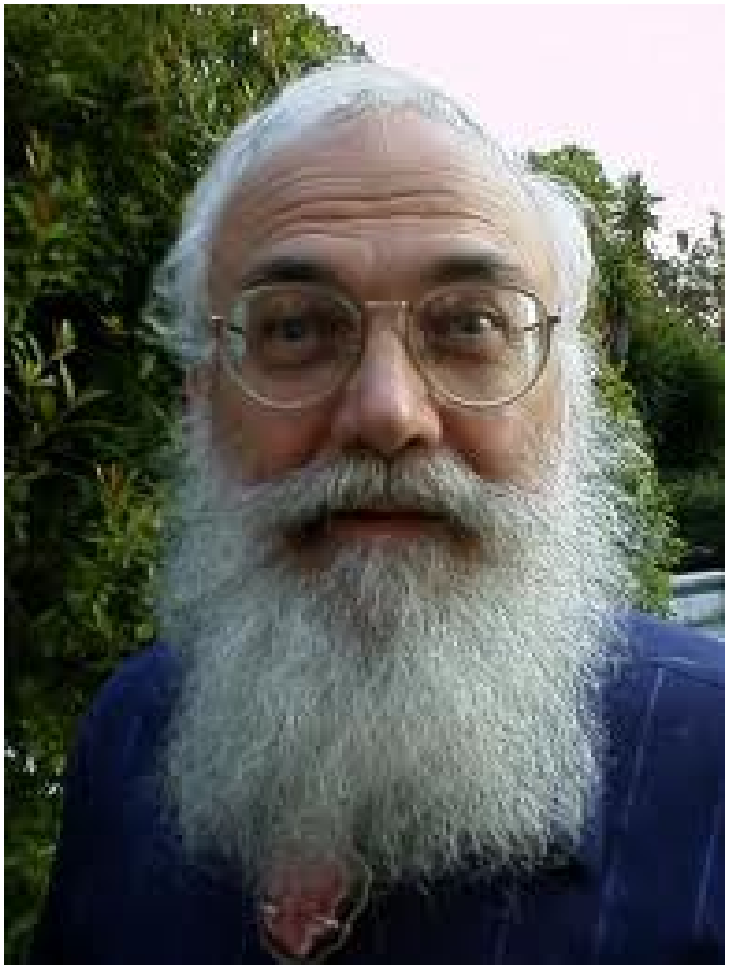
Early computers

More computing, more problems

The major cause of the software crisis is that the machines have become several orders of magnitude more powerful! To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have ***gigantic computers****, programming has become an* ***equally gigantic problem****.*

— Edsger Dijkstra, The Humble Programmer (1972)

Jon Clarebout



1995 - WaveLab

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

— Jonathan Buckheit and David Donoho, WaveLab and Reproducible Research (1995)

SIAM Conference on Computational Science & Engineering

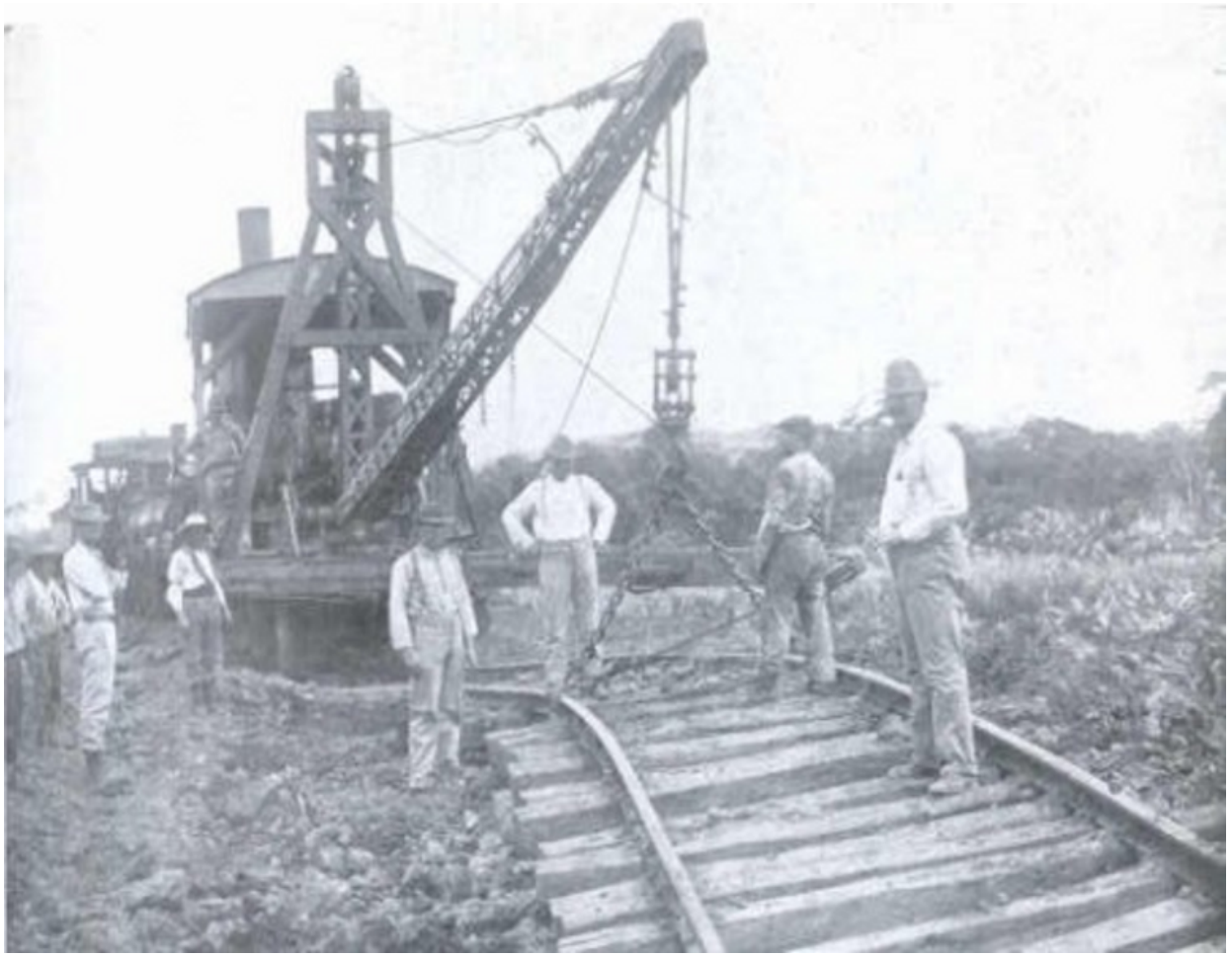
2011

- MS148/MS155 Verifiable, reproducible research and computational science

2013

- MS180 Impacts of open access and reproducibility on verification and validation
- MS205/MS224 Reproducibility and computationally intensive, data-driven research

Make the dirt fly!



No Silverbullet

Changing ...

- how we **conduct** our computational work
- how we **train** scientists
- how we **communicate** with each other
- how we **recognize** scientific contributions

Best Practices

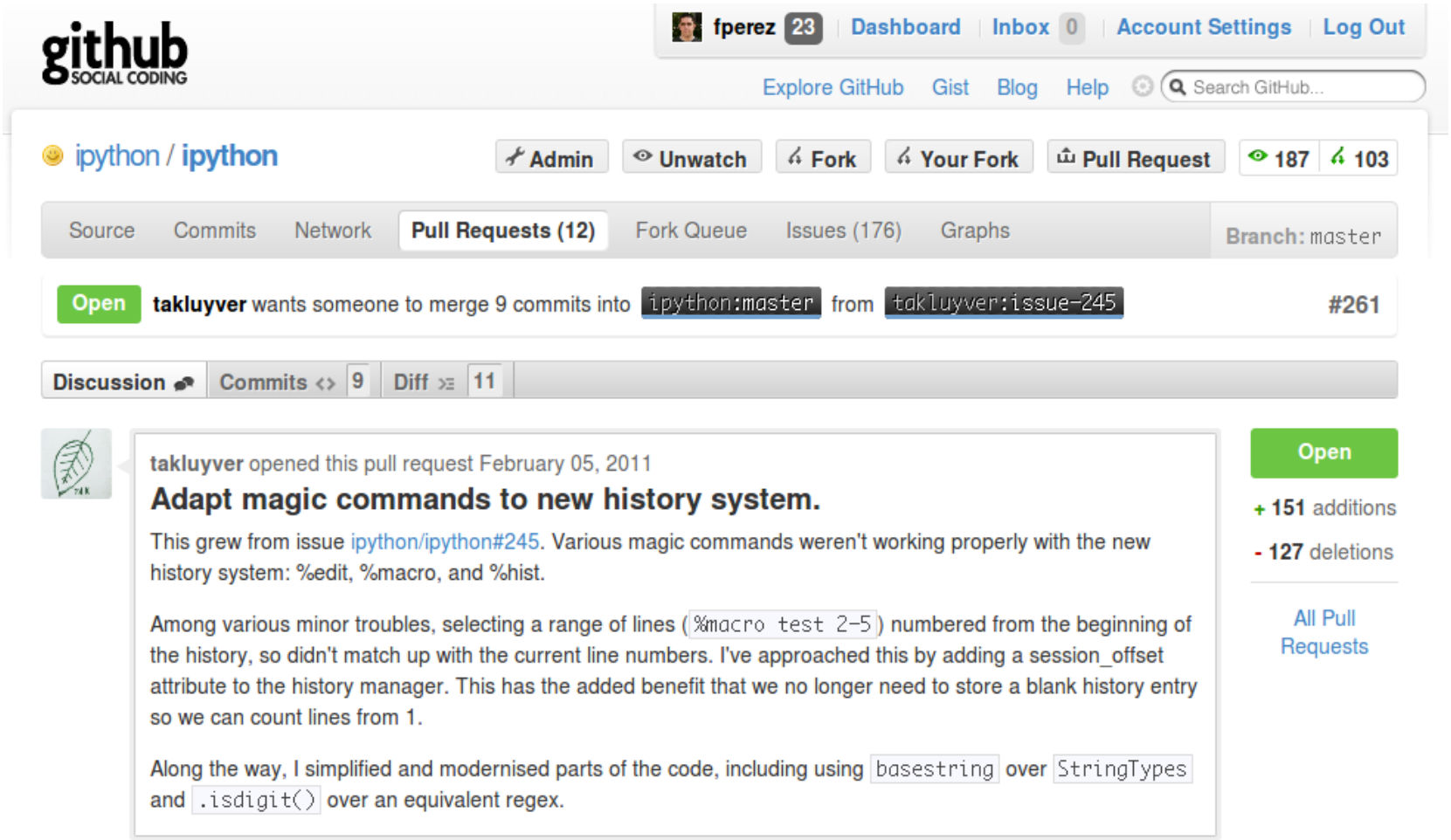
Best Practices for Scientific Computing

Greg Wilson [*], D.A. Aruliah [†], C. Titus Brown [‡], Neil P. Chue Hong [§], Matt Davis [¶], Richard T. Guy [‖], Steven H.D. Haddock [**], Katy Huff [††], Ian M. Mitchell [‡‡], Mark D. Plumbley [§§], Ben Waugh [¶¶], Ethan P. White [***], Paul Wilson [†††]

Scientists spend an increasing amount of time building and using software. However, most scientists are never taught how to do this efficiently. As a result, many are unaware of tools and practices that would allow them to write more reliable and maintainable code with less effort. We describe a set of best practices for scientific software development that have solid foundations in research and experience, and that improve scientists' productivity and the reliability of their software.

arXiv:1210.0530v3 [cs.MS] 29 Nov 2012

Version Control



Community of Practice

How to Scale a Code in the Human Dimension

Matthew J. Turk (matthewturk@gmail.com)
Columbia University

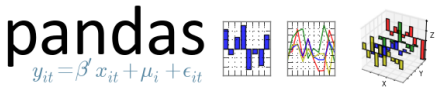
Abstract: As scientists' needs for computational techniques and tools grow, they cease to be supportable by software developed in isolation. In many cases, these needs are being met by communities of practice, where software is developed by domain scientists to reach pragmatic goals and satisfy distinct and enumerable scientific goals. We present techniques that have been successful in growing and engaging communities of practice, specifically in the yt and Enzo communities.

arXiv:1301.7064v1 [astro-ph.IM] 29 Jan 2013

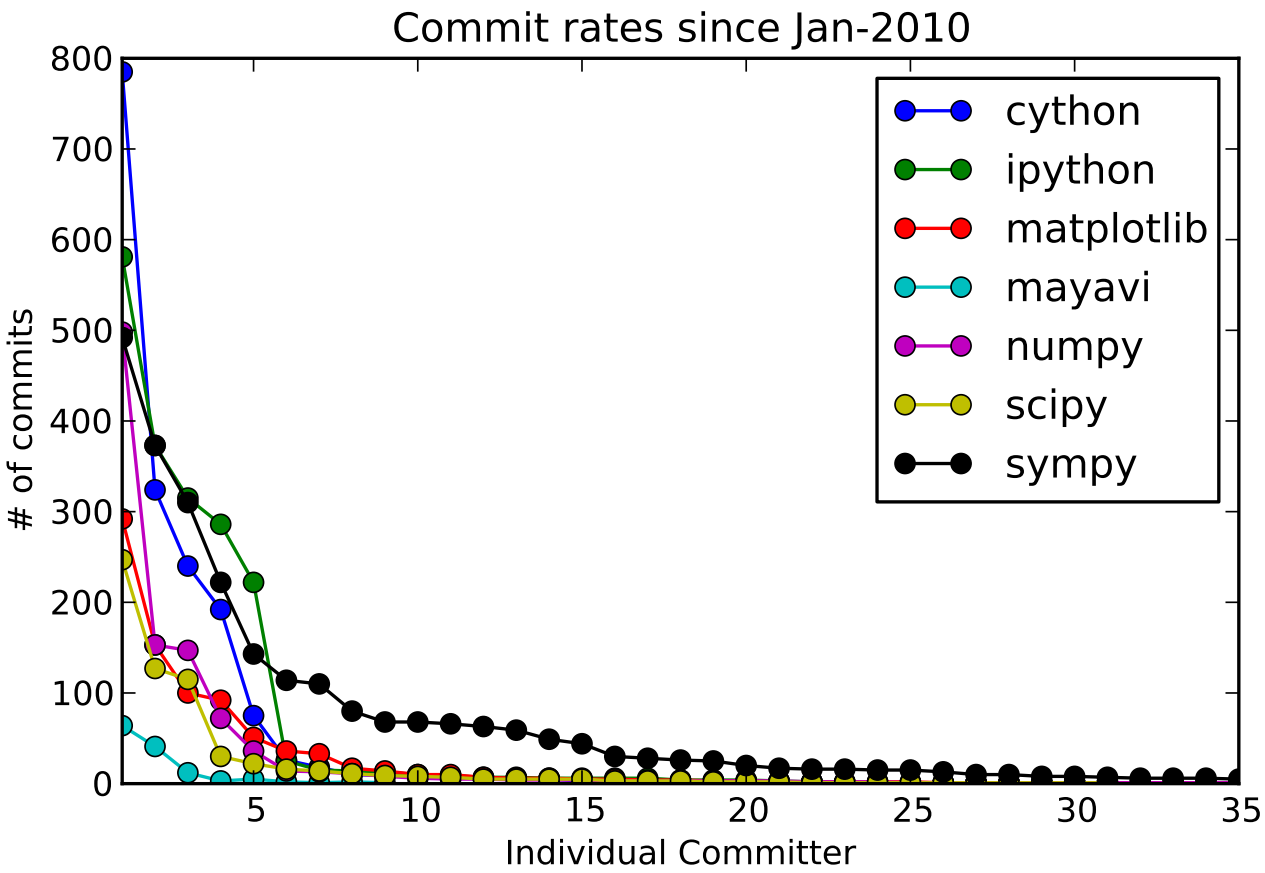
Open source ecosystem

IPython

SymPy



Where is Everyone?



Topics

- Education
- Publication
- Forensics & scientific integrity
- Provenance tracking & literate programming

Questions