CODECHECK: An open-science initiative to facilitate sharing of computer programs and results presented in scientific publications.

Stephen J Eglen	Cambridge Computational Biology Institute
https://sje30.github.io	University of Cambridge
sje30@cam.ac.uk	@StephenEglen
Daniel Nüst	Institute for Geoinformatics
https://nordholmen.net	University of Münster
daniel.nuest@uni-muenster.de	@nordholmen

Declarations

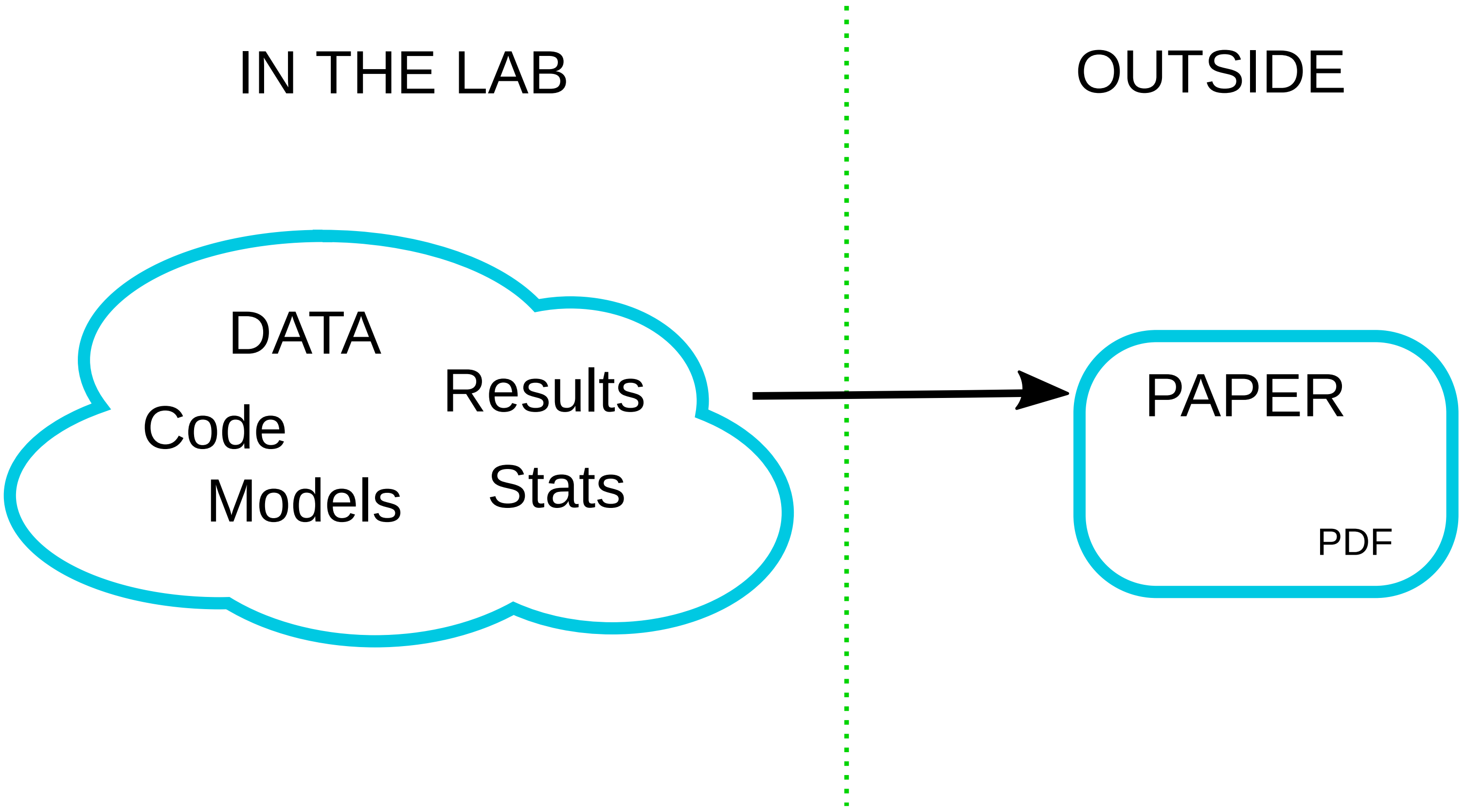
Affiliate editor of *bioRxiv*. Senior editor of *Scientific Data*.

Acknowledgements

Mozilla mini science grant, UK Software Sustainability Institute.

Editors @ *Gigascience*, *eLife*, *Scientific Data*.

Science as an inverse problem



Why share code?

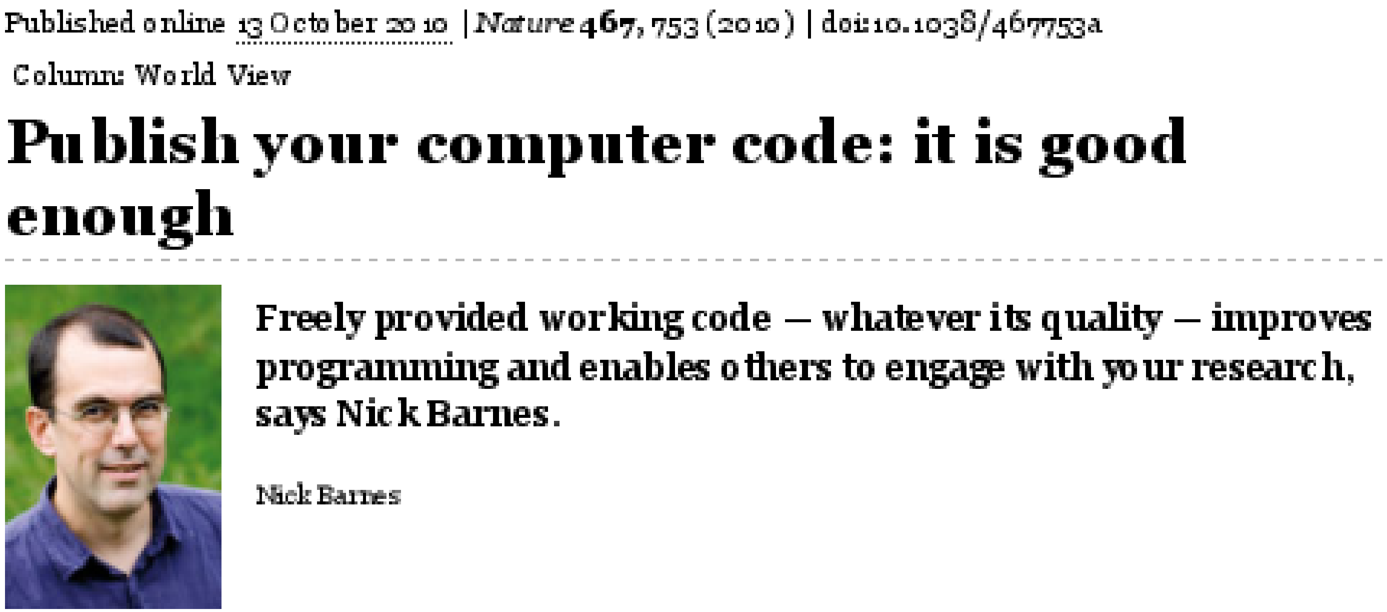
> An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures.

Buckheit & Donoho (1995)

> The problem is that most modern science is so complicated, and most journal articles so brief, it’s impossible for the article to include details of many important methods and decisions made by the researcher

Marwick (2015)

Approaches to code sharing



Published online 13 October 2010 | *Nature* 467, 753 (2010) | doi:10.1038/467753a

Column: World View

Publish your computer code: it is good enough

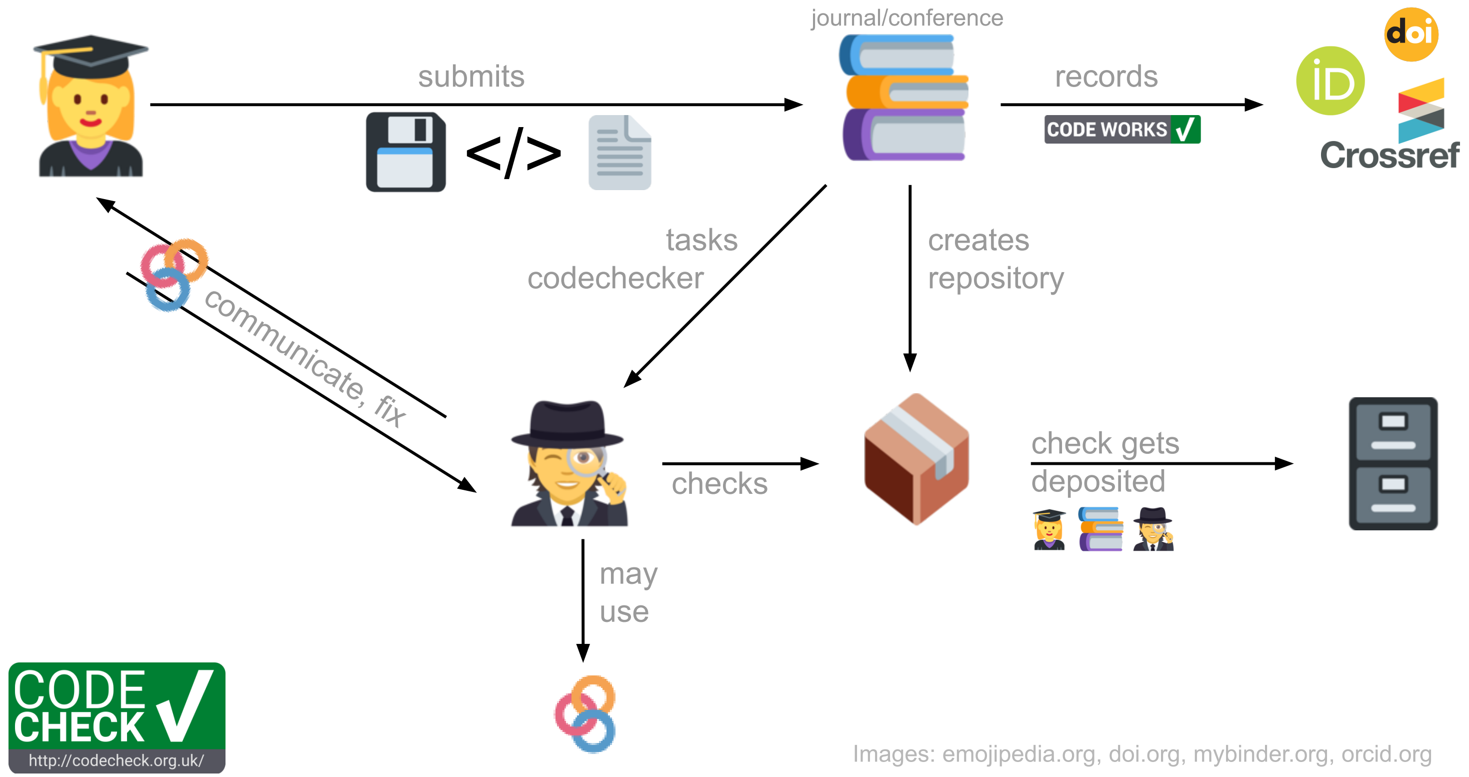
Freely provided working code — whatever its quality — improves programming and enables others to engage with your research, says Nick Barnes.

Nick Barnes

- Informal 'code buddy' system
- Community-led *research compedia*.
- Code Ocean (Nature trial)
- Certify reproducibility with confidential data (CASCAD) (Pérignon et al 2019)

The CODECHECK philosophy

- Systems like Code Ocean set the bar high by "making code reproducible *forever* for *everyone*".
- CODECHECK simply asks "was the code reproducible *once* for *someone* else?"
- We check the code runs and generates the expected number of output files.
- The contents of those output files are not checked, but are available for others to see.
- The validity of the code is *not* checked.



Four princicples underlying CODECHECK

1. CODECHECKERS are **humans** and communication is key.
2. CODECHECKERS **record** but don’t investigate or fix.
3. **Credit** is given to CODECHECKERS.
4. Workflows are scripted, **auditable**, and they work (or *worked once*).

Who does the work?

1. **AUTHOR** provides code/data and instructions on how to run.
2. **CODECHECKER** runs code and writes certificate.
3. **PUBLISHER** oversees process, helps depositing artifacts, and persistently publishes certificate.

Who benefits?

1. **AUTHOR** gets early check that "code works"; gets snapshot of code archived and increased trust in stability of results.
2. **CODECHECKER** gets insight in latest research and methods, credit from community, and citable object.
3. **PUBLISHER** Gets citable certificate with code/data bundle to share and increases reputation of published articles.
4. **PEER REVIEWERS** can see certificate rather than check code themselves.
5. **READER** Can check certificate and build upon work immediately.

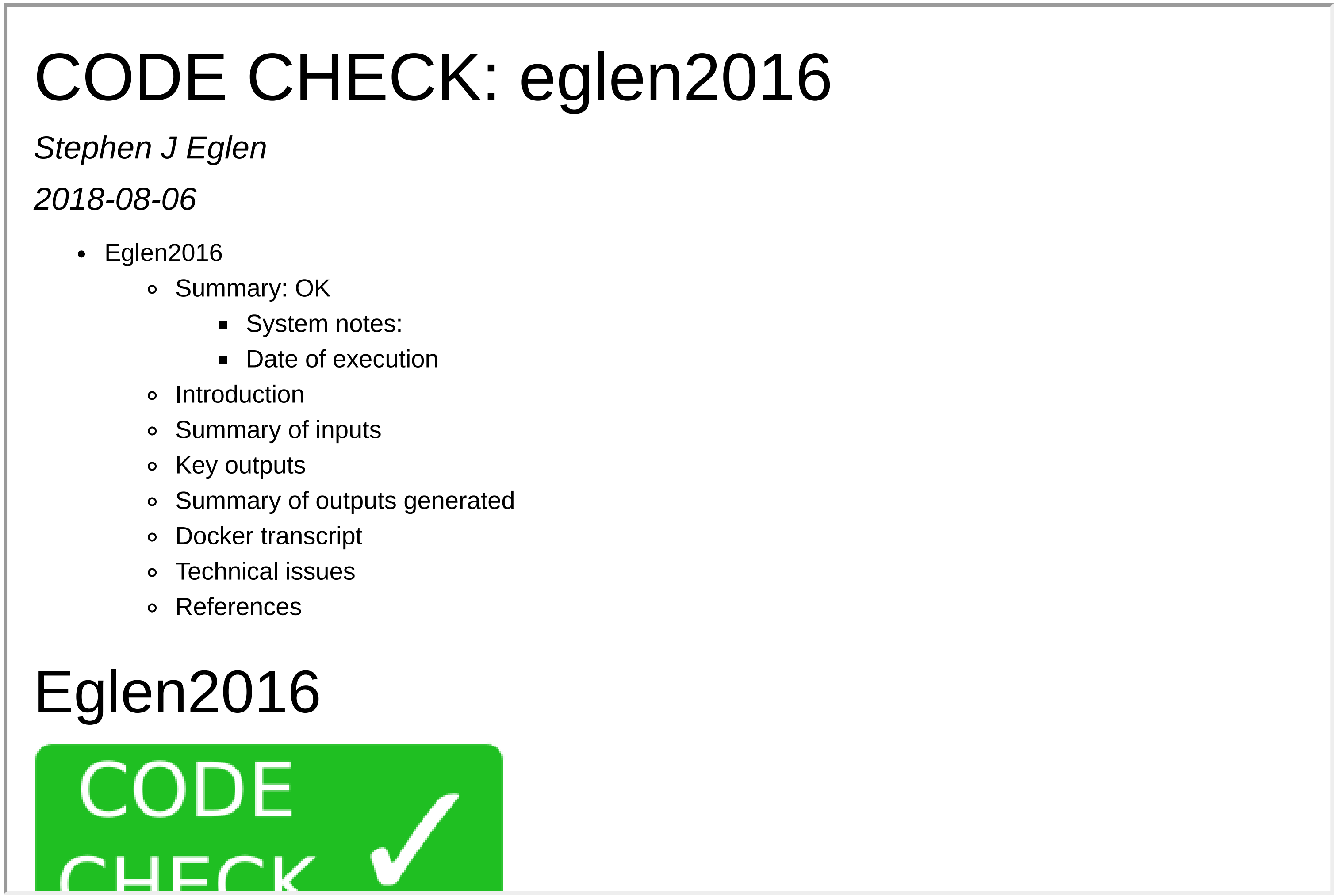
Technology

We currently promote the following environment:

1. Author provides code + data. We need LICENSE, MANIFEST, and README or Makefile.
2. Codechecker repository on *GitHub* stores code and data.
3. Use software (*Docker*, *Python virtualenvs*, *R renv*) for fresh environments.
4. *Makefile* to provide human- and machine-readable description of workflow.
5. *Binder* for interactively sharing environments between author and Codechecker.
6. *Rmarkdown* currently used to author certificates.
7. Certificates and snapshot of data/code/outputs deposited on *Zenodo* by Codechecker.
8. Currently reliant on publishers for deposition of metadata to relevant sites (ORCID, CrossRef, Publons).

Example certificate (online)

https://sje30.github.io/codecheck/eglen2016/eglen2016-crc.html



CODE CHECK: eglen2016

Stephen J Eglen

2018-08-06

- Eglen2016
  - Summary: OK
    - System notes:
    - Date of execution
  - Introduction
  - Summary of inputs
  - Key outputs
  - Summary of outputs generated
  - Docker transcript
  - Technical issues
  - References

Eglen2016

Limitations

1. CODECHECKER time is valuable, so needs credit.
2. Very easy to cheat the system, but who cares?
3. Author's code/data must be freely available.
4. Deliberately low threshold for gaining a certificate.
5. High-performance compute needs consideration.
6. Cannot (yet) support all thinkable/existing workflows and languages.
7. Needs publisher involvement.

Next steps

1. How to wrap up meta data of certificate and artifacts such that they are useful and reusable.
2. Embedding into journal workflows.
3. Training a community of codecheckers.
4. Generate portfolio of examples.

For more information please see: http://codecheck.org.uk