Cultivating research software

http://dx.doi.org/10.6084/m9.figshare.3840303

20th September 2016, Rydberg Seminars, Lund
Neil Chue Hong (@npch), Software Sustainability Institute
ORCID: 0000-0002-8876-7606 | N.ChueHong@software.ac.uk
Modern research is impossible without software

From thrown-together scripts, through an abundance of complex spreadsheets, to the millions of lines of code behind large-scale infrastructure, there are few areas where software does not play a fundamental part in research.
Computational Chemistry - CASTEP

From the first implementation of a DFT algorithm to a completely new code to community supported software

- Individual
- Group
- Consortium
- W/ industry
- Community
- Active

Software advances < hardware speedup
LOTAR: storing aeronautical models

- Time between CAD Versions: 6 months
- Life of CAD System: 10 years
- Life of Product: 70 years +

Production | Services | Spares | Legal Liability
---|---|---|---
CAD Obsolete | CAD Forgotten

10 years | 20 | 30 | 40 | 50 | 60

Image courtesy PDES Inc
Slide from Sean Barker, BAE SYSTEMS, DPC Designed to Last
All software is “legacy code”

• “Many of us have tried to discover ways to prevent code from becoming legacy. But ... prevention is imperfect. Even the most disciplined development team, knowing the best principles, using the best patterns, and following the best practices will create messes from time to time. The rot still accumulates. It’s not enough to prevent the rot – you have to be able to reverse it.”
The Software Sustainability Institute enables researchers to exploit software effectively

**Increasing skills**

“We were aiming to calculate the binding affinity of a drug to a protein. The Institute taught me to become a software manager, I can’t speak highly enough of them”

*Christopher Woods*  
*Chemistry researcher*

**Improving software**

“The code used to crash whenever it encountered ‘bad’ data. Now most of the users aren’t even aware of the change but can get on with their work without interruption”

*Tomasz Zielinski*  
*Systems Biology developer*

**Enabling researchers**

“I’d never have been able to use [HPC] on my own, but now we have software that can run on it. Biomass could be used in co-firing coal stations, or in making bio-oil. So the outputs of our model – potential crop yield - feed into that.”

*Prof. Gail Taylor*  
*Biofuels researcher*
Software sustainability is the capacity of the software to endure. In other words, sustainability means that the software will continue to be available in the future, on new platforms, meeting new needs.
BETTER SOFTWARE BETTER RESEARCH
www.software.ac.uk
The UK research community relies on software

Do you use research software?
- Yes: 92%
- No: 8%

What would happen to your research without software?
- Would be impossible: 68%
- Possible, but difficult: 21%
- No effect: 11%
- Develop their own software: 56%
- Have no formal software training: 71%

Survey of researchers from 15 Russell Group universities conducted by SSI between August - October 2014. 406 respondents covering representative range of funders, discipline and seniority.
The cost of UK research that relies on software

£840m

Investment in 2013-2014 financial year, an amount that has risen by 3% on average over last four years

30%

Of total research investment has been spent on research which relies on software over the last four financial years

Analysis of data from 49,650 grant titles and abstracts published on Gateway to Research covering 2010-2014.
The UK research community doesn’t have the skills

56% Of UK researchers develop their own research software

71% Of UK researchers have had no formal software development training

140,000 UK researchers are relying on their own coding skills

Survey of researchers from 15 Russell Group universities conducted by SSI between August - October 2014. 406 respondents covering representative range of funders, discipline and seniority.
Reproducible research drives the need for better software

Limited research funding drives the need for efficient software development
Many landmark findings in preclinical oncology research are not reproducible, in part because of inadequate cell lines and animal models.

Raise standards for preclinical cancer research.

47 out of 53 “landmark” publications could not be replicated.

56% of analyses could not be repeated, of which 30% were because of software issues. 50% did not state software version, 39% did not provide raw data. Only 11% could be reproduced satisfactorily.
Mainstream attention has focused on problems

Software engineering

Of more than academic interest

Professors’ unprofessional programs have created a new profession

Mar 26th 2016 | Oxford | From the print edition

SOME programmers call it “spaghetti code”. Though error-strewn, it works—most of the time. It is likely to have been written in an out-

C:\\lab\\
fixtures -o
data.exe

...

...ERROR

...why scientific programming does not compute

Nature, doi:10.1038/467775a
“You must publish your software because it’s important for research”

When Data Is Not Enough

BY DON MONROE
COMMUNICATIONS OF THE ACM, Vol. 10.1145/2833318

“Several correspondents have requested that we share the raw data, and while we would welcome the possibility of doing this, for governance reasons it is simply not possible. However, these data are available from the HSCIC to those with adequate provision for data management and security. Large datasets such as the one used for our analyses (\approx 150m) observations require specialist computer equipment and software to setup and undertake the analyses.”

This raises an important issue that is often neglected in discussions on transparency in science: the need for analytic code to be shared, as well as underlying data. While pseudonymised individual patient data often poses a reidentification risk, this is rarely the case with the analysis programs written in Stata, R, or other packages to analyse that data (excepting, at worst, some individual manual commands to correct errors in individual patients’ data).

Sharing such code can be hugely informative for those wishing to understand in detail the analytic choices made. It also facilitates sensitivity analyses, to interrogate the impact of large numbers of specific individual analytic choices (which can often be discretionary) on the overall result. To take this very concrete example: were Freemantle et al. to share their code for this paper, then those who have access to the same underlying data through their own licenses could review the impact of the original team’s analytic choices very rapidly.

The BMJ should require all papers to share their analytic code.
Of 601 papers in ACM Computer Science journals and proceedings, only 85 provided a link to software. For 176 the software could not be obtained.

Collberg, Proebsting, Warren, University of Arizona TR 14-04, 2015
http://reproducibility.cs.arizona.edu/v2/RepeatabilityTR.pdf
The results presented in the Report “Ancient Ethiopian genome reveals extensive Eurasian admixture throughout the African continent” were affected by a bioinformatics error.

Llorente et al. Science, 350, 6262
doi:10.1126/science.aad2879
## Barriers to Data and Code Sharing in Computational Science

Survey of Machine Learning Community, NIPS (Stodden, 2010):

<table>
<thead>
<tr>
<th>Code</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>77%</td>
<td>Time to document and clean up</td>
</tr>
<tr>
<td>52%</td>
<td>Dealing with questions from users</td>
</tr>
<tr>
<td>44%</td>
<td>Not receiving attribution</td>
</tr>
<tr>
<td>40%</td>
<td>Possibility of patents</td>
</tr>
<tr>
<td>34%</td>
<td>Legal Barriers (i.e. copyright)</td>
</tr>
<tr>
<td>-</td>
<td>Time to verify release with admin</td>
</tr>
<tr>
<td>30%</td>
<td>Potential loss of future publications</td>
</tr>
<tr>
<td>30%</td>
<td>Competitors may get an advantage</td>
</tr>
<tr>
<td>20%</td>
<td>Web/disk space limitations</td>
</tr>
</tbody>
</table>


Special Issue Reproducible Research Computing in Science and Engineering July/August 2012, 14(4)

Howison and Herbsleb (2013) "Incentives and Integration In Scientific Software Production" CSCW 2013.
Research Culture Needs Changing

Things I get credit for:
• Publishing papers
• Getting grants
• Societal impact (maybe)

Things I don’t get credit for:
• Releasing my software
• Making my software easy to use
• Supporting software for others to use
• Investing effort in learning new tools
• Being helpful

Ideas versus Implementation? Different forms of credit? Support?
Better software means considering how we train, develop, support and recognise software, and the people and careers that are the real infrastructure.
The Software Sustainability Institute

A national facility for cultivating better, more sustainable, research software to enable world-class research

- Software reaches boundaries in its development cycle that prevent improvement, growth and adoption
- Providing the expertise and services needed to negotiate to the next stage
- Developing the policy and tools to support the community developing and using research software

Supported by EPSRC Grant EP/H043160/1 + EPSRC/ESRC/BBSRC grant EP/N006410/1
Software

Helping the community to develop software that meets the needs of reliable, reproducible, and reusable research

Training

Delivering essential software skills to researchers via CDTs, institutions & doctoral schools

Outreach

Exploiting our platform to enable engagement, delivery & uptake

Collecting evidence on the community’s software use & sharing with stakeholders

Bringing together the right people to understand and address topical issues

Policy

Community

Would be impossible 68%
Possible, but difficult 21%
No effect 11%
Consultancy
50+ projects

Website & blog
150+ contributed articles
20,000 unique visitors per month
3,000 Twitter followers

Research
740 researchers
50,000 grants analysed

Guides
80+ guides
50,000 readers

Courses
35+ UK SWC workshops
1000+ learners

Outreach
130+ evaluations
4 surgeries

Policy
740 researchers
50,000 grants analysed

Workshops
20+ workshops organised

Training
2100 signatures

Campaigns
13 issues highlighted

Fellowship
61 domain ambassadors

Community
300+ RSEs engaged
Good enough practices

• Make sure you’re developing software in a reasonably good way
• Check that your colleagues and collaborator are too, and help them if they aren’t
• Good Enough Practices for Scientific Computing
Learn to be a carpenter

Teach basic lab skills for scientific computing so that researchers can do more in less time and with less pain.

Teach basic concepts, skills and tools for working more effectively with data. Workshops are designed for people with little to no prior computational experience.

Open source learning, that can be tailored to disciplines. “Train the trainers”: building a capable base of instructors.

admin@software-carpentry.org

admin@datacarpentry.org
Sharing Detailed Research Data Is Associated with Increased Citation Rate

Heather A. Piwowar, Roger S. Day, Douglas B. Frisema

Published: March 21, 2007 • DOI: 10.1371/journal.pone.0000308

Code Sharing Is Associated with Research Impact in Image Processing

In computational sciences such as image processing, publishing usually isn’t enough to allow other researchers to verify results. Often, supplementary materials such as source code and measurement data are required. Yet most researchers choose not to make their code available because of the extra time required to prepare it. Are such efforts actually worthwhile, though?
Get and give credit for software

- **Mechanisms**
  - Software citation e.g. Software Citation Working Group [https://www.force11.org/group/software-citation-working-group](https://www.force11.org/group/software-citation-working-group)

- **Tools**
  - Researcher Identifiers e.g. ORCID [http://orcid.org/](http://orcid.org/)
  - Alt-Metrics e.g. ImpactStory [http://impactstory.org/](http://impactstory.org/)

- **Be a better reviewer**
  - Ask to see code and data
  - Be constructive in your criticism
Publishing software papers is easy

http://openresearchsoftware.metajnl.com
Use the ecosystem of infrastructure and tools

GitHub, GitLab, jupyter, bitbucket, mercurial, Libraries.io, Jenkins, Zenodo, Travis CI, Buildbot, docker, Vagrant, CodeMeta, figshare.
Cultivate a community and encourage contributions

• Open source is more than just a license
  ▪ It’s about making it easier for people to collaborate
  ▪ And reducing replication of effort

• Encouraging and expanding at the right pace
  ▪ Efficient tools for research, not “generic” software
  ▪ Increased robustness, reuse

• Being open from the start rather than waiting until it’s “ready”
  ▪ Code is never perfect, always evolving
Incentives are important

Incentives and Integration In Scientific Software Production

Rewrite by original team: address fragility

Exploit new techniques / architectures

Facilitate hardware sales

Optimised for hardware

Fork to add specific functionality

Maintained separately

Software Sustainability Institute
UK STEM graduate career paths

Source: The Scientific Century, Royal Society, 2010 (revised to reflect first stage clarification from “What Do PhD's Do?” study)
Arman Bilge, Lexington High School wins MA state science fair using CIPRES

The Origin & Spread of HIV-1 Subtype B in the Americas

MSSEF 2012 Winners by Place and Name

<table>
<thead>
<tr>
<th>Exhibitor</th>
<th>School</th>
<th>Title</th>
<th>Place</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ackel, Ryan</td>
<td>Falmouth Academy</td>
<td>Effects of Nicotine and Viscous on Memory</td>
<td>Team 1st Place</td>
</tr>
<tr>
<td>Athalye, Anish</td>
<td>Mass. Academy of Math &amp; Science</td>
<td>Formation in Hammsande</td>
<td>1st Place</td>
</tr>
<tr>
<td>Shupatiya, Surya</td>
<td>Lexington High School</td>
<td>Investigating the Spread of the Influenza A Virus: A Phylogenetic Anal</td>
<td>Team 1st Place</td>
</tr>
<tr>
<td>Bilge, Arman</td>
<td>Lexington High School</td>
<td>The Origin &amp; Spread of HIV-1 Subtype B in the Americas</td>
<td>1st Place</td>
</tr>
<tr>
<td>Dodd, Oliver</td>
<td>Needham High School</td>
<td>Cancer Growth Regulators</td>
<td>1st Place</td>
</tr>
<tr>
<td>Meron, Addie</td>
<td>Westfield High School</td>
<td>The Adverse Effects of Consumer and</td>
<td>Team 1st Place</td>
</tr>
</tbody>
</table>

References

Software sustainability is more than writing good code; it’s about **cultivating** skills, community and cultural change…

…and it starts with you.

Slides: [http://dx.doi.org/10.6084/m9.figshare.3840303](http://dx.doi.org/10.6084/m9.figshare.3840303)
Further reading

Find out more about the SSI

- Community Engagement (Lead: Shoaib Sufi)
  - Fellowship Programme
  - Events and Workshops
- Consultancy (Lead: Steve Crouch)
  - Open Call for Projects / Collaborations
  - Software Evaluation
- Policy and Publicity (Lead: Simon Hettrick)
  - Case Studies / Policy Campaigns
  - Software and Research Blog
- Training (Lead: Aleksandra Nenadic)
  - Software Carpentry (300+ students/year)
  - Guides and Top Tips
- Journal of Open Research Software (Editor: Neil Chue Hong)

Collaboration between universities of Edinburgh, Manchester, Oxford and Southampton
Supported by EPSRC Grant EP/H043160/1 + EPSRC/ESRC/BBSRC grant EP/N006410/1