Journal of
open research software

Stodden, V and Miguez, S 2014 Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research. *Journal of Open Research Software,* 2(1): e21, pp.1-6, DOI: http://dx.doi.org/10.5334/jors.ay

ISSUES IN RESEARCH SOFTWARE

# Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research

Victoria Stodden* and Sheila Miguez*

The goal of this article is to coalesce a discussion around best practices for scholarly research that utilizes computational methods, by providing a formalized set of best practice recommendations to guide computational scientists and other stakeholders wishing to disseminate reproducible research, facilitate innovation by enabling data and code re-use, and enable broader communication of the output of computational scientific research. Scholarly dissemination and communication standards are changing to reflect the increasingly computational nature of scholarly research, primarily to include the sharing of the data and code associated with published results. We also present these Best Practices as a living, evolving, and changing document at http://wiki.stodden.net/Best_Practices.

**Keywords:** best practices; reproducible research; archiving; data sharing; code sharing; wiki; open science; computational science; scientific method

## Introduction

The goal of this article is to coalesce a discussion around best practices for scholarly research that utilizes computational methods, by providing a formalized set of best practice recommendations to guide computational scientists and other stakeholders wishing to disseminate reproducible research, facilitate innovation by enabling data and code re-use, and enable broader communication of the output of computational scientific research. Pervasive digitization is changing the practice of science by enabling massive data collection and storage, the tools to carry out and record analyses of these data, and also by providing a mechanism for communicating these new digital scholarly objects through the Internet. The potential is enormous: the technology is available to permit the open communication not only of the results of scientific investigation, but also the tools and data required to verify, extend, and understand the knowledge.

There is a movement within the computational science community to adapt communication standards to include the data and code associated with published findings, called the Reproducible Research Movement [1]. An ICERM workshop in December of 2012 on "Reproducibility in Computational Experimental Mathematics" [2] produced a workshop report with recommendations for enabling reproducibility and reliability in computational scientific findings [3, 4]. The July/August 2012 issue of IEEE Computing in Science and Engineering focused on Reproducible Research [5] and called for "changing the culture" of scientific research [6]. A Roundtable at Yale Law School in 2009 focused on the issue of reproducibility by bringing together computational scientists from many different disciplines and producing a declaration addressing the need for data and code sharing in computational science [7, 8]. Over the past few years many editorials and commentaries have continued these efforts [9–13]. The theme is similar: Without the data and computer codes that underlie scientific discoveries, published findings are all but impossible to verify. Computational results are frequently of a complexity that makes a complete enumeration of the steps taken to arrive at a result prohibitive in typical scientific publications today. As noted in 2009,

> At conferences and in publications, it's now completely acceptable for a researcher to simply say, "here is what I did, and here are my results." Presenters devote almost no time to explaining why the audience should believe that they found and corrected errors in their computations. The presentation's core isn't about the struggle to root out error — as it would be in mature fields — but is instead a sales pitch: an enthusiastic presentation of ideas and a breezy demo of an implementation. Computational science has nothing like the elaborate mechanisms of formal proof in mathematics or meta-analysis in empirical science. Many users of scientific computing aren't even trying to follow

* Department of Statistics, Columbia University, USA
vcs@stodden.net, shekay@gmail.com

> a systematic, rigorous discipline that would in principle allow others to verify the claims they make. How dare we imagine that computational science, as routinely practiced, is reliable! [1]

A necessary response to this crisis is the adoption of the practice of reproducible computational research, in which all details of the computations — the underlying data and the code that generated the results — are made conveniently available to others.

In this document we envision a computational environment that facilitates reproducibility as a digital concept beginning from data and tracing through the computational steps taken to achieve the published results. This distinguishes it from the replication of the experiment from first principles, including for example the regeneration of the raw data and the reimplementation of the data analysis de novo. We introduce the concepts of *vertical collaboration* and *horizontal collaboration*, to distinguish between the act of building on previously published research and that of carrying out joint research at the same point in time. We do not try to enumerate optimal environments for all possible research settings, rather we outline use cases with the hope of spurring greater discussion and development in this area of research. Although some best practice documents do exist for digital archivists [14], we know of only one other resource designed to communicate best practices for scientific computing [15]. We encapsulate some of these ideas in a wiki designed to facilitate the development and communication of best practices for computational scientists.

## Developing Best Practices

A typical computational scientist today is being inundated with new software tools to help with research [16], new requirements for publication [17], and evolving standards as his or her field responds to the changing nature and increasing quantity of available data [18]. Because of the speed of the changes occurring in scientific research, we chose to implement the best practice recommendations given in this paper as a wiki, available at http://wiki.stodden.net/Best_Practices. The hope is that parties with specialized or more complete knowledge will be able to add their expertise to the best practices document, to create a maximally useful document.

What follows is a series of principles for producing really reproducible computational science, and examples of implementations. We take as a starting point a National Academies of Science 2003 report, "Sharing Publication-Related Data and Materials: Responsibilities of Authorship in the Life Sciences" [19] stating:

> **Principle 1. (Chapter 3) Authors should include in their publications the data, algorithms, or other information that is central or integral to the publication—that is, whatever is necessary to support the major claims of the paper and would enable one skilled in the art to verify or replicate the claims.**
>
> This is a *quid pro quo*—in exchange for the credit and acknowledgement that come with publishing in a peer-reviewed journal, authors are expected to provide the information essential to their published findings. (p. 5)
>
> **Principle 2. (Chapter 3) If central or integral information cannot be included in the publication for practical reasons (for example, because a dataset is too large), it should be made freely (without restriction on its use for research purposes and at no cost) and readily accessible through other means (for example, on-line). Moreover, when necessary to enable further research, integral information should be made available in a form that enables it to be manipulated, analyzed, and combined with other scientific data.** (p. 5)
>
> **Principle 3. (Chapter 3) If publicly accessible repositories for data have been agreed on by a community of researchers and are in general use, the relevant data should be deposited in one of these repositories by the time of publication.** (p. 6)

Here Principle 1 calls for the dissemination of data, software, and all information necessary for a researcher to "verify or replicate the claims" made in the publication. Principles 2 and 3 provide guidance for the implementation of Principle 1. We adapt and extend these ideas into a series of Best Practice principles for computational scientists generally, and include these on the associated wiki. This is not meant to be a comprehensive list of all possible best practices in all circumstances, but a first cut at a generic case with the means provided through the wiki for modification and improvement.

## Best Practice Principles for Computational Science

1. **Open licensing should be used for data and code.** Best Practices indicate making the data and code maximally available and open for re-use. One way to make this legally possible is through the use of open licensing [20] and the Reproducible Research Standard [21]. This document assumes you have the legal right to make the data and code publicly available, or can obtain permission from the data and code owners. Best practices indicate negotiating open licensing for data and code with collaborators prior to beginning the research project. [21a]
2. **Workflow tracking should be carried out during the research process.** Provenance, workflow tracking, and publishing environments are important tools that help enable reproducibility and reuse by others, while minimizing the burden on the researcher. For example, using a version control system such as git or mercurial throughout the project simplifies making the code available at the time of

publication. Setting up an issue tracker and/or an agile planning board for example can make communication more efficient, as can using an open notebook as a record of provenance and workflow. For an example of work that follows these processes, refer to [22].

3. **Data must be available and accessible.** Availability and accessibility can be broken down into three sub-discussions.
   i. **Version Control for Data:** At minimum, provide a version for datasets you generate or collect. If you did not generate or collect the data yourself, provide a link and citation to the source of each dataset you incorporated, including which *version* of the data you used (if the data source does not provide version information, provide the exact time and date you accessed the data). As of yet there are no standards or conventions being widely practiced, but this is a very active topic. An additional best practice would be to include a DOI (digital objet identifier) and a hash for bit-level identification of the data [22a].
   ii. **Raw Data Availability:** Results should be reproduced from the earliest digital data in the experiment, whether that is raw data coming from instruments or observations, or data as accessed from a secondary source. It defeats the purpose to supply a "cleaned" version of the data if it is impossible to access the methodology of the cleaning, for example. The goal is that all data manipulations be made transparent, beginning with the initial version of the data with which the researcher started working. Meta-data should accompany the raw data. Meta-data should be machine and human-readable and use standard terminology [23].
   iii. **External and Redundant Storage:** In the simplest case, there are no external data files, for example in some simulations. In the most complex case, data are massive, distributed, and possibly updated in real time. The intermediary cases involve data files that can be readily downloaded and accessed by the user. Going roughly from the simplest to the most challenging cases:
      - **Simulated Data:** In the case of simulated data, sharing the code that generated the data are enough if the code executes reasonably quickly. When a simulation takes an extended amount of time to regenerate the simulated data, pre-calculated data should be provided along with the code used to generate them.
      - **"Small" Static Data:** We classify small data as datasets less than 2GB in size, but this is a relative term that may change depending on your system, download speeds, and repository storage capacity. If you are able to store your datasets at your institution and link to them from your institutional webpage, that is a good step. It will help your citation count, help others find your data, and help verification of your work. But it is *insufficient*. You must make your datasets available at an external repository dedicated to providing access to scientific datasets in perpetuity. These datasets should be versioned as discussed previously to enable citation to the particular version that will permit verification of the findings based on it in the paper.
      - **Large "Static" Data:** Datasets greater than 2GB encounter a number of problems smaller datasets do not. The first is access, since uploading and download very large files is very time-consuming if not prohibitive. If you created the dataset yourself, you may have to make a one-time upload. If you did not create the dataset yourself, it is likely sufficient to cite the version of the third party data that you accessed and provide the computer code you used to manipulate the data.

        Large data are very likely to come with its own infrastructure. It may already reside in a domain-specific repository designed for access, such as the Sloan Digital Sky Survey, the National Institute for Health's caBIG data sharing portal or its Genome-wide Association Studies, EarthCube, or a number of others. Each of these have or are developing policies on data re-integration to permit uploading of data that has undergone changes, or it may be possible to link to / cite the version of the dataset(s) you used in your research and provide code that replicates the manipulations you carried out on that snapshot of the data.

        Infrastructure for large data are becoming available for researchers beyond these groups of domain-specific data repositories. Both Globus Online and HUBzero provide different types of computational environments for non-domain-specific scientific research and their own methods for data availability. Both are geared toward cloud computation, as is the National Science Foundation's XSEDE scientific computing environment. TheDataHub.org is an entirely open source data repository. Many of these infrastructure efforts provide suggested citations and versioning for data, and this is just as crucial as it is in the small data case.
      - **Streaming Data:** These data seem like the most challenging case but are actually likely to fall into one of the above categories. Published results must be obtained on some amount of fixed data, and this particular dataset can be readily shared as above. In these cases it is likely scientifically relevant to validate models on future streams of data, but that is left to the domain of new, potentially publishable research that will share its data when published.

Some domain specific dissemination platforms include the Machine Learning Open Source Software (MLOSS) (both software and data), The Stanford Microarray Database, and the Protein Data Bank (PDB).

There are exceptions to this principle, including confidential data and proprietary data. Workarounds should be attempted and may exist for confidential data [24] and proprietary data [25].

4. **Code and methods must be available and accessible.** Input values should be included with code and scripts that generated the results, along with random number generator seeds if randomization is used. Version control should be utilized for code development, facilitating re-use by others. This discussion can be broken into subdiscussions.
   - **Version Control for Code / Making the Code Available Externally:** These goals can be accomplished together by using a hosted version control system with a public facing option. There are many advantages to using version control for the code you and your collaborators write during a project, and releasing the code to the wider world using version control is important. Doing so permits others to know precisely which version of the code generated what results, allows others to make modifications and feed them back into the system without disrupting the original code, and perhaps most importantly permits a community to develop around the research questions, complete with mature functionality for bug tracking and fixes, new code developments, centralized code dissemination, and collaboration. Here is an example of scientific code associated with a published paper, available on GitHub.com, and a second example with the code available on BitBucket.org. This paper (http://arxiv.org/abs/1201.3035) has its code available on Github.com at https://github.com/ketch/RK-opt and this paper (http://arxiv.org/abs/1111.6583) has its code available on BitBucket.org at https://bitbucket.org/ahmadia/pyclaw-sisc-rr.
   - **Version Control for Environments / Making Environments Available and Documented:** This practice is gaining traction in the research community, and is already common in industry. Along with the code, store information about the code's environment in version control. Vagrant and Docker are two technologies to consider for use. For example, the BrainScaleS (Brain-inspired multiscale computation in neuromorphic hybrid systems) project provides a Docker image [26] for the neural network simulators nest, neuron, brian, with PyNN and music. A researcher who uses this technology stack can include a Dockerfile with their repository.
   - **Code Samples and Test Data:** Projects in the research community such as DataVerse Network [27], and ResearchCompendia.org [28] allow code and data to be shared with some ability to run the code. In the wider software community, projects such as Jenkins, Travis CI, and drone.io are projects that allow software projects to run jobs limited to the environments that these system support. Outputs from these runs can also be shared. Authors should provide some code samples with test parameters and data sets that demonstrate the codes use.
   - **"Really Big" Codebases:** These codebases are likely already in version control, but are of a complexity that makes visual interpretation of the code next to impossible. Reproducibility in this case requires testing the functionality of the code, to ensure whether it is operating as the researchers expect. Common software testing methods can be applied to these codebases, such as unit tests, integration tests, and regression tests [15]. Such testing is not broadly implemented in shared scientific software, but it is standard practice in industry and the open source software community.
5. **All 3rd party data and software should be cited.** If you use data you did not collect from scratch, or code you did not write, however little, cite it. Include the **source**, include the **author**, and include the **date and time** you accessed the data or code you used. Best practices indicate including a unique identifier in your citation, preferably a SHA-1 hash such as The DataVerse Network's UNF (see http://thedata.org/book/universal-numerical-fingerprint). The git version control system also uses SHA-1 hashing to identify and track code. Having a unique identifier is important since it provides a check that the data and code are what they are thought to be and it gives a way of versioning and establishing provenance.
   - Help people cite the data and code you release. Include a suggested citation such as:

```
Stodden V, Guo P, Ma Z (2013) "Toward
Reproducible Computational Research: An
Empirical Analysis of Data and Code Policy
Adoption by Journals." PLoS ONE 8(6):
e67111.
doi:10.1371/journal.pone.0067111
```

   - **Citation Standards:** There are several entities working to establish citation standards for data, for example ORCID (http://about.orcid.org/faq), DataCite is another (http://www.datacite.org) and EZID (http://n2t.net/ezid), but citation standards for scientific code are not as well covered. The "Code as a Research Object" [29] project from Mozilla Science, GitHub, and Figshare is a working proof of concept that generates a DOI for a code repository in GitHub. Assigning a Digital Object Identifier is excellent and helps establish provenance and citation. At the moment there are no @data or @code fields for BibTex entries. In the meantime best practices would suggest using the @misc field to create a citation for data or code. Here is the BibTex format for @misc:

```
   @MISC{citation_key,
   required_fields [, optional_fields] }
Required fields: none
Optional fields: author, title,
howpublished,
month, year, note, key
```

- Here is an example of a code citation, as suggested by http://www.ict.swin.edu.au/research/projects/helix/helix-cite.html:

  Rajesh Vasa, Markus Lumpe and Allan Jones, Helix - Software Evolution Data Set, http://www.ict.swin.edu.au/research/projects/helix, Swinburne University of Technology, 2010.
- Here is the corresponding BibTex code:

```
@MISC{Helix10a,
   title = {{Helix - Software Evolution
Data
Set}},
   author={Rajesh Vasa, Markus Lumpe, and
Allan
Jones}
   howpublished = {\url{http://http://www.
ict.
swin.edu.au/research/projects/helix}},
   year = {2010},
   key = {Helix10a},
   url = {http://www.ict.swin.edu.au/
research/
projects/helix}
   }
```

  Note that you may need \usepackage{url} to execute.
- **Code and plagiarism:** Unlike other authors' text, code can and perhaps should be re-used exactly as written, but all use should be cited, just as is standard practice for research articles. Terms of use accompanying the software would of course be respected. Best practices include licensing software under an attribution-only license such as the MIT license or the Modified BSD license as recommended in [20] and [21].

6. **Influences from sources external to the research process:**
   - There are often specific data and code sharing guidelines associated with funded research. For example,
     - National Institutes of Health Funded Research: http://grants.nih.gov/grants/policy/data_sharing/data_sharing_guidance.htm
     - National Science Foundation Funded Research: NSF Data Management Plan, Jan. 2011. http://www.nsf.gov/bfa/dias/policy/dmp.jsp
     - Welcome Trust Funded Research: http://www.wellcome.ac.uk/About-us/Policy/Spotlight-issues/Data-sharing/
     - University of Minnesota "Funding Agency and Data Management Guidelines."
     - Stanford University Office of Technology Licensing: http://otl.stanford.edu/about/resources/about_resources.html
   - In these cases, for example researchers funded by the National Institutes for Health or the National Science Foundation, are bound by the conditions of their grant. We hope that conflicts between this set of best practices and other requirements do not exist but in the event that they do we advocate two reactions. The first is to follow all legally binding requirements, of course, the second is to voice concern to the source of the requirements if they could be improved by moving closer to this set of best practices. This situation can happen in reverse as well. If new ideas arise that improve reproducibility and reliability of research findings, and advancement of scientific discovery, we hope these made their way into this set of best practices to improve them. This is one reason for implementing the set in a wiki format, to permit new knowledge and practice to surface and be incorporated into our current state of the art.

## Conclusion

This article attempts to seed a conversation around best practices for publishing computational scientists, through the traditional medium of the published paper and a community-editable wiki. This can be seen as a response to the Reproducible Research movement, through which computational scientists have been moving toward research practices that include making the data and code underlying a published results conveniently available. Open questions remain. This effort has several lofty goals, including clarifying best practices for computational scientists, establishing community standards, providing a central discussion point for evolving best practices, accelerating discoveries by facilitating reproducible computational science and data and code re-use, and supporting the transfer of the technology underlying scientific results, with the aim of increasing the reliability of published findings and addressing the credibility crisis in computational science. This article is intended as a set of discussion points to help advance these goals.

## References


1. **Donoho, D L, Maleki, A, Rahman, I U, Shahram, M** and **Stodden, V** 2009 Reproducible Research in Computational Harmonic Analysis. *Computing in Science and Engineering,* 11(1): 8–18. DOI: http://dx.doi.org/10.1109/MCSE.2009.15
2. **Stodden, V, Borwein, J** and **Bailey, D H** 2013 Setting the Default to Reproducible. *Computational Science Research,* 46(5). Available at: https://www.siam.org/news/news.php?id=2078
3. **Reproducibility in Computational and Experimental Mathematics** 2012 ICERM. Available at: http://icerm.brown.edu/tw12-5-rcem/
4. **Bailey, D H, Borwein, J M** and **Stodden, V** 2013 Set the Default to "Open". *Notices of the American Mathematical Society*, June 2013.
5. **Stodden, V** 2012 Reproducible Research for Scientific Computing. *Computing in Science and Engineering*, 14(4): 11–56. DOI: http://dx.doi.org/10.1109/MCSE.2012.82
6. **Mitchell, I M, LeVeque, R J and Stodden, V** 2012 Reproducible research for scientific computing: Tools and strategies for changing the culture. *Computing*

*in Science and Engineering,* 14(4): 13–17. DOI: http://dx.doi.org/10.1109/MCSE.2012.38

7. **Stodden, V** 2009 Yale Roundtable on Reproducible Research. Data and Code Sharing Roundtable.. Available at: http://www.stanford.edu/~vcs/Conferences/RoundtableNov212009/
8. **Yale Roundtable Participants** 2010 Reproducible Research. *Computing in Science and Engineering,*12(5): 8–13. DOI: http://dx.doi.org/10.1109/MCSE.2010.113
9. 2011 Devil in the details. *Nature,* 470(7334): 305–306. DOI: http://dx.doi.org/10.1038/470305b
10. **Hanson, B, Sugden, A** and **Alberts, B** 2011 Making Data Maximally Available. *Science,* 331(6018): 649–649. DOI: http://dx.doi.org/10.1126/science.1203354
11. **Merali, Z** 2010 Computational science: ...Error. *Nature News,* 467(7317): 775–777.
12. **Barnes, N** 2010 Publish your computer code: it is good enough. *Nature News*, 467(7317): 753–753.
13. **Morin, A, Urban, J, Adams, P, Foster, I, Sali, A, Baker, D,** et al 2012 Shining light into black boxes. *Science,* 336(6078): 159–160. DOI: http://dx.doi.org/10.1126/science.1218263
14. **The University Library of the University of Illinois at Urbana-Champaign**. Best Practices for Creating Digital Collections. Available at: http://www.library.illinois.edu/dcc/bestpractices/contents.html
15. **Wilson, G, Aruliah, D A, Brown, C T, Chue Hong, N P, Davis, M, Guy, R T,** et al 2014 Best Practices for Scientific Computing. *PLoS Biology*, 12(1): e1001745. DOI: http://dx.doi.org/10.1371/journal.pbio.1001745
16. **Stodden, V** 2012 Reproducible Research: Tools and Strategies for Scientific Computing. *Computing in Science and Engineering,* 14(4): 11–12. DOI: http://dx.doi.org/10.1109/MCSE.2012.82
17. **Stodden, V, Guo, P** and **Ma, Z** 2013 Toward Reproducible Computational Research: An Empirical Analysis of Data and Code Policy Adoption by Journals. *PLoS ONE,* 8(6): e67111. DOI: http://dx.doi.org/10.1371/journal.pone.0067111
18. **Lazer, D, Pentland, A, Adamic, L, Aral, S, Barabási, A-L, Brewer, D,** et al 2009 Computational Social Science. *Science,* 323(5915): 721–723. DOI: http://dx.doi.org/10.1126/science.1167742
19. **Committee on Responsibilities of Authorship in the Biological Sciences NRC** 2003 Sharing Publication-Related Data and Materials: Responsibilities of Authorship in the Life Sciences. The National Academies Press. Available at: http://www.nap.edu/openbook.php?record_id=10613
20. Open Source Licenses. Available at: http://opensource.org/licenses [Last accessed 04 April 2014].
21. **Stodden, V** 2009 Enabling Reproducible Research: Open Licensing for Scientific Innovation. Rochester, NY. *Social Science Research Network,* 1362040. Available at: http://papers.ssrn.com/abstract=1362040.

21a.See e.g. http://blog.stodden.net/2012/12/24/data-access-going-the-way-of-journal-article-access/

22. **Boettiger, C** Lab Notebook. Available at: http://carlboettiger.info/lab-notebook.html [Last accessed 04 April 2014].

22a. **Altman, M,** and **King, G** 2007 A Proposed Standard fr the Scholarly Citations of Quantitative Data. *D-Lib Magazine*, March/April 2007. Available at: http://www.dlib.org/dlib/march07/altman/03altman.html

23. **DataONE**. metadata. Available at: http://www.dataone.org/best-practices/metadata [Last accessed 04 April 2014].
24. **Dwork, C** 2006 Differential Privacy. In: Bugliesi, M, Preneel, B, Sassone, V and Wegener, I (eds.) *Automata, Languages and Programming.* Berlin, Heidelberg: Springer. pp. 1–12. Available at: http://research.microsoft.com/apps/pubs/default.aspx?id=64346 [Last accessed 04 April 2014].
25. **Stodden, V** 2011 Innovation and Growth through Open Access to Scientific Research: Three Ideas for High-Impact Rule Changes. Rochester, NY. *Social Science Research Network,* 1757982. Available at: http://papers.ssrn.com/abstract=1757982
26. **brainscales/neural-networks Docker index** 2014 Available at: https://index.docker.io/u/brainscales/neural-networks/
27. **Crosas, M** 2011 The Dataverse Network®: An Open-Source Application for Sharing, Discovering and Preserving Data. *D-Lib Magazine,* January 2011: 17(1/2). Available at: http://www.dlib.org/dlib/january11/crosas/01crosas.html [Last accessed 04 April 2014].
28. **Stodden, V, Miguez, S** and **Seiler, J** (forthcoming 2014) Introducing ResearchCompendia: CyberInfrastructure for Reproducibility and Collaboration in Computational Science.
29. How should software be cited? 2014 mozillascience/code-research-object GitHub Issue Tracker. Available at: https://github.com/mozillascience/code-research-object/issues/12

**How to cite this article**: Stodden, V and Miguez, S 2014 Best Practices for Computational Science: Software Infrastructure and Environments for Reproducible and Extensible Research. *Journal of Open Research Software,* 2(1): e21, pp. 1-6, DOI: http://dx.doi.org/10.5334/jors.ay

**Published:** 9 July 2014

**Copyright**: © 2014 The Author(s). This is an open-access article distributed under the terms of the Creative Commons Attribution 3.0 Unported License (CC-BY 3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited. See http://creativecommons.org/licenses/by/3.0/.

]u[ *Journal of Open Research Software* is a peer-reviewed open access journal published by Ubiquity Press.

OPEN ACCESS