

Towards standard practices for sharing computer code and programs in neuroscience

Stephen J. Eglén^{1,†}, Ben Marwick², Yaroslav O. Halchenko³, Michael Hanke^{4,5},
Shoab Sufi⁶, Pádraig Gleeson⁷, R. Angus Silver⁷, Andrew P. Davison⁸,
Linda Lanyon⁹, Mathew Abrams⁹, Thomas Wachtler¹⁰,
David J. Willshaw¹¹, Christophe Pouzat¹², Jean-Baptiste Poline^{13,†}

February 8, 2016

¹ Cambridge Computational Biology Institute, Department of Applied Mathematics and Theoretical Physics, University of Cambridge, UK

² Department of Anthropology, University of Washington, Seattle, WA 98195-3100 USA

³ Department of Psychological and Brain Sciences, Dartmouth College, Hanover, NH 03755 USA

⁴ Institute of Psychology II, Otto-von-Guericke-University Magdeburg, 39106 Magdeburg, Germany

⁵ Center for Behavioral Brain Sciences, 39106 Magdeburg, Germany

⁶ Software Sustainability Institute, University of Manchester, UK

⁷ Department of Neuroscience, Physiology and Pharmacology, University College London, UK

⁸ Unité de Neurosciences, Information et Complexité, CNRS, Gif sur Yvette, France

⁹ International Neuroinformatics Coordinating Facility, Karolinska Institutet, Stockholm, Sweden

¹⁰ Department of Biology II, Ludwig-Maximilians-Universität München, Germany

¹¹ Institute for Adaptive and Neural Computation, School of Informatics, University of Edinburgh, UK

¹² MAP5 Paris-Descartes University and CNRS UMR 8145, 45 rue des Saints-Pères, 75006 Paris, France

¹³ Henry H. Wheeler, Jr. Brain Imaging Center, Helen Wills Neuroscience Institute, University of California, Berkeley, USA

† Corresponding authors: S.J.Eglen@damtp.cam.ac.uk, jbpoline@gmail.com

Background

2 Many areas of neuroscience are now critically dependent on computational tools to help
understand the large volumes of data being created. Furthermore, computer models are
4 increasingly being used to help predict and understand the function of the nervous sys-
tem. Many of these computations are complex and often cannot be concisely reported in
6 the methods section of a scientific article. In a few areas there are widely used software
packages for analysis (e.g., SPM, FSL, AFNI, BrainVoyager, FreeSurfer in neuroimaging)
8 or simulation (e.g. NEURON, NEST, Brian). However, we often write new computer
programs to solve specific problems in the course of our research. Some of these pro-
10 grams may be relatively small scripts that help analyze all of our data, and these rarely
get described in papers. As authors, how best can we maximize the chances that other
12 scientists can reproduce our computations or reuse our methods on their data? Is our
research reproducible¹?

14 To date, the sharing of computer programs underlying neuroscience research has
been the exception (see below for some examples), rather than the rule. However, there
16 are many potential benefits to sharing these programs, including increased understand-
ing and reuse of your work. Furthermore, open source programs can be scrutinized and
18 improved, whereas the functioning of closed source programs remains forever unclear².
Funding agencies, research institutes and publishers are all gradually developing policies
20 to reduce the withholding of computer programs relating to research³. The Nature fam-
ily of journals has recently published opinion pieces in favor of sharing whatever code is
22 available, in whatever form^{4,5}. More recently, since October 2014, all Nature journals re-
quire papers to include a statement declaring *whether* the programs underlying central re-
24 sults in a paper are available. In April 2015 *Nature Biotechnology* offered recommendations
for providing code with papers and began asking referees to give feedback on their ability
26 to test code that accompanies submitted manuscripts⁶. In July 2015 F1000Research stated
that “Software papers describing non-open software, code and/or web tools will be re-
28 jected” ([http://f1000research.com/channels/f1000-faculty-reviews/for-authors/
article-guidelines/software-tool-articles](http://f1000research.com/channels/f1000-faculty-reviews/for-authors/article-guidelines/software-tool-articles)). Also in July 2015, BioMed Central in-
30 troduced a minimum standards of reporting checklist for BMC Neuroscience and several
other journals, requiring submissions to include a code availability statement and for
32 code to be cited using a DOI or similar unique identifier⁷. **We believe that all journals
should adopt policies that highly encourage, or even mandate, the sharing of software
34 relating to journal publications.**

What should be shared?

36 It may not be obvious what to share, especially for complex projects with many collabora-
tors. As advocated by Claerbout and Donoho, for computational sciences the scholarship
38 is not the article; the “scholarship is the complete software [...]”^{8,9}. So, ideally, you should
share as much code and data as is needed to allow others to reproduce your work, but

40 this may not be possible or practical. However, it is expected that you will share key
parts of the work, e.g. implementations of novel algorithms or analyses. At a mini-
42 mum, we suggest following the recommendation of submission of work to ModelDB¹⁰,
i.e. to share enough code, data and documentation to allow at least one key figure from
44 your manuscript to be reproduced. However, by adopting appropriate software tools, as
mentioned in the next section, it is now relatively straightforward to share the materials
46 required to regenerate *all* figures and tables. On the other hand, code that is not novel
because it is already available, or that you feel that is unlikely to be of use to others need
48 not be shared. This includes code that performs simple preprocessing or statistical tests,
or code that deals with local computing issues such as hardware and software configu-
50 rations. Finally, if your work is computationally intensive and requires a long time to
run (e.g. many weeks), you may prefer to provide a small “toy” example to demonstrate
52 your code.

By getting into the habit of sharing as much as possible, not only do you help others
54 who wish to reproduce your work (which is a basic tenet of the scientific method), you
will be helping other members of your laboratory, or even yourself in the future. By
56 sharing your code publicly, you are more likely to write higher-quality code¹¹, and you
will know where to find it after you’ve moved on from the project¹², rather than the code
58 disappearing on a colleague’s laptop when they leave your group. You will be part of a
community and benefit from the code shared by others, thus contributing to a reduction
60 in software development time for yourself and others.

Simple steps to help you share your code

62 Once you have decided *what* you plan to share, here are some simple guidelines for *how* to
share your work. Ideally, these principles should be followed throughout the lifetime of
64 your project, not just at the end when you wish to publish your results. Guidelines similar
to these have been proposed recently in many areas of science^{13–15}, suggesting that they
66 are part of norms that are emerging across disciplines. In the ‘further reading’ section
below, we list some specific proposals from other fields that expand on the guidelines we
68 suggest here.

Version control Use a version control system (such as Git) to develop the code¹⁶. The
70 version control database can then be easily and freely shared with others using
sites such as <http://github.com>¹⁷ or <https://bitbucket.org>. These sites allow
72 you fine control over private versus public access to your code. This means that you
can keep your code repository private during its development, and then publicly
74 share the repository at a later stage e.g. at the time of publication. It also makes it
easy for others to contribute to your code, and to adapt it for their own uses.

76 **Persistent URLs** Generate stable URLs (such as a DOI) for key versions of your soft-
ware. Unique identifiers are a key element in demonstrating the integrity and re-

78 producibility of research¹⁸, and allow referencing of the exact version of your code
used to produce figures. DOIs can be obtained freely and routinely with sites such
80 as <http://zenodo.org> and <http://figshare.com>. If your work includes com-
puter models of neural systems, you may wish to consider depositing these models
82 in established repositories such as ModelDB¹⁰, Open Source Brain¹⁹, INCF Software
Center²⁰ or NITRC²¹. Some of these sites allow for private sharing of repositories
84 with anonymous peer reviewers. Journal articles that include a persistent URL to
code deposited in a trusted repository meet the requirements of level two of the
86 ‘analytic methods (code) transparency’ standard of the TOP guidelines¹³.

License Choose a suitable license for your code to assert how you wish others to reuse
88 your code. For example, to maximize reuse, you may wish to use a permissive
license such as MIT or BSD²². Licenses are also important to protect you from others
90 misusing your code. Visit <http://choosealicense.com/> to get a simple overview
of which license to choose, or [http://www.software.ac.uk/resources/guides/
92 adopting-open-source-licence](http://www.software.ac.uk/resources/guides/adopting-open-source-licence) for a detailed guide.

Etiquette When working with code written by others, observe Daniel Kahneman’s ‘re-
94 producibility etiquette’²³ and have a discussion with the authors of the code to give
them a chance to fix bugs or respond to issues you have identified before you make
96 any public statements. Cite their code in an appropriate fashion.

Documentation Contrary to popular expectations, you do not need to write extensive
98 documentation or a user’s guide for the code still be to useful to others⁴. How-
ever, it is worth providing a minimal README file to give an introduction to what
100 the code does, and how to run it. For example, you should provide instructions
on how to regenerate a key result, or a particular figure from a paper. Literate
102 programming methods, where code and narrative text are interwoven in the same
document, make documentation semi-automatic and can save a lot of time when
104 preparing code to accompany a publication^{24,25}. However, these methods admit-
tedly take more time to write in the first instance, and you should be prepared to
106 rewrite documentation when rewriting code. In any cases, well-documented code
allows for easier re-use and checking.

108 **Tools** Consider using modern, widely used software tools that can help with making
your computational research reproducible. Many of these tools have already been
110 used in neuroscience and serve as good examples to follow, for example Org mode²⁶,
IPython/Jupyter²⁷ and Knitr²⁸. Virtualization environments, such as VirtualBox
112 appliances and Docker containers, can also be used to encapsulate or preserve all
of the computational environment so that other users can run your code without
114 having to install numerous dependencies²⁹.

Case studies As well as the examples listed above in Tools²⁶⁻²⁸, there are many prior

116 examples to follow when sharing your code. For example, some prominent exam-
118 ples of reproducible research in computational neuroscience include Vogels et al.³⁰
and Waskom et al.³¹; see <https://github.com/WagnerLabPapers> for details. The
120 ModelDB repository contains over 1000 computational models deposited with in-
structions for reproducing key figures to papers e.g. [https://senselab.med.yale.
122 edu/ModelDB/showModel.cshtml?model=93321](https://senselab.med.yale.edu/ModelDB/showModel.cshtml?model=93321) for a model of activity-dependent
conductances³².

Data Any experimental data collected alongside the software should also be released.
124 For small datasets, this could be stored alongside the software, although it may
be preferable to store experimental data separately in an appropriate repository.
126 Both PLOS and Scientific Data maintain useful lists of subject-specific and gen-
eral repositories for data storage, see [http://journals.plos.org/plosbiology/
128 s/data-availability#loc-recommended-repositories](http://journals.plos.org/plosbiology/s/data-availability#loc-recommended-repositories) and [http://www.nature.
com/sdata/data-policies/repositories](http://www.nature.com/sdata/data-policies/repositories).

130 **Standards** Use of community standards where appropriate should be encouraged. In
computational neuroscience for example, PyNN³³ and NeuroML³⁴ are widely used
132 formats for making models more accessible and portable across multiple simula-
tors.

134 **Tests** Testing the code has long been recognized as a critical step in software industry but
the practice is not widely adopted yet by researchers. We recommend including test
136 suites that demonstrate the code is producing the correct results³⁵. These tests can
be at a low level (testing each individual function, called unit testing) or at a higher
138 level (e.g. testing that the program yields correct answers on simulated data)³⁶.
Linking tests to continuous integration services (such as Travis CI, [https://travis-
140 ci.org](https://travis-ci.org)) allows these tests to be automatically run each time a change is made to the
code, ensuring failing tests are immediately flagged and can be dealt with quickly.

Further reading (note to editor: please make this a box feature)

Varsha Khodiyar 2015 Code Sharing — read our tips and share your own. Scientific Data Blog, February 19, 2015. <http://blogs.nature.com/scientificdata/2015/02/19/code-sharing-tips/>

Leveque Randall 2013. Top ten reasons to not share your code (and why you should anyway). SIAM News, April 2013, [http://sinews.siam.org/DetailsPage/tabid/607/ArticleID/386/](http://sinews.siam.org/DetailsPage/tabid/607/ArticleID/386/Top-Ten-Reasons-To-Not-Share-Your-Code-and-why-you-should-anyway.aspx)

[Top-Ten-Reasons-To-Not-Share-Your-Code-and-why-you-should-anyway.aspx](http://sinews.siam.org/DetailsPage/tabid/607/ArticleID/386/Top-Ten-Reasons-To-Not-Share-Your-Code-and-why-you-should-anyway.aspx)

Stodden V., & Miguez, S., 2014. Best practices for computational science: software infrastructure and environments for reproducible and extensible research. *Journal of Open Research Software*. 2(1), p.e21. DOI: <http://doi.org/10.5334/jors.ay>

Stodden, V., Leisch, F., & Peng, R. (Eds.). (2014). *Implementing reproducible research*. CRC press, Chapman and Hall.

Halchenko, Y. O. and Hanke, M. (2015). Four aspects to make science open “by design” and not as an after-thought. *GigaScience*, 4. DOI: <http://doi.org/10.1186/s13742-015-0072-7>

Sandve, G. K., Nekrutenko, A., Taylor, J., & Hovig E (2013) Ten simple rules for reproducible computational research. *PLoS Comput Biol* 9:e1003285.

142

Online communities discussing code sharing (note to editor: please make this a box feature)

StackExchange and related projects StackExchange is a network of free and highly active question-and-answer websites. Two members of the network are relevant to questions of code sharing: <http://stackoverflow.com/> which is dedicated to questions about programming in any language in any context, and <http://academia.stackexchange.com/questions/tagged/reproducible-research> which is focused questions relating to reproducible research in academic context. A related project is <https://neurostars.org/> which is a similar free public Q&A website focused on neuroinformatics questions, and with many questions on software packages, etc.

Scientists for Reproducible Research This is an international multi-disciplinary email list that discusses a wide range of issues relating to code sharing: <https://groups.google.com/forum/#!forum/reproducible-research>

GitHub GitHub is an online repository for computer code and programs that has a large community of researchers that develop and share their code openly on the site. GitHub is the largest and most active code sharing site (others include BitBucket and GitLab) and has convenient tools for facilitating efficient collaborative coding³⁷. If you are using an open source program you may find a community of users and developers active on GitHub, where you can ask questions and report problems.

144 **Closing remarks**

146 Changing the behaviors of neuroscientists so that they make their code more available
148 will likely be resisted by those who do not see the community benefits as outweighing
150 the personal costs of the time and effort required to share code³⁸. The community ben-
152 efits, in our view, are obvious and substantial: we can demonstrate more robustly and
154 transparently the reliability of our results, we can more easily adapt methods developed
by others to our data, and the impact of our work increases as others can similarly reuse
our methods on their data. Thus, we will endeavor to lead by example, and follow all
these practices as part of our future work in all scientific publications. Even if the code
we produce today will not run ten years from now, it will still be a more precise and
complete expression of our analysis than the text of the methods section in our paper.

156 However, exhortations such as this editorial are only a small part of making code
158 sharing a normal part of doing neuroscience; many other activities are important. All re-
160 searchers should be trained in sound coding principles; such training is provided by or-
ganizations such as Software Carpentry³⁶ and through national neuroinformatics nodes,
e.g. <http://python.g-node.org>. Furthermore, we should request code and data when
reviewing, and submit to and review for journals that support code sharing. Grant pro-

posals should be checked for mentions of code availability, and we should encourage
162 efforts toward openness in hiring, promotion, and reference letters³⁹. Funding agencies
and publishers should also consider mandating code sharing by default. This combina-
164 tion of efforts on a variety of fronts will increase the visibility of research accompanied
by open source code, and demonstrate to others in the discipline that code sharing is a
166 desirable activity that helps move the field forward.

We believe that the sociological barriers to code sharing are harder to overcome than
168 the technical ones. Currently, academic success is strongly linked to publications and
there is little recognition for producing and sharing code. Code may also be seen as
170 providing a private competitive advantage to researchers. We challenge this view and
propose that code be regarded as part of the research products which should be shared by
172 default, and that there should be an obligation to share code for those conducting publicly
funded research. We hope our code availability review will help establish such sharing
174 as the norm. Moreover, we are advocating for code sharing as part of a broader culture
change embracing transparency, reproducibility, and re-usability of research products.

176 Acknowledgments

This article is based upon discussions from a workshop to encourage sharing in neu-
178 roscience, held in Cambridge, December 2014. It was financially supported and orga-
nized by the International Neuroinformatics Coordinating Facility (<http://www.incf.org>),
180 with additional support from the Software Sustainability institute (<http://www.software.ac.uk>).

182 References

1. *Challenges in irreproducible research* <<http://www.nature.com/nature/focus/reproducibility>>.
184
2. Vihinen, M. No more hidden solutions in bioinformatics. *Nature* **521**, 261 (2015).
- 186 3. Morin, A *et al.* Research priorities. Shining light into black boxes. *Science* **336**, 159–160 (2012).
- 188 4. Barnes, N. Publish your computer code: it is good enough. *Nature* **467**, 753 (2010).
5. Ince, D. C., Hatton, L. & Graham-Cumming, J. The case for open computer pro-
190 grams. *Nature* **482**, 485–488 (2012).
6. Rebooting review. *Nature Biotech* **33**, 319–319 (2015).
- 192 7. Kenall, A. *et al.* Better reporting for better research: a checklist for reproducibility. *BMC Neuroscience* **16**, 1–3 (2015).
- 194 8. Claerbout, J. & Karrenbach, M. *Electronic documents give reproducible research a new meaning in Proc. 62nd Ann. Int. Meeting of the Soc. of Exploration Geophysics* (1992), 601–604. <<http://library.seg.org/doi/pdf/10.1190/1.1822162>>.
196

- 198 9. Donoho, D. L. An invitation to reproducible computational research. *Biostatistics (Oxford, England)* **11**, 385–8 (2010).
- 200 10. Hines, M. L., Morse, T., Migliore, M., Carnevale, N. T. & Shepherd, G. M. ModelDB: A Database to Support Computational Neuroscience. *Journal of Computational Neuroscience* **17**, 7–11 (2004).
- 202 11. Easterbrook, S. M. Open code for open science? *Nature Geosci* **7**, 779–781 (2014).
- 204 12. Halchenko, Y. O. & Hanke, M. Four aspects to make science open “by design” and not as an after-thought. *GigaScience* **4**, 31 (2015).
- 206 13. Nosek, B. A. *et al.* Promoting an open research culture. *Science* **348**, 1422–1425 (2015).
- 208 14. Miguel, E. *et al.* Promoting transparency in social science research. *Science (New York, NY)* **343**, 30 (2014).
- 210 15. Stodden, V., Guo, P. & Ma, Z. *How journals are adopting open data and code policies in The First Global Thematic IASC Conference on the Knowledge Commons: Governing Pooled Knowledge Resources* (2012).
- 212 16. Blischak, J. D., Davenport, E. R. & Wilson, G. A Quick Introduction to Version Control with Git and GitHub. *PLoS Comput. Biol.* **12**, e1004668 (2016).
- 214 17. Ram, K. Git can facilitate greater reproducibility and increased transparency in science. *Source Code Biol. Med.* **8**, 7 (2013).
- 216 18. Vasilevsky, N. A. *et al.* On the reproducibility of science: unique identification of research resources in the biomedical literature. *PeerJ* **1**, e148 (2013).
- 218 19. Gleeson, P., Silver, A. & Cantarelli, M. in *Encyclopedia of Computational Neuroscience* (eds Jaeger, D. & Jung, R.) 1–3 (Springer New York, 2014).
20. *INCF Software Center* <<http://software.incf.org/>>.
- 220 21. Poline, J.-B. & Kennedy, D. in *Encyclopedia of Computational Neuroscience* (eds Jaeger, D. & Jung, R.) (Springer, 2014).
- 222 22. Stodden, V. Enabling reproducible research: Open licensing for scientific innovation. *International Journal of Communications Law and Policy* **13**, 1–25 (2009).
- 224 23. Kahneman, D. A new etiquette for replication. *Soc. Psychol.* **45**, 310 (2014).
- 226 24. Schulte, E., Davison, D., Dye, T. & Dominik, C. A multi-language computing environment for literate programming and reproducible research. *Journal of Statistical Software* **46**, 1–24 (2012).
- 228 25. Gentleman, R. & Lang, D. T. Statistical analyses and reproducible research. *Journal of Computational and Graphical Statistics* (2012).
- 230 26. Delescluse, M., Franconville, R., Joucla, S., Lieury, T. & Pouzat, C. Making neurophysiological data analysis reproducible: why and how? *J. Physiol. Paris* **106**, 159–170 (2011).
- 232

27. Stevens, J.-L. R., Elver, M. & Bednar, J. A. An automated and reproducible workflow
234 for running and analyzing neural simulations using Lancet and IPython Notebook.
Front. Neuroinform. **7**, 44 (2013).
28. Eglén, S. J. *et al.* A data repository and analysis framework for spontaneous neural
236 activity recordings in developing retina. *Gigascience* **3**, 3 (2014).
29. Boettiger, C. An introduction to Docker for reproducible research. *ACM SIGOPS*
238 *Operating Systems Review* **49**, 71–79 (2015).
30. Vogels, T. P., Sprekeler, H., Zenke, F., Clopath, C & Gerstner, W. Inhibitory plastic-
240 ity balances excitation and inhibition in sensory pathways and memory networks.
242 *Science* **334**, 1569–1573 (2011).
31. Waskom, M. L., Kumaran, D., Gordon, A. M., Rissman, J. & Wagner, A. D. Fron-
244 toparietal representations of task context support the flexible control of goal-directed
cognition. *J. Neurosci.* **34**, 10743–10755 (2014).
32. Liu, Z, Golowasch, J, Marder, E & Abbott, L. F. A model neuron with activity-
246 dependent conductances regulated by multiple calcium sensors. *J. Neurosci.* **18**, 2309–
248 2320 (1998).
33. Davison, A. P. *et al.* PyNN: A Common Interface for Neuronal Network Simulators.
250 *Frontiers in Neuroinformatics* **2**, 11 (2009).
34. Cannon, R. C. *et al.* LEMS: A language for expressing complex biological models in
252 concise and hierarchical form and its use in underpinning NeuroML 2. *Frontiers in*
Neuroinformatics **8** (2014).
35. Axelrod, V. Minimizing bugs in cognitive neuroscience programming. *Front. Psy-*
254 *chol.* **5**, 1435 (2014).
36. Wilson, G. *et al.* Best practices for scientific computing. *PLoS Biol.* **12**, e1001745 (2014).
37. Tippmann, S. My digital toolbox: Nuclear engineer Katy Huff on version-control
258 systems. *Nature*. <<http://www.nature.com/news/my-digital-toolbox-nuclear-engineer-katy-huff-on-version-control-systems-1.16014>> (2014).
38. Stodden, V. The scientific method in practice: Reproducibility in the computational
260 sciences. *MIT Sloan School Working Paper 4773-10* (2010).
39. LeVeque, R. J., Mitchell, I. M. & Stodden, V. Reproducible research for scientific com-
262 puting: Tools and strategies for changing the culture. *Computing in Science and Engi-*
264 *neering* **14**, 13 (2012).