Thoughts for the Roundtable on "Data and Code Sharing in the Computational Sciences"

Hosted by the Information Society Project, Yale Law School

November 21, 2009

Randall J. LeVeque[1]

My main research area is the development of numerical methods and software, and so my thoughts primarily concern sharing of code rather than data.

There are different types of computer programs (*codes*, for short). Some programs are developed as *software packages* that are meant to be distributed in some manner and used by people other than the authors.

Here the focus is on *research codes* that are developed (perhaps using software packages) to solve a particular problem or to test out a new algorithmic idea. The issues involved in sharing research codes are somewhat different than software development and distribution.

Arguments against sharing code. Authors are naturally reluctant to share research codes for many reasons. Below are some thoughts on a few of the reasons and how they might be addressed. The points below are mostly adapted from [1], which contains a longer discussion.

- *By the time a paper is published, the authors no longer possess the code that generated the results shown.* Research codes often evolve to solve new problems, and trying to go back and reproduce earlier results may be impossible. This can even happen during the writing of a single paper with multiple examples. This of course is a problem of scientific methodology in computational science that goes well beyond the question of whether the code should be shared with others. Regardless of what will ultimately be distributed, it's important for researchers in computational science to develop better *reproducible research* habits. There are many tools available to help with this, in particular *version control software* such as Subversion, Mercurial, GIT, and others that track changes to a set of files. Although originally developed to help organize large software projects with multiple authors, such systems are very useful even for personal projects since one can keep track of the version number used for creating a particular result and revert to exactly that version at a later time if needed.

- *It is a lot of work to clean up a code to the point where someone else can even use it, let alone read it.* While ideally all published programs would be nicely structured and easily readable with ample comments, as a first step it would be valuable simply to provide and archive the working code that produced the results in a paper. This can be extremely valuable simply in order to check, for example, what parameter values were used for some part of an algorithm that were omitted from the paper. The current culture in computational science is that codes are often made publicly available only if they are being offered as software packages for others to use, with the requirements that go along with this: easy installation on multiple platforms, a nice user interface, adequate documentation, etc. It will be hard to convince people to expose their research codes to public scrutiny. Moreover, posting such a code opens the door for random people to send email asking for help getting the program to run or adapting it to solve their problems. New paradigms and expectations are required, a cultural change that will take some time to accomplish.

[1] Department of Applied Mathematics, University of Washington, Box 352420, Seattle, WA 98195-2420, rjl@uw.edu. Version of November 30, 2009.

- *A working program for solving a scientific or engineering problem is a valuable piece of intellectual property.* This is a very real concern, particularly since many research codes have taken years to develop and the authors hope to publish many more papers based on the code. Why make it globally available along with the first publication? A possible counterargument is that, as mentioned above, it is not generally easy to take someone else's research code and work with it, particularly a large code that is not designed as software. Those who want to extend it to solve a new problem will often need assistance from the original authors, if they are willing, and in return will give proper credit. If the culture changes to the point where all authors are expected to share the code used in a paper, then it would be difficult to hide the fact that the work is an extension of a prior code. Authors may also find that their work is cited more frequently if they provide code that simplifies the use of their work by others. But there are obviously legal and copyright issues to be carefully considered in regard to intellectual property rights.

- *It is impossible to share a research code if it is based on commercial, proprietary, or copyrighted software that cannot be redistributed, or that only runs on hardware most readers do not have access to.* This may be true, but generally the original research portion of the code is the authors' own work. As mentioned above, making this portion of the code available for inspection can be very useful in understanding and reproducing the results of a publication even if the reader cannot run the full code.

Why the time is ripe to encourage change. In spite of the serious objections raised above, there is a growing recognition among computational scientists and mathematicians that we need to develop better ways to validate and compare new methods that are proposed, and to facilitate building on past work. There has been an explosion of new methods proposed in the literature, many of which are implemented and tested only by their inventors. Referees and readers often could not reproduce the results of a publication from what is presented in the paper even if they were inclined to try programming it from scratch. Meaningful comparison with other methods proposed for the same problem is often nearly impossible. Frustration is growing with the current state of affairs.

Luckily the tools to facilitate change are also growing rapidly. For example, there are now many different version control systems available, most of them open source and readily available. There are also many interesting ideas and packages to facilitate *literate programming* along with reproducible research, making it easier to document codes as one goes along. These tools make it easier and more enjoyable to write research codes in a way that the author can later understand and build on, leading to personal advantages for the authors' own research. The work of graduate students and postdoctoral fellows also has a greater chance of being reused if they are encouraged to use such tools. Researchers may first adopt reproducible and literate programming techniques for these reasons, but then find that it also becomes easier to contemplate sharing the code with others.

Rapid improvements in internet speed, web interfaces, cloud computing, storage space, etc. also make it much easier to archive and share codes than ever before. Even the largest research codes are generally very small relative to the massive data sets people worry about sharing, so archiving multiple copies of codes used to produce different published results is not unreasonable. (And an advantage of version control software is that it stores only differences between versions, making it trivial to store multiple versions with little wasted storage.)

The downside of all the new ideas and tools is that there are so many possible systems, each with a certain startup cost and learning curve, that it can be daunting to know where to start. While researchers may recognize that long-term productivity would increase by adopting these tools, the startup costs take time away from current research. The rapid pace of development and evolution of these tools also means that documentation is often inadequate and that tools keep changing or disappearing, supplanted by a better approach. This is a natural part of the evolution, but until things have settled down a bit more there may be a relatively small number of researchers willing to participate in the process.

What journals might do. Some journals provide authors with the opportunity to submit supplementary material that is available online but not in the printed article. Often this consists of color versions of figures or animations that complement the printed copy in a natural way. In some cases authors are allowed or even encouraged to post the computer codes that generated the results in the paper, but this seems to be fairly rare. This is a service that more journals could be encouraged to provide. A permanent archive that would not disappear when the author changes institutions or rearranges webpages and forgets to update links would be greatly preferable to the current informal system that many authors adopt.

In this regard one journal to watch is the newly established *Mathematical Programming Computation*. In fact they go further, strongly encouraging authors to provide source code or at least executables for use by the referees in judging the paper. The description of their policies is available on the journal's webpage (mpc.zib.de). I think it is unlikely that well established journals will go this far any time soon, for fear of mass exodus of authors and reviewers. However, providing the opportunity for archiving codes and encouraging authors to take advantage of it may be viewed favorably by journals as a way of adding value to their product. Reviewers may find access to the codes very valuable in judging papers, as long as this does not place too much additional responsibility on them. If these services are available and some authors start taking advantage of them, others may be encouraged to follow.

The adoption of some sort of *Reproducible Research Standard* as proposed by Stodden[2], with a recognizable logo that could be applied to a paper, would help make readers aware of such archives and convince authors that they should follow suit.

References

[1] R. J. LeVeque. Python tools for reproducible research on hyperbolic problems. *Computing in Science and Engineering*, 11:19–27, 2009. Codes to accompany paper are available at http://www.clawpack.org/links/cise09

[2] V. Stodden. Enabling reproducible research: licensing for scientific innovation. *Int. J. Commun. Law & Policy*, 13:1–25, 2009.