Testimony of
Dr. Victoria Stodden
Columbia University

Before the House Committee on Science, Space and Technology
Subcommittee on Research

Hearing On
Scientific Integrity & Transparency
March 5, 2013

Thank you Chairman Bucshon, Ranking Member Lipinski, and other members of the Subcommittee for the opportunity to speak with you today.

I am Victoria Stodden, assistant professor of statistics at Columbia University. My research is on reproducibility of results in computational science. Reproducibility is a new challenge, brought about by advances in scientific research capability due to immense changes in technology over the last two decades. It is widely recognized as a defining hallmark of science and directly impacts the transparency and reliability of findings, and is taken very seriously by the scientific community.

Federally Funded Digital Archives are Necessary for Scientific Integrity and Accelerate Scientific Discovery

Massive computation has begun a transformation of the scientific enterprise that will finish with computation absolutely central to the scientific method. From the ability to capture data, methods, create simulations, and provide dissemination mechanisms, science has gone digital. We need federally funded archives for the scientific data and software associated with research publications. Convenient access to data and software is a necessary step in enabling reproducibility in computational science, and preservation ensures reproducibility persists. Because of their broad impact, the federal agencies that fund scientific research play a key role in facilitating the dissemination and archiving of the data and software associated with scientific findings that scientists or universities cannot play on their own. Data archives that are discipline specific and directly funded are necessary for the validation of published results, and permits others to use these resources to accelerate economic and scientific competitiveness. Openly available data and methods will maximize the downstream discoveries that could be made the information contain in the data and the know-how of methods contained in the code. This availability means curious STEM students, for example, can try their hand at replicating published results from the data and software, and learn about the science (and perhaps contribute further discoveries!).

For example other countries, such as Sweden, the U.K., the Netherlands, and Germany, are steps ahead in creating a long-term data archive for the social sciences with a standing similar to that of a national archive. This is a solution to the public good problem of access to scientific data and code. I believe separate funding is required to establish such archives in America, since using research grant funds is unpredictable and unreliable. Funding agencies need to treat this as a mandate and plan to protect data and code availability for 25 years. Archived data and code should be linked with all publications that use either of them, in order for reproducibility to be effective.

Background on the Reproducibility Issue

First, I will provide some background on the reproducibility issue. Recent technological advances have accelerated scientific research in three principal ways: increasing in our ability to collect and store vast amounts of data; increasing the computer power needed to analyze these data and perform computationally intensive science; and providing a mechanism for the rapid transmission of digital scholarly objects (such as research articles, data, or computer software) via the Internet. These three changes have revolutionized scientific research and computation is emerging as absolutely central to the scientific enterprise. In keeping with longstanding scientific norms, the scientific community has responded to these technological changes by calling for modifications of the standards of scientific communication: making available the computational aspects of the research – the code and data – that generated published scientific findings at the time of publication.[1] This is commonly called the “Reproducible Research Movement.”

The communication of scientific research was established around the goal of reproducibility – providing sufficient information to other researchers so that they are able to verify the new results. This is still the overarching goal of scientific publishing, but these technological changes are requiring us to update our standards of communication. Computational steps are typically too complex and numerous to be described in the traditional scientific publication. Researchers will need to provide both the data and the code with the computational steps as a routine part of scientific publishing. In computational science today, the published research article is rarely sufficient for the findings to be validated.

[1] The *Reproducible Research* movement: see e.g. "Reproducible Research: Addressing the Need for Data and Code Sharing in Computational Science," with Yale Roundtable Participants, Computing in Science and Engineering, 12(5) 8-13, Sep./Oct. 2010 (attached); and D. Donoho et al. “Reproducible Research in Computational Harmonic Analysis,” Computing in Science and Engineering, 11(1) 8-18, Jan 2009.

This is not to say published results are necessarily wrong, or that there is a lack of integrity on the part of scientists. What is happening is that access to the data and software is needed in order to validate and understand new scientific claims. In short, scientific communication needs to catch up with the recent technological changes in scientific research and this is not something any single researcher can do on their own. The scientific community is responding with piecemeal independent efforts however, including sessions on reproducibility at major scientific conferences and meetings, dedicated workshops and journal issues (see appendices), standards on data and code access requirements by journals, subject specific repositories with data deposit requirements, independently releasing data and code, and the development of software research tools to help with data and code sharing. These efforts, while immensely laudable since they do not result in direct career advancement or reward for the scientists responsible, are minuscule and largely token compared to the scale of change that needs to happen. Science is a peer-guided endeavor and these are the main options scientists have for creating change. A larger effort is needed and this is where the federal funding agencies come in.

The scientific community has been rocked by episodes like the case at Duke University where published results about a new statistical medical assessment test could not be verified prior to the start of clinical trials. In an embarrassing scandal, the trials were eventually cancelled after the underlying research was found contain errors. Many in the scientific community feel that these errors would have been caught much earlier, well before clinical trials had started, if the associated data and software were made routinely available when computational results are published.

Some scientists feel strongly enough about the importance of reproducible research they have self archived their data and code. For example, David Donoho's Wavelab package (http://www-stat.stanford.edu/~wavelab), my Sparselab package (http://sparselab.stanford.edu), and the papers contained in http://www.RunMyCode.org. The event at Duke University prompted the Institute of Medicine to produce a report requiring data and software submission, for validation and reproducibility purposes, to be submitted to the FDA prior to clinical trial approval. Their report, "Evolution of Translational Omics: Lessons Learned and the Path Forward," was released on March 23, 2012 at http://www.iom.edu/Reports/2012/Evolution-of-Translational-Omics.aspx . These and other efforts, while laudable, cannot come close to enabling reproducibility for all computational findings that are published today and going forwards. A funding agency level solution is needed.

Open Data and Software Accelerate Scientific Discovery, Innovation, and Economic Growth

For an individual researcher, making data and software available takes time. It takes time for professionals to archive and curate these objects, and to ensure they are properly linked to the published results. I believe that these efforts are both essential to the integrity of the scholarly record and vastly more efficient over the long run than the current method of publication (omitting the associated research data and code) since it is then much easier to ensure the accuracy of published scientific findings.

Making research data and software conveniently available also has valuable corollary effects beyond validating the original associated published results. Other researchers can use them for new research, linking datasets and augmenting results in other areas, or applying the software and methods to new research applications. These powerful benefits will accelerate scientific discovery. Benefits can also accrue to private industry. Again, data and software availability permit business to apply these methods to their own research problems, link with their own datasets, and accelerate innovation and economic growth.

Scientific research is not intended to produce viable market-ready products. It produces scientific knowledge about our world. When the data and code are made conveniently available this opens entirely new possibilities for others to commercialize and ready these discoveries for market. The discoveries and technologies are made openly available as part of publication. Raw facts are not subject to copyright (499 US 340 (1991)) and data can be readily open to catalyze innovation across scientific disciplines and across industry.

American competitiveness can only be increased as we increase the integrity of our scholarly record, and as we make access to scientific innovations, data, and their implementation broadly available to other researchers and to industry.

The Federal Agencies are Vital to Ensuring Reproducible Computational Science

Since January 18 of 2011 the National Science Foundation has required a two page “Data Management Plan” be submitted with every grant application. The Plan requested that the applicant explain how “the proposal will conform to NSF policy on the dissemination and sharing of research results.” The NSF policy referred to follows, “NSF ... expects investigators to share with other researchers, at no more than incremental cost and within a reasonable time, the data, samples, physical collections and other supporting materials created or gathered in the course of the work. It also encourages grantees to share software and inventions or otherwise act to make the innovations they embody widely useful and usable.” The National Institutes for Health has a similar policy, “We believe that data sharing is essential for expedited translation of research results into knowledge, products, and procedures to improve human health. The NIH endorses the sharing of final research data to serve these and other important scientific goals. The NIH expects and supports the timely release and sharing of final research data from NIH-supported studies for use by other researchers” (NIH grants greater than $500,000 must include a data sharing plan).

If enforced, these guidelines would help shift computational research toward reproducibility. These guidelines are generally not enforced however, and I believe this is for two reasons. One, the guidelines are not well defined in terms of what constitutes data and how it should be shared. Two, sharing is costly and the funding agency should provide mechanisms and appropriate repositories for data and code deposit. At the moment, these guidelines seem like an unfunded mandate and this should change. Federal agencies should be provided with the funds to support the open availability of data and code. This should take the form of repositories maintained by the funding agencies that can curate, preserve, and make these digital scholarly objects openly available.

The OSTP Executive Memorandum Reinforces Efforts Towards Reproducible Computational Research

On February 22, 2013, the Office of Science and Technology Policy in the Whitehouse released an executive memorandum giving federal agencies with more than $100 million in research funding six months to devise plans to facilitate open access to scientific publications and scientific data. Data is defined as “digital recorded factual material commonly accepted in the scientific community as necessary to validate research findings including data sets used to support scholarly publications.” Software is equally as important as data in validating computational science and I hope the committee understands “data” as referred to in the Executive Memorandum as including both data and software.

Standards for data sharing will vary by discipline and by research problem. Some areas, especially those that tend to make big capital investments in data collection devices such as telescopes or genome sequencing machines, are relatively advanced in terms of data sharing. Others have almost no experience with the issue. Since software is typically generated by the researchers themselves, organized methods for software sharing have not come into prominence. The different types of research funded by federal agencies may require different sharing requirements. What is clear is that they will need funds and resources to support the need for open data and software. The costs of data sharing increase markedly with the amount of curation desired, for example, meta-data, variable labeling, versioning, citation and unique identifiers, archiving, repository creation, data standards, release requirements and sequestration, among others.

Barriers to Open Data and Software: A Collective Action Problem Faces Scientists

As the scientific community reaches toward reproducibility of computational science through open data and software, there are a number of barriers. The first, mentioned earlier, is that a change in the standards of research dissemination is a classic collective action problem. One person may change their behavior but unless others follow, he or she is penalized for deviating from the norm, in this case spending time on data and code release rather than more publications (publications are recognized and rewarded in scientific careers, unlike data and code production). Federal agency action is required to break this gridlock and shift the community toward data and software sharing together. Federal agency action is also required to ensure that scientists receive credit, through citation and attribution, for their data and software contributions to science. One important step was taken by the National Science Foundation in October of 2012 when it permitted grant applicants to list research products, such as citable data and software, in biographical sketches, rather than restricting the list of contributions to publications only. More steps like this should be taken, including providing citation recommendations for data and software re-use, and expectations that use of data or software be cited or claimed as original to the author.

Not all datasets or software are worthy of the same levels of curation. Curation can be costly in terms of human hours and it stands to reason that widely used datasets and software with potentially broad applicability should receive the majority of the curation resources. Provisions can be made that curation levels be increased for data or software that is used more than expected.

There must be ways for data and software users to provide feedback on difficulties they found in re-use, and ways for these corrections and improvements to be acted upon. Such a mechanism can help establish standards for data and code curation and release within a community.

The kind of sharing infrastructure associated with data and associated with software are very different. Data is typically shared as an annotated file in repository, whereas software is much more interaction, typically shared through a version control system, perhaps with an overlay for web access such as GitHub.com (for open source software, not scientific software specifically). Reproducibility demands that we consider both the data and code associated with published computational results, and each of these have very different infrastructure needs for open accessibility. What version is used, how to manage updates and changes to data or software, what meta-data is needed, what standards apply and what documentation is expected, and how to link to the associated publication differ for data and for software.

Intellectual Property Issues in Open Scientific Data and Software

Intellectual Property Law comes to bear on both scientific data and software. Longstanding scientific norms encourage reproducible research, and scientists find it natural to openly share their findings, data, software, such that results may be understood and validated by others. Copyright adheres by default to both the scientific manuscript and software, and adhere to the original "selection and arrangement" of the data, although not to the raw facts themselves. This has resulted in efforts to apply open licensing to scholarly digital objects such that they may be shared as is natural in the scientific community: use my work, validate it, build on it, but make sure I am given appropriate citation for my contributions.[2] A broad fair use exception for scientific research that includes data and software would align Intellectual Property Law with scientific norms and needs for reproducibility, and maximize future discoveries and use of the data and code, both within the research community and within industry.

With software established as an indispensible tool in scientific discovery, a computational researcher can be faced with an unexpected conflict: conform to scientific norms of reproducibility and reveal the software that generated the results, or seek a software patent and license access to the code. Traditionally the research methodology was contained in the published report, but in the computational sciences methodology is encapsulated within a potentially patentable medium, software. It is important that the needs of science, especially those of reproducibility, remain paramount to patenting in order to promote high integrity scientific research. Making data and code available in an archives the goals for transparency and technology transfer embodied in the Bayh-Dole Act, and can be done in a way that is coordinated and harmonious between the relevant funding agencies.

[2] For discussions of open licensing for computational scientific research see e.g. V. Stodden, "The Legal Framework for Reproducible Scientific Research: Licensing and Copyright," Computing in Science and Engineering, 11(1), 2009; and see also V. Stodden, "Enabling Reproducible Research: Open Licensing for Scientific Innovation," International Journal of Communications Law and Policy, Issue 13, 2009. Available at http://ijclp.net/old_website/article.php?doc=1&issue=13_2009

Conclusion

The issue of reproducibility in computational science cuts across all manner of disciplines and research areas, from the liberal arts to engineering. The solutions are not obvious, but it is clear they can emerge with experience and action. It is imperative that data and code are made conveniently available with published research findings. Data and software availability do not, by themselves, ensure reproducibility of published computational findings, but they are an essential step toward the solution.

Thank you for the opportunity to testify this morning. I look forward to answering any questions you may have.

Setting the Default to Reproducible

Reproducibility in Computational and Experimental Mathematics

Developed collaboratively by the ICERM workshop participants[1]

Compiled and edited by the Organizers

V. Stodden, D. H. Bailey, J. Borwein, R. J. LeVeque, W. Rider, and W. Stein

Abstract

Science is built upon foundations of theory and experiment validated and improved through open, transparent communication. With the increasingly central role of computation in scientific discovery this means communicating all details of the computations needed for others to replicate the experiment, i.e. making available to others the associated data and code. The "reproducible research" movement recognizes that traditional scientific research and publication practices now fall short of this ideal, and encourages all those involved in the production of computational science – scientists who use computational methods and the institutions that employ them, journals and dissemination mechanisms, and funding agencies – to facilitate and practice really reproducible research.

This report summarizes discussions that took place during the ICERM Workshop on Reproducibility in Computational and Experimental Mathematics, held December 10-14, 2012. The main recommendations that emerged from the workshop discussions are:

1. It is important to promote a culture change that will integrate computational reproducibility into the research process.
2. Journals, funding agencies, and employers should support this culture change.
3. Reproducible research practices and the use of appropriate tools should be taught as standard operating procedure in relation to computational aspects of research.

The workshop discussions included presentations of a number of the diverse and rapidly growing set of software tools available to aid in this effort. We call for a broad implementation of these three recommendations across the computational sciences.

Introduction

The emergence of powerful computational hardware, combined with a vast array of computational software, presents unprecedented opportunities for researchers in mathematics and science. Computing is rapidly becoming the backbone of both theory and experiment, and essential in data analysis, interpretation, and inference.

Unfortunately the scientific culture surrounding computational work has evolved in ways that often make it difficult to verify findings, efficiently build on past research, or even to apply the basic tenets of the scientific method to computational procedures. Bench scientists are taught to keep careful lab notebooks documenting all aspects of the materials and

[1]For a list of participants see Appendix H or the workshop webpage `http://icerm.brown.edu/tw12-5-rcem`.
Version of February 16, 2013.

methods they use including their negative as well as positive results, but computational work is often done in a much less careful, transparent, or well-documented manner. Often there is no record of the workflow process or the code actually used to obtain the published results, let alone a record of the false starts. This ultimately has a detrimental effect on researchers' own productivity, their ability to build on past results or participate in community efforts, and the credibility of the research among other scientists and the public [6].

There is increasing concern with the current state of affairs in computational science and mathematics, and growing interest in the idea that doing things differently can have a host of positive benefits that will more than make up for the effort required to learn new work habits. This research paradigm is often summarized in the computational community by the phrase "reproducible research." Recent interest and improvements in computational power have led to a host of new tools developed to assist in this process. At the same time there is growing recognition among funding agencies, policy makers, and the editorial boards of scientific journals of the need to support and encourage this movement. [2]. A number of workshops have recently been held on related topics, including a Roundtable at Yale Law School [14] a workshop as part of the Applied Mathematics Perspectives 2011 conference [2, 7], and several minisymposia at other conferences, including SIAM Conferences on Computational Science and Engineering 2011 and ICIAM 2011.

The ICERM Workshop on Reproducibility in Computational and Experimental Mathematics, held December 10-14, 2012, provided the first opportunity for a broad cross section of computational scientists and mathematicians, including pure mathematicians who focus on experimental mathematics or computer-assisted proofs, to discuss these issues and brainstorm ways to improve on current practices. The first two days of the workshop focused on introducing the themes of the meeting and discussing policy and cultural issues. In addition to introductory talks and open discussion periods, there were panels on funding agency policies and on journal and publication policies. The final three days featured many talks on software tools that help achieve reproducibility and other more technical topics in the mornings. Afternoons were devoted to breakout groups discussing specific topics in more depth, which resulted in recommendations and other outcomes. Breakout group topics included: reproducibility tools, funding policies, publication policies, numerical reproducibility, taxonomy of terms, reward structure and cultural issues, and teaching reproducible research techniques. [3] We also held a tutorial on version control the day before the official start of the workshop. [4]

Both in the workshop and in this report the terms "reproducible research" and "reproducibility" most often refer to the ability to recreate computational results from the data and code used by the original researcher [11]. This is related to but distinct from both the notions of "numerical reproducibility" of computational results, referring to when the same program may give different results due to hardware or compiler issues, particular in the context of parallel computing, and "repeatability," when an experiment is conducted independently from first principles. A taxonomy of reproducibility concepts is developed in Appendix A and a discussion of numerical reproducibility appears in Appendix B.

[2]See National Science Foundation Data Management Plan `http://`, ACM Publications Policy `http://`

[3]See the workshop program at `http://icerm.brown.edu/tw12-5-rcem`.

[4]`http://icerm.brown.edu/tw12-5-rcem-tutorial`.

About this document

This document reports on the three main recommendations emerging from the workshop discussions:

1. It is important to promote a culture change that will integrate computational reproducibility into the research process.
2. Journals, funding agencies, and employers should support this culture change.
3. Reproducible research practices and the use of appropriate tools should be taught as standard operating procedure in relation to computational aspects of research.

The recommendations are each discussed in turn in the three sections of this document, and we include five appendices that develop important topics in further detail. Besides the appendices mentioned above on taxonomy and numerical reproducibility, there are appendices on best practices for publishing research, the state of reproducibility in experimental mathematics, and tools to aid in reproducible research. An initial draft of this document was presented to participants and discussed on the final day of the workshop, and participants were able to give input on the final draft before submission.

In addition to this document, a number of other products emerged from the workshop. Video of the talks is available at `http://icerm.brown.edu/video_archive`, and numerous topical references were collected on the workshop wiki [5]. The workshop webpage and the wiki also contain participant thought pieces, slides from the talks, and breakout group reports. Readers are invited to contribute to the wiki. A snapshot of the wiki is appended at the end of the report as Figure 1.

1. Changing the Culture and Reward Structure

For reproducibility to be fostered and maintained, workshop participants agreed that cultural changes need to take place within the field of computationally based research that instill the open and transparent communication of results as a default. Such a mode will increase productivity — less time wasted in trying to recover output that was lost or misplaced, less time wasted trying to double-check results in the manuscript with computational output, and less time wasted trying to determine whether other published results (or even their own) are truly reliable. Open access to any data used in the research and to both primary and auxiliary source code also provides the basis for research to be communicated transparently creating the opportunity to build upon previous work, in a similar spirit as open software provided the basis for Linux. Code and data should be made available under open licensing terms as discussed in Appendix F. [9] This practice enables researchers both to benefit more deeply from the creative energies of the global community and to participate more fully in it. Most great science is built upon the discoveries of preceding generations and open access to the data and code associated with published computational science allows this tradition to continue. Researchers should be encouraged to recognize the potential benefits of openness and reproducibility.

[5]Available at `http://is.gd/RRlinks`, see Figure 1.

It is also important to recognize that there are costs and barriers to shifting to a practice of reproducible research, particularly when the culture does not recognize the value of developing this new paradigm or the effort that can be required to develop or learn to use suitable tools. This is of particular concern to young people who need to earn tenure or secure a permanent position. To encourage more movement towards openness and reproducibility, it is crucial that such work be acknowledged and rewarded. The current system, which places a great deal of emphasis on the number of journal publications and virtually none on reproducibility (and often too little on related computational issues such as verification and validation), penalizes authors who spend extra time on a publication rather than doing the minimum required to meet current community standards. Appropriate credit should given for code and data contributions including an expectation of citation. Another suggestion is to instantiate yearly award from journals and/or professional societies, to be awarded to investigators for excellent reproducible practice. Such awards are highly motivating to young researchers in particular, and potentially could result in a sea change in attitudes. These awards could also be cross-conference and journal awards; the collected list of award recipients would both increase the visibility of researchers following good practices and provide examples for others.

More generally, it is unfortunate that software development and data curation are often discounted in the scientific community, and programming is treated as something to spend as little time on as possible. Serious scientists are not expected to carefully test code, let alone document it, in the same way they are trained to properly use other tools or document their experiments. It has been said in some quarters that writing a large piece of software is akin to building infrastructure such as a telescope rather than a creditable scientific contribution, and not worthy of tenure or comparable status at a research laboratory. This attitude must change if we are to encourage young researchers to specialize in computing skills that are essential for the future of mathematical and scientific research. We believe the more proper analog to a large scale scientific instrument is a supercomputer, whereas software reflects the intellectual engine that makes the supercomputers useful, and has scientific value beyond the hardware itself. Important computational results, accompanied by verification, validation, and reproducibility, should be accorded with honors similar to a strong publication record [7].

Several tools were presented at the workshop that enable users to write and publish documents that integrate the text and figures seen in reports with code and data used to generate both text and graphical results, such as IPython, Sage notebooks, Lepton, knitr, and Vistrails. Slides for these talks are available on the wiki [1] and Appendix E discusses these and other tools in detail.

The following two sections and the appendices outline ideas from the workshop on ways in which journals, funding agencies, and employers can support reproducibility.

2. Funding Agencies, Journals, Employers Should Support This Change

Incentives in scholarly research are influenced by three main sources, the funding agency, dissemination processes such as journals, and employers such as those on tenure committees and lab managers. The workshop discussions mentioned the role of each of them in shifting to a culture of reproducible computational research.

The Role of Funding Agency Policy

Workshop participants suggested that funding sources, both government agencies and private foundations, consider establishing some reasonable standards for proposals in the arena of mathematical and computational science. If such standards become common among related agencies this would significantly simplify the tasks involved in both preparing and reviewing proposals, as well as supporting a culture change toward reproducible computational research.

For example, workshop participants recommend that software and data be "open by default" unless it conflicts with other considerations. Proposals involving computational work might be required to provide details such as:

- Extent of computational work to be performed.
- Platforms and software to be utilized.
- Reasonable standards for dataset and software documentation, including reuse (some agencies already have such requirements [8]).
- Reasonable standards for persistence of resulting software and dataset preservation and archiving.
- Reasonable standards for sharing resulting software among reviewers and other researchers.

In addition, we suggest that funding agencies might add "reproducible research" to the list of specific examples that proposals could include in their requirements such as "Broader Impact" statements. Software and dataset curation should be explicitly included in grant proposals and recognized as a scientific contribution by funding agencies. Templates for data management plans could be made available that include making software open and available, perhaps by funding agencies, or by institutional archiving and library centers. [6]

Participants also suggested that statements from societies and others on the importance of reproducibility could advance the culture change. In addition, funding agencies could provide support for training workshops on reproducibility, and cyberinfrastructure for reproducibility at scale, for both large projects and long-tail research efforts. Funding agencies are key to the promotion of a culture that embraces reproducible research, due to their central importance in the research process. We turn to journals next, and then employers.

The Role of Journal Policy

There is a need to produce a set of "best practices" for publication of computational results i.e. any scientific results in which computation plays a role, for example in empirical research, statistical analysis, or image processing. We recommend that a group representing several professional societies in the mathematical sciences be formed to develop a set of best practices for publication of research results. Such guidelines would be useful

[6]For examples see http://scholcomm.columbia.edu/data-management/data-management-plan-templates/, http://www2.lib.virginia.edu/brown/data/NSFDMP.html

to the editorial boards of many journals, as well as to authors, editors, and referees who are concerned about promoting reproducibility. Best practices may be tailored to different communities, but one central concern, for which there was almost unanimous agreement by the workshop participants, is the need for full disclosure of salient details regarding software and data use. This should include specification of the dataset used (including URL and version), details of the algorithms employed (or references), the hardware and software environment, the testing performed, etc., and would ideally include availability of the relevant computer code with a reasonable level of documentation and instructions for repeating the computations performed to obtain the results in the paper. [7].

There is also a need for better standards on how to include citations for software and data in the references of a paper, instead of inline or as footnotes. Proper citation is essential both for improving reproducibility and in order to provide credit for work done developing software and producing data, which is a key component in encouraging the desired culture change [7].

Workshop participants agreed that it is important that a set of standards for reviewing papers in the computational arena be established. Such a set of standards might include many or all of the items from a "best practices" list, together with a rational procedure for allowing exceptions or exclusions. Additionally, provisions are needed to permit referees to obtain access to auxiliary information such as computer codes or data, and the ability to run computational tests of the results in submitted papers, if desired.

Different journals may well adopt somewhat different standards of code and data disclosure and review [12], but it is important that certain minimal standards of reproducibility and rigor be maintained in all refereed journal publications. Along these lines, it may be desirable for the computational claims of a manuscript to be verifiable at another site such as RunMyCode.org, or on another computer system with a similar configuration.

Some related issues in this arena include: (a) anonymous versus public review, (b) persistence (longevity) of code and data that is made publicly available, and (c) how code and data can be "watermarked," so that instances of uncited usage (plagiarism) can be detected and provenance better established (d) how to adjudicate disagreements that inevitably will arise.

Very rigorous verification and validity testing, along with a full disclosure of computational details, should be required of papers making important assertions, such as the computer-assisted proof of a long-standing mathematical result, new scientific breakthroughs, or studies that will be the basis for critical policy decisions [13].

Proper consideration of openness constraints can enable a larger community to participate in the goals of reproducible research. This can include issues such as copyright, patent, medical privacy, personal privacy, security, and export issues. This is discussed further in Appendix F.

It was recognized that including such details in submitted manuscripts (or, at the least, in supplementary materials hosted by the journal) is a significant departure from established practice, where few such details are typically presented. But these changes will be required if the integrity of the computational literature is to be maintained. Computational approaches have become central to science and cannot be completely documented and

[7]See for example `http://software.ac.uk/so-exactly-what-software-did-you-use`

transparent without the full disclosure of computational details. Appendix D contains the full list of workshop suggestions.

The Role of Employers and Research Managers

The third source of influence on the research process stems from employers – tenure and promotion committees and research managers at research labs. Software and dataset contributions, as described in the previous two subsections, should be rewarded as part of expected research practices. Data and code citation practices should be recognized and expected in computational research. Prizes for reproducible research should also be recognized in tenure and promotion decisions.

Institutional libraries can also play a role in supporting a culture change toward reproducible research. As mentioned above, they can and do provide template data management plans, but they are also highly experienced in archiving, stewardship and dissemination of scholarly objects. Greater coordination between departments and the institute's library system could help provide the support and resources necessary to manage and maintain digital scholarly output, including datasets and code [4].

3. Teaching Reproducibility Skills

Proficiency in the skills required to carry out reproducible research in the computational sciences should be taught as part of the scientific methodology, along with teaching modern programming and software engineering techniques. This should be a standard part of any computational research curriculum, just as experimental or observational scientists are taught to keep a laboratory notebook and follow the scientific method. Adopting appropriate tools (see Appendix E) should be encouraged, if not formally taught, during the training and mentoring of students and postdoctoral fellows. Without a change in culture and expectations at this stage, reproducibility will likely never enter the mainstream of mathematical and scientific computing.

We see at least five separate ways in which these skills can be taught: full academic courses, incorporation into existing courses, workshops and summer schools, online courses or self-study materials, and last but certainly not least, teaching-by-example on the part of mentors.

Although a few full-scale courses on reproducibility have been attempted (see the wiki for links), we recognize that adding a new course to the curriculum or the students' schedules is generally not feasible. It seems more effective as well as more feasible to incorporate teaching the tools and culture of reproducibility into existing courses on various subjects, concentrating on the tools most appropriate for the domain of application. For example, several workshop participants have taught classes in which version control is briefly introduced and then students are required to submit homework by pushing to a version control repository as a means of encouraging this habit.

A list of potential curriculum topics on reproducibility are listed in Appendix G. Ideally, courseware produced at one institution should be shared with others under an appropriate open license.

Conclusion

The goal of computational reproducibility is to provide a solid foundation to computational science, much like a rigorous proof is the foundation of mathematics. Such a foundation permits the transfer of knowledge that can be understood, implemented, evaluated, and used by others. This reports discusses the efforts of participants and organizers of the ICERM workshop on "Reproducibility in Computational and Experimental Mathematics" to formulate steps toward the ideal of reproducible computational research. We identified three key recommendations emerging from workshop discussions, calling for a culture change toward reproducible research, mapping roles for funding agencies, journals, and employers to support this change, and emphasizing that methods and best practices for reproducible research must be taught. We also include detailed appendices on related issues that arose in the workshop discussions, including a taxonomy of terms, numerical reproducibility, best practices for publishing reproducible research, a summary of the state of experimental mathematics, and tools to aid in reproducible research. To capture the phenomenal level of engagement by workshop participants, we collate further information, including their talk slides, thought pieces, and further references on the workshop wiki.

References

[1] Applied mathematics perspectives 2011 workshop on reproducible research: Tools and strategies for scientific computing. http://wiki.stodden.net/AMP2011/, 2012.

[2] Applied mathematics perspectives 2011 workshop on reproducible research: Tools and strategies for scientific computing. http://stodden.net/AMP2011/, 2009.

[3] David H. Bailey, Roberto Barrio, and Jonathan M. Borwein. High precision computation: Mathematical physics and dynamics. *Applied Mathematics and Computation*, 218:10106–10121, 2012.

[4] Christine Borgman. Research data, reproducibility, and curation. In *Digital Social Research: A Forum for Policy and Practice, Oxford Internet Institute Invitational Symposium*, 2012.

[5] Jonathan M. Borwein and David H. Bailey. *Mathematics by Experiment: Plausible Reasoning in the 21st Century*. A K Peters (Taylor and Francis), Natick, MA, 2008.

[6] David Donoho, Arian Maleki, Morteza Shahram, Victoria Stodden, and Inam Ur Rahman. Reproducible research in computational harmonic analysis. *Computing in Science and Engineering*, 11, January 2009.

[7] Randall LeVeque, Ian Mitchell, and Victoria Stodden. Reproducible research for scientific computing: Tools and strategies for changing the culture. *Computing in Science and Engineering*, pages 13–17, July 2012.

[8] Brian Matthews, Brian McIlwrath, David Giaretta, and Ester Conway. The significant properties of software: A study. http://www.jisc.ac.uk/media/documents/programmes/preservation/spsoftware_report_redacted.pdf, 2008.

[9] Victoria Stodden. Enabling reproducible research: Licensing for scientific innovation. *International Journal of Communications Law and Policy*, pages 1–25, 2009.

[10] Victoria Stodden. The legal framework for reproducible scientific research. *Computing in Science and Engineering*, 11, January 2009.

[11] Victoria Stodden. Trust your science? open your data and code. *Amstat News*, 2011.

[12] Victoria Stodden, Peixuan Guo, and Zhaokun Ma. 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.

[13] Greg Wilson, D. A. Aruliah, C. Titus Brown, Neil P. Chue Hong, Matt Davis, Steven H. D. Haddock Richard T. Guy, Katy Huff, Ian M. Mitchell, Mark Plumbley, Ben Waugh, Ethan P. White, and Paul Wilson. Best practices for scientific computing.

[14] Yale law school, data and code sharing roundtable. `http://www.stanford.edu/~vcs/Conferences/RoundtableNov212009/`, 2009.

Appendices

These appendices contain some additional material arising from workshop discussions. We have avoided including long lists of references in these appendices. Instead, many links have been collected and categorized on the workshop wiki, which can be referred to for more examples, additional tools, articles, editorials, etc.[8]

A Terminology and Taxonomy

The terms “reproducible research” and “reproducibility” are used in many different ways to encompass diverse aspects of the desire to make research based on computation more credible and extensible. Lively discussion over the course of the workshop has led to some suggestions for terminology, listed below. We encourage authors who use such terms in their work to clarify what they mean in order to avoid confusion.

There are several possible levels of reproducibility, and it seems valuable to distinguish between the following:

- *Reviewable Research.* The descriptions of the research methods can be independently assessed and the results judged credible. (This includes both traditional peer review and community review, and does not necessarily imply reproducibility.)
- *Replicable Research.* Tools are made available that would allow one to duplicate the results of the research, for example by running the authors' code to produce the plots shown in the publication. (Here tools might be limited in scope, e.g., only essential data or executables, and might only be made available to referees or only upon request.)
- *Confirmable Research.* The main conclusions of the research can be attained independently without the use of software provided by the author. (But using the complete description of algorithms and methodology provided in the publication and any supplementary materials.)
- *Auditable Research.* Sufficient records (including data and software) have been archived so that the research can be defended later if necessary or differences between independent confirmations resolved. The archive might be private, as with traditional laboratory notebooks.
- *Open or Reproducible Research.* Auditable research made openly available. This comprised well-documented and fully open code and data that are publicly available that would allow one to (a) fully audit the computational procedure, (b) replicate and also independently reproduce the results of the research, and (c) extend the results or apply the method to new problems.

Other terms that often arise in discussing reproducibility have specific meanings in computational science. In particular the widely-used acronym V&V (verification & valida-

[8]For the wiki see `http://icerm.brown.edu/tw12-5-rcem-wiki.php` or `http://is.gd/RRlinks`

tion) makes it difficult to use “verify” or “validate” more generally. These terms are often defined as follows:

- *Verification.* Checking that the computer code correctly solves the mathematical problem it claims to solve. (Does it solve the equation right?)
- *Validation.* Checking that the results of a computer simulation agree with experiments or observations of the phenomenon being studied. (Does it solve the right equation?)

The term “Uncertainty Quantification (UQ)” is also commonly used in computational science to refer to various approaches to assessing the effects of all of then uncertainties in data, models, and methods on the final result, which is often then viewed as a probability distribution on a space of possible results rather than a single answer. This is an important aspect of reproducibility in situations where exact duplication of results cannot be expected for various reasons.

The *provenance* of a computational result is a term borrowed from the art world, and refers to a complete record of the source of any raw data used, the computer programs or software packages employed, etc. The concept of provenance generally includes a record of changes that the dataset or software has undergone.

B Numerical Reproducibility

Numerical round-off error and numerical differences are greatly magnified as computational simulations are scaled up to run on highly parallel systems. As a result, it is increasingly difficult to determine whether a code has been correctly ported to a new system, because computational results quickly diverge from standard benchmark cases. And it is doubly difficult for other researchers, using independently written codes and distinct computer systems, to reproduce published results.

One solution is to utilize some form of higher precision arithmetic, such as Kahan’s summation or “double-double” arithmetic. In many cases, such higher precision arithmetic need only be used in global summations or other particularly sensitive operations, so that the overall runtime is not greatly increased. Such measures have dramatically increased reproducibility in various codes, ranging from climate simulations to computational physics applications [3].

But it is clear that this solution will not work for all applications. Other approaches include interval arithmetic (which potentially can provide provable bounds on final results), affine arithmetic (which works better than interval arithmetic in some cases), and also some proposed new tools, currently under development at U.C. Berkeley, that can pin down numerically sensitive sections of code and take corrective actions. in any event, additional study and research is in order. Certainly the available tools for high-precision computation need to be significantly refined so as to be usable and highly reliable for a wide range of users.

It is clear that these issues must be addressed with greater diligence by authors of all manuscripts presenting results of numeric computations. They must be more careful to state exactly what levels of numeric precision (32-bit, 64-bit or higher precision) have been used, and to present evidence that their selected precision is adequate to achieve a reasonable level of reproducibility in their results.

One of the foundations of reproducibility is how to deal with (and set standards for) difficulties such as numerical round-off error and numerical differences when a code is run on different systems or different numbers of processors. Such difficulties are magnified as problems are scaled up to run on very large, highly parallel systems.

Computations on a parallel computer system present particularly acute difficulties for reproducibility since, in typical parallel usage, the number of processors may vary from run to run. Even if the same number of processors is used, computations may be split differently between them or combined in a different order. Since computer arithmetic is not commutative, associative, or distributive, achieving the same results twice can be a matter of luck. Similar challenges arise when porting a code from one hardware or software platform to another.

The IEEE Standards for computer arithmetic resulted in significant improvements in numerical reproducibility on single processors when they were introduced in the 1970s. Some work is underway on extending similar reproducibility to parallel computations, for example in the Intel Mathematics Kernel Library (MKL), which can use used to provide parallel reproducibility for mathematical computations.

Additional issues in this general arena include: (a) floating-point standards and whether they being adhered to on the platform in question, (b) changes that result from different levels of optimization, (c) changes that result from employing library software, (d) verification of results, and (e) fundamental limits of numerical reproducibility, what are reasonable expectations and what are not.

The foundation of numerical reproducibility is also grounded in the computing hardware and in the software stack. Studies on silent data corruption (SDC) have documented SDC in field testing, as discussed in some of the references on the wiki.

Field data on supercomputer DRAM memory failures have shown that advanced error correcting codes (ECC) are required and technology roadmaps suggest this problem will only get worse in the coming years. Designing software that can do some or all of identification, protection, and correction will become increasingly important. Still, there is much work being done to quantify the problem on current and next generation hardware and approaches to addressing it. Several United States and international governmental reports have been produced on the need for, outlining ongoing research in, and proscribing roadmaps.

These foundational components set a limit to the achievable reproducibility and make us aware that we must continually assess just how reproducible our methods really are.

C The State of Experimental Mathematics

Automatic theorem proving has now achieved some truly impressive results such as fully formalized proofs of the Four color theorem and the Prime number theorem. While such tools currently require great effort, one can anticipate a time in the distant future when all truly consequential results are so validated.

The emerging discipline of experimental mathematics, namely the application of high-performance computing technology to explore research questions in pure and applied mathematics, raises numerous issues of computational reproducibility [5]. Experimental mathematics research often press the state-of-the-art in very high precision computation (often hundreds or thousands of digits), symbolic computation, graphics and parallel computation. There is a need to carefully document algorithms, implementation techniques, computer environments, experiments and results, much as with other forms of computation-based research. Even more emphasis needs to be placed on aspects of such research that are unique to this discipline: (a) Are numeric precision levels (often hundreds or even thousands of digits) adequate for the task at hand? (b) What independent consistency checks have been employed to validate the results? (c) If symbolic manipulation software was employed (e.g., Mathematica or Maple), exactly which version was used?[9] (c) Have numeric spot-checks been performed to check derived identities and other results? (d) Have symbolic manipulations been validated, say by using two different symbolic manipulation packages?

Such checks are often required, because even the best symbolic and numeric computation packages have bugs and limitations, which bugs are often only exhibited when doing state-of-the-art research computations. Workshop participants identified numerous instances of such errors in their work, underscoring the fact that one cannot place unquestioned trust in such results.

D Best Practices for Publishing Research

Publishing can take many forms – traditional journal publication is one avenue but other electronic options are increasingly being used. Traditional publications are also frequently complemented by "supplementary materials" posted on a journal's website or in other archival-quality data or code repositories.

A number of suggestions were made regarding best practices for publications of research results. To aid in reproducibility, the available materials should ideally contain:

- A precise statement of assertions to be made in the paper.
- A statement of the computational approach, and why it constitutes a rigorous test of the hypothesized assertions.
- Complete statements of, or references to, every algorithm employed.
- Salient details of auxiliary software (both research and commercial software) used in the computation.
- Salient details of the test environment, including hardware, system software and the number of processors utilized.

[9]Indeed, one needs to know which precise functions were called, with what parameter values and environmental settings?

- Salient details of data reduction and statistical analysis methods.
- Discussion of the adequacy of parameters such as precision level and grid resolution.
- Full statement (or at least a valid summary) of experimental results.
- Verification and validation tests performed by the author(s).
- Availability of computer code, input data and output data, with some reasonable level of documentation.
- Curation: where are code and data available? With what expected persistence and longevity? Is there a site for site for future updates, e.g. a version control repository of the code base?
- Instructions for repeating computational experiments described in the paper.
- Terms of use and licensing. Ideally code and data "default to open", i.e. a permissive re-use license, if nothing opposes it.
- Avenues of exploration examined throughout development, including information about negative findings.
- Proper citation of all code and data used, including that generated by the authors.

Several publications have adopted some requirements for reproducibility (e.g., Biostatistics, TOMS, IPOL, or conferences such as SIGMOD). In addition to those discussed in the main article, some other recommendations arose in discussions and break-out groups to change the culture in relation to reproducibility in publications. Journals or other publications could offer certifications of reproducibility that would kite-mark a paper satisfying certain requirements, as done by the journal Biostatistics, for example. Certification could also come from an independent entity such as RunMyCode.org. Journals could also create reproducible overlay issues for journals that collect together reproducible papers. Linking publications to sites where code and data are hosted will help shift toward reproducible research. For example, the *SIAM Journal on Imaging Science* provides cross-referencing with the peer-reviewed journal *Image Processing On Line (IPOL)* and encourage authors to submit software to IPOL. Other sites such as RunMyCode.org or Wakari might be used in a similar way. Finally, all code and data should be labeled with author information.

E Tools to aid in reproducible research

A substantial portion of the workshop focused on tools to aid in replicating past computational results (by the same researcher and/or by others) and to assist in tracking the provenance of results and the workflow used to produce figures or tables, along with discussion of the policy issues that arise in connection with this process.

Some tools are aimed at easing literate programming and publishing of computer code, either as commented code or in notebook environments. Other tools help capture the provenance of a computational result and/or the complete software software environment used to run a code. Version control systems have been around for decades, but new tools facilitate the use of version control both for collaborative work and for archiving projects along with the complete history. Collaborative high performance computational tools, while still infrequently used, now allow researchers at multiple locations to explore climate or ocean flow models in real time. Less sophisticated but instructive applets generated in geometry or computer algebra packages can easily be shared and run over the internet. We gives an overview of tools in these various categories. A list of links to these tools and many others can also be found on the wiki.

Literate programming, authoring, and publishing tools. These tools enable users to write and publish documents that integrate the text and figures seen in reports with code and data used to generate both text and graphical results. In contrast to notebook-based tools discussed below, this process is typically not interactive, and requires a separate compilation step. Tools that enable literate programming include both programming-language-specific tools such as WEB, Sweave, and knitr, as well as programming-language-independent tools such as Dexy, Lepton, and noweb. Other authoring environments include SHARE, Doxygen, Sphinx, CWEB, and the Collage Authoring Environment.

Tools that define and execute structured computation and track provenance. Provenance refers to the tracking of chronology and origin of research objects, such as data, source code, figures, and results. Tools that record provenance of computations include VisTrails, Kepler, Taverna, Sumatra, Pegasus, Galaxy, Workflow4ever, and Madagascar.

Integrated tools for version control and collaboration. Tools that track and manage work as it evolves facilitate reproducibility among a group of collaborators. With the advent of version control systems (e.g., Git, Mercurial, SVN, CVS), it has become easier to track the investigation of new ideas, and collaborative version control sites like Github, Google Code, BitBucket, and Sourceforge enable such ideas to be more easily shared. Furthermore, these web-based systems ease tasks like code review and feature integration, and encourage collaboration.

Tools that express computations as notebooks. These tools represent sequences of commands and calculations as an interactive worksheet with pretty printing and integrated displays, decoupling content (the data, calculations) from representation (PDF, HTML, shell console), so that the same research content can be presented in multiple ways. Examples include both closed-source tools such as MATLAB (through the publish and app features), Maple, and Mathematica, as well as open-source tools such as IPython, Sage, RStudio (with knitr), and TeXmacs.

Tools that capture and preserve a software environment. A major challenge in reproducing computations is installing the prerequisite software environment. New tools make it possible to exactly capture the computational environment and pass it on to someone who wishes to reproduce a computation. For instance, VirtualBox, VMWare, or Vagrant can be used to construct a virtual machine image containing the environment. These images are typically large binary files, but a small yet complete text description (a recipe to create the virtual machine) can be stored in their place using tools like Puppet, Chef, Fabric, or shell scripts. Blueprint analyzes the configuration of a machine and outputs its text description. ReproZip captures all the dependencies, files and binaries of the experiment, and also creates a workflow specification for the VisTrails system in order to make the execution and exploration process easier. Application virtualization tools, such as CDE (Code, Data, and Environment), attach themselves to the computational process in order to find and capture software dependencies.

Computational environments can also be constructed and made available in the cloud, using Amazon EC2, Wakari, RunMyCode.org and other tools. VCR, or Verifiable Computational Research, creates unique identifiers for results that permits their reproduction in the cloud.

Another group are those tools that create an integrated software environment for research that includes workflow tracking, as well as data access and version control. Examples include Synapse/clearScience and HUBzero including nanoHUB.

Interactive theorem proving systems for verifying mathematics and computation. "Interactive theorem proving", a method of formal verification, uses computational proof assistants to construct formal axiomatic proofs of mathematical claims. Examples include coq, Mizar, HOL4, HOL Light, ProofPowerHOL, Isabelle, ACL2, Nuprl, Veritas, and PVS. Notable theorems such as the Four Color Theorem have been verified in this way, and Thomas Hales's Flyspeck project, using HOL Light and Isabelle, aims to obtain a formal proof of the Kepler conjecture. Each one of these projects produces machine-readable and exchangeable code that can be integrated in to other programs. For instance, each formula in the web version of NIST's authoritative Digital Library of Mathematical Functions may be downloaded in TeX or MathML (or indeed as a PNG image) and the fragment directly embedded in an article or other code. This dramatically reduces chances of transcription error and other infelicities being introduced.

While we have organized these tools into broad categories, it is important to note that users often require a collection of tools depending on their domain and the scope of reproducibility desired. For example, capturing source code is often enough to document algorithms, but to replicate results on high-performance computing resources, for example, the build environment or hardware configuration are also important ingredients. Such concerns have been categorized in terms of the depth, portability, and coverage of reproducibility desired.

The development of software tools enabling reproducible research is a new and rapidly growing area of research. We think that the difficulty of working reproducibly will be significantly reduced as these and other tools continue to be adopted and improved. The scientific, mathematical, and engineering communities should encourage the development of such tools by valuing them as significant contributions to scientific progress.

F Copyright and licensing

The copyright issue is pervasive in software and can affect data, but solutions have been created through open licensing and public domain dedication. Copyright adhere to all software and scripts as an author types, and care must be taken when sharing these codes that permission is given for others to copy, reproduce, execute, modify and otherwise use the code. For reproducibility of scientific findings an attribution-only license is recommended, such as the Apache, MIT, or Modified BSD license [10]. Copyright does not adhere to raw facts, and so the raw numbers in a dataset do not fall under copyright. But datasets can have copyright barriers to reuse if they contain “original selection and arrangement” of the data, and for this reason dedication to the public domain is suggested using the Creative Commons CC0 license for example [9]. In addition, dataset authors can provide a citation suggestion for others who use the dataset. These steps will permit shared code and data to be copied, run on other systems, modified, and results replicated, and help encourage a system of citation for both code and data.

Limits to disclosure of data also include issues such as release of individual data for medical records, census data, and for example Google search data is not publicly shareable except in the aggregate. Of course “the aggregate” is defined differently in each domain. We also recognize that legal standards in different jurisdictions (e.g. European Union, United States, Japan) can vary and that each individual needs to apprise themselves of the most substantial differences.

The algorithms embodied in software can be patentable and the author or institution may choose to seek a patent. Patents create a barrier to access and it is recommended to license the software to commercial entities through a traditional patent, and permit open access for research purposes. If patents restrict access to code this can inhibit reproducibility, access to methods, and scientific progress. Within the commercial sphere, there is a need for avenues to allow audit such as non-disclosure agreements (NDA) and independent agents for auditing similar to financial audits. Public disclosure of algorithms and code can prevent patenting by others, and ensure that such scholarly objects remain in the public domain.

G The teaching and training of reproducibility skills

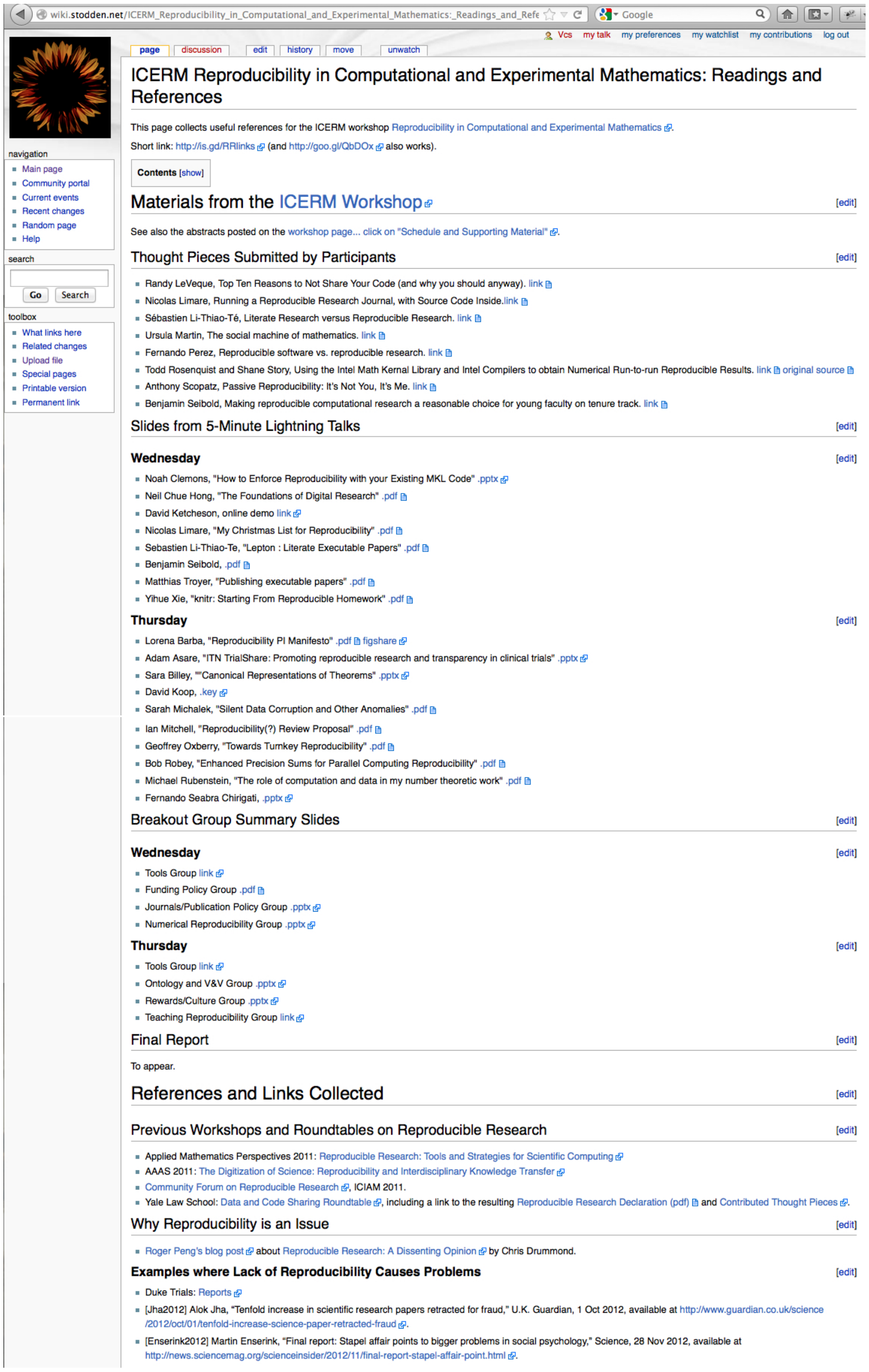
The breakout group on Teaching identified the following topics as ones that instructors might consider including in a course on scientific computing with an emphasis on reproducibility. Some subset of these might be appropriate for inclusion in many other courses.

- version control and use of online repositories,
- modern programming practice including unit testing and regression testing,
- maintaining “notebooks” or “research compendia”,
- recording the provenance of final results relative to code and/or data,
- numerical / floating point reproducibility and nondeterminism,
- reproducibility on parallel systems,
- dealing with large datasets,
- dealing with complicated software stacks and use of virtual machines,
- documentation and literate programming,
- IP and licensing issues, proper citation and attribution.

The fundamentals/principles of reproducibility can and should taught already at the undergraduate level. However, care must be taken to not overload the students with technicalities whose need is not clear from the tasks assigned to them. Collaborative projects/assignments can be a good motivation.

H Workshop Participants

Aron Ahmadia, Dhavide Aruliah, Jeremy Avigad, David Bailey, Lorena Barba, Blake Barker, Sara Billey, Ron Boisvert, Jon Borwein, Brian Bot, Andre Brodtkorb, Neil Calkin, Vincent Carey, Ryan Chamberlain, Neil Chue Hong, Timothy Clem, Noah Clemons, Constantine Dafermos, Andrew Davison, Nathan DeBardeleben, Andrew Dienstfrey, David Donoho, Katherine Evans, Sergey Fomel, Juliana Freire, James Glimm, Sigal Gottlieb, Josh Greenberg, Tom Hales, Nicolas Hengartner, David Ketcheson, Matt Knepley, David Koop, Randall LeVeque, Nicolas Limare, Elizabeth Loew, Ursula Martin, Bruce McHenry, Chris Mentzel, Sarah Michalak, Ian Mitchell, Victor Moll, Hatef Monajemi, Akil Narayan, Peter Norvig, Travis Oliphant, Peter Olver, Geoffrey Oxberry, Fernando Perez, Konrad Polthier, Bill Rider, Robert Robey, Todd Rosenquist, Michael Rubinstein, Thomas Russell, Fernando Seabra Chirigati, Li-Thiao-Te Sebastien, Benjamin Seibold, Loren Shure, Philip Stark, William Stein, Victoria Stodden, Benjamin Stubbs, Andrew Sutherland, Matthias Troyer, Jan Verschelde, Stephen Watt, Greg Wilson, Carol Woodward, Yihui Xie.



ICERM Reproducibility in Computational and Experimental Mathematics: Readings and References

This page collects useful references for the ICERM workshop Reproducibility in Computational and Experimental Mathematics.

Short link: http://is.gd/RRlinks (and http://goo.gl/QbDOx also works).

Contents [show]

Materials from the ICERM Workshop [edit]

See also the abstracts posted on the workshop page... click on "Schedule and Supporting Material".

Thought Pieces Submitted by Participants [edit]

- Randy LeVeque, Top Ten Reasons to Not Share Your Code (and why you should anyway). link
- Nicolas Limare, Running a Reproducible Research Journal, with Source Code Inside.link
- Sébastien Li-Thiao-Té, Literate Research versus Reproducible Research. link
- Ursula Martin, The social machine of mathematics. link
- Fernando Perez, Reproducible software vs. reproducible research. link
- Todd Rosenquist and Shane Story, Using the Intel Math Kernal Library and Intel Compilers to obtain Numerical Run-to-run Reproducible Results. link original source
- Anthony Scopatz, Passive Reproducibility: It's Not You, It's Me. link
- Benjamin Seibold, Making reproducible computational research a reasonable choice for young faculty on tenure track. link

Slides from 5-Minute Lightning Talks [edit]

Wednesday [edit]

- Noah Clemons, "How to Enforce Reproducibility with your Existing MKL Code" .pptx
- Neil Chue Hong, "The Foundations of Digital Research" .pdf
- David Ketcheson, online demo link
- Nicolas Limare, "My Christmas List for Reproducibility" .pdf
- Sebastien Li-Thiao-Te, "Lepton : Literate Executable Papers" .pdf
- Benjamin Seibold, .pdf
- Matthias Troyer, "Publishing executable papers" .pdf
- Yihue Xie, "knitr: Starting From Reproducible Homework" .pdf

Thursday [edit]

- Lorena Barba, "Reproducibility PI Manifesto" .pdf figshare
- Adam Asare, "ITN TrialShare: Promoting reproducible research and transparency in clinical trials" .pptx
- Sara Billey, ""Canonical Representations of Theorems" .pptx
- David Koop, .key
- Sarah Michalek, "Silent Data Corruption and Other Anomalies" .pdf
- Ian Mitchell, "Reproducibility(?) Review Proposal" .pdf
- Geoffrey Oxberry, "Towards Turnkey Reproducibility" .pdf
- Bob Robey, "Enhanced Precision Sums for Parallel Computing Reproducibility" .pdf
- Michael Rubenstein, "The role of computation and data in my number theoretic work" .pdf
- Fernando Seabra Chirigati, .pptx

Breakout Group Summary Slides [edit]

Wednesday [edit]

- Tools Group link
- Funding Policy Group .pdf
- Journals/Publication Policy Group .pptx
- Numerical Reproducibility Group .pptx

Thursday [edit]

- Tools Group link
- Ontology and V&V Group .pptx
- Rewards/Culture Group .pptx
- Teaching Reproducibility Group link

Final Report [edit]

To appear.

References and Links Collected [edit]

Previous Workshops and Roundtables on Reproducible Research [edit]

- Applied Mathematics Perspectives 2011: Reproducible Research: Tools and Strategies for Scientific Computing
- AAAS 2011: The Digitization of Science: Reproducibility and Interdisciplinary Knowledge Transfer
- Community Forum on Reproducible Research, ICIAM 2011.
- Yale Law School: Data and Code Sharing Roundtable, including a link to the resulting Reproducible Research Declaration (pdf) and Contributed Thought Pieces.

Why Reproducibility is an Issue [edit]

- Roger Peng's blog post about Reproducible Research: A Dissenting Opinion by Chris Drummond.

Examples where Lack of Reproducibility Causes Problems [edit]

- Duke Trials: Reports
- [Jha2012] Alok Jha, "Tenfold increase in scientific research papers retracted for fraud," U.K. Guardian, 1 Oct 2012, available at http://www.guardian.co.uk/science/2012/oct/01/tenfold-increase-science-paper-retracted-fraud.
- [Enserink2012] Martin Enserink, "Final report: Stapel affair points to bigger problems in social psychology," Science, 28 Nov 2012, available at http://news.sciencemag.org/scienceinsider/2012/11/final-report-stapel-affair-point.html.

Figure 1: Snapshot of the Workshop Wiki as of January 20, 2013

Reproducible Research

Addressing the Need for Data and Code Sharing in Computational Science

By the Yale Law School Roundtable on Data and Code Sharing

Roundtable participants identified ways of making computational research details readily available, which is a crucial step in addressing the current credibility crisis.

Progress in computational science is often hampered by researchers' inability to independently reproduce or verify published results. Attendees at a roundtable at Yale Law School (www.stodden.net/RoundtableNov212009) formulated a set of steps that scientists, funding agencies, and journals might take to improve the situation. We describe those steps here, along with a proposal for best practices using currently available options and some long-term goals for the development of new tools and standards.

Why It Matters

Massive computation is transforming science. This is clearly evident from highly visible launches of large-scale data mining and simulation projects such as those in climate change prediction,[1] galaxy formation (www.mpa-garching.mpg.de/galform/virgo/millennium/), and biomolecular modeling (www.ks.uiuc.edu/Research/namd). However, massive computation's impact on science is also more broadly and fundamentally apparent in the heavy reliance on computation in everyday science across an ever-increasing number of fields.

Computation is becoming central to the scientific enterprise, but the prevalence of relaxed attitudes about communicating computational experiments' details and the validation of results is causing a large and growing credibility gap.[2] Generating verifiable knowledge has long been scientific discovery's central goal, yet today it's impossible to verify most of the computational results that scientists present at conferences and in papers.

To adhere to the scientific method in the face of the transformations arising from changes in technology and the Internet, we must be able to reproduce computational results. Reproducibility will let each generation of scientists build on the previous generations' achievements. Controversies such as ClimateGate,[3] the microarray-based drug sensitivity clinical trials under investigation at Duke University,[4] and prominent journals' recent retractions due to unverified code and data[5,6] suggest a pressing need for greater transparency in computational science.

Traditionally, published science or mathematics papers contained both the novel contributions and the information needed to effect reproducibility—such as detailed descriptions of the empirical methods or the mathematical proofs. But with the advent of computational research, such as empirical data analysis and scientific code development, the bulk of the actual information required to reproduce results is *not* obvious from an article's text; researchers must typically engage in extensive efforts to ensure the underlying methodologies' transmission. By and large, researchers today aren't sufficiently prepared to ensure reproducibility, and after-the-fact efforts—even heroic ones—are unlikely to provide a long-term solution. We need both disciplined ways of working reproducibly and community support (and even pressure) to ensure that such disciplines are followed.

On 21 November 2009, scientists, lawyers, journal editors, and funding representatives gathered for the Yale Law School Roundtable to discuss how data and code might be integrated with tradition research publications (www.stodden.net/RoundtableNov212009). The inspiration for the roundtable came from the example set by members of the genome research community who organized to facilitate the open release of the genome sequence data. That community gathered in Bermuda in 1996 to develop a cooperative strategy both for genome decoding and for managing the resulting data. Their meeting produced the *Bermuda Principles*, which shaped data-sharing practices among researchers in that community, ensuring rapid data release (see www.ornl.gov/sci/techresources/Human_Genome/research/bermuda.shtml). These principles have been reaffirmed and extended several times, most recently in a July 2009 *Nature* article.[7] Although the computational research community's particular incentives and pressures differ from those in human genome sequencing, one of our roundtable's key goals was to produce a publishable document that discussed data and code sharing.

Yale Law School Roundtable Participants

Writing Group Members:

- Victoria Stodden, Information Society Project, Yale Law School;
- David Donoho, Department of Statistics, Stanford University;
- Sergey Fomel, Jackson School of Geosciences, The University of Texas at Austin;
- Michael P. Friedlander, Department of Computer Science, University of British Columbia;
- Mark Gerstein, Computational Biology and Bioinformatics Program, Yale University;
- Randy LeVeque, Department of Applied Mathematics, University of Washington;
- Ian Mitchell, Department of Computer Science, University of British Columbia;
- Lisa Larrimore Ouellette, Information Society Project, Yale Law School;
- Chris Wiggins, Department of Applied Physics and Applied Mathematics, Columbia University.

Additional Authors:

- Nicholas W. Bramble, Information Society Project, Yale Law School
- Patrick O. Brown, Department of Biochemistry, Stanford University
- Vincent J. Carey, Harvard Medical School
- Laura DeNardis, Information Society Project, Yale Law School
- Robert Gentleman, Director, Bioinformatics and Computational Biology, Genentech
- J. Daniel Gezelter, Department of Chemistry and Biochemistry, University of Notre Dame
- Alyssa Goodman, Harvard-Smithsonian Center for Astrophysics, Harvard University
- Matthew G. Knepley, Computation Institute, University of Chicago
- Joy E. Moore, Seed Media Group
- Frank A. Pasquale, Seton Hall Law School
- Joshua Rolnick, Stanford Medical School
- Michael Seringhaus, Information Society Project, Yale Law School
- Ramesh Subramanian, Department of Computer Science, Quinnipiac University, and Information Society Project, Yale Law School

In reproducible computational research, scientists make all details of the published computations (code and data) conveniently available to others, which is a necessary response to the emerging credibility crisis. For most computational research, it's now technically possible, although not common practice, for the experimental steps—that is, the complete software environment and the data that generated those results—to be published along with the findings, thereby rendering them verifiable. At the Yale Law School Roundtable, we sought to address this in practical terms by providing current best practices and longer-term goals for future implementation.

Computational scientists can reintroduce reproducibility into scientific research through their roles as scientists, funding decision-makers, and journal editors. Here, we discuss best practices for reproducible research in each of these roles as well as address goals for scientific infrastructure development to facilitate reproducibility in the future.

The Scientist's Role

Roundtable participants identified six steps that computational scientists can take to generate reproducible results in their own research. Even partial progress on these recommendations can increase the level of reproducibility in computational science.

Recommendation 1: When publishing computational results, including statistical analyses and simulation, provide links to the source-code (or script) version and the data used to generate the results to the extent that hosting space permits. Researchers might post this code and data on

- an institutional or university Web page;
- an openly accessible third-party archived website designed for code sharing (such as Sourceforge.net, BitBucket.org, or Github.com); or
- on a preprint server that facilitates code and data sharing (such as Harvard's Dataverse Network; see http://thedata.org).

Recommendation 2: Assign a unique ID to each version of released code, and update this ID whenever the code and data change. For example, researchers could use a version-control system for code and a unique identifier such as the Universal Numerical Fingerprint (http://thedata.org/book/unf-implementation) for data. Such an identifier facilitates version tracking and encourages citation.[8] (As another example, the PubMed Central reference number applies to all manuscripts funded by the US National Institutes of Health, creating a unique, citable digital object identifier for each; see http://publicaccess.nih.gov/citation_methods.htm.)

Recommendation 3: Include a statement describing the computing environment and software version used in the publication, with stable links to the accompanying code and data. Researchers might also include a virtual machine. A VM image with compiled code, sources, and data that can reproduce published tables and figures would let others explore the parameters

The Protein Data Bank

One example of agency-facilitated openness is the Protein Data Bank. Created in 1971, PDB's aim is to share "information about experimentally determined structures of proteins, nucleic acids, and complex assemblies" (see www.pdb.org/pdb/home/home.do). PDB has become a standard within the structural biology community during the nearly 40 years of effort to balance relationships among the journals, the author-scientists, and the database itself.

The PDB is part of a worldwide effort funded by a variety of agencies, with main hubs in the US, Japan, and Europe. With the rise of the Web, PDB usage became more intimately connected with publication, first with the understanding that data were to be available within months or a year of publication, then—owing to the coordinated decisions of the editors of *Nature*, *Science*, *Cell*, and the *Proceedings of the National Academy of Sciences*—as a simple and effective precondition for publication.[1] This has in turn enabled an entire field of statistical studies and molecular dynamics based on the structural data, a feat impossible without access to each publication's data.

More information on *Nature*'s data requirement policies is available at www.nature.com/authors/editorial_policies/availability.html; *Science* requirements are included in its general author information at www.sciencemag.org/about/authors/prep/gen_info.dtl#dataavail.

Reference

1. "The Gatekeepers," editorial, *Nature Structural Biology*, vol. 5, no. 3, 1998, pp. 165–166; www.nature.com/nsmb/wilma/v5n3.892130820.html.

around the publication point, examine the algorithms used, and build on that work in their own new research.

Recommendation 4: Use open licensing for code to facilitate reuse, as suggested by the Reproducible Research Standard.[9,10]

Recommendation 5: Use an open access contract for published papers (http://info-libraries.mit.edu/scholarly/mit-copyright-amendment-form) and make preprints available on a site such as arXiv.org, PubMed Central, or Harvard's Dataverse Network to maximize access to the work. However, the goal of enhanced reproducibility applies equally to both open access journals and commercial publications.

Recommendation 6: To encourage both wide reuse and coalescence on broad standards, publish data and code in nonproprietary formats whenever reasonably concordant with established research practices, opting for formats that are likely to be readable well into the future when possible.

The Funding Agency's Role

Funding agencies and grant reviewers have a unique role due to their central position in many research fields. There are several steps they might take to facilitate reproducibility.

Recommendation 1: Establish a joint-agency-funded archival organization for hosting—perhaps similar to the Protein Data Bank (see the "Protein Data Bank" sidebar)—and include a system for permitting incoming links to code and data with stable unique identifiers. For example, PubMed Central could be extended to permit code and data upload and archiving (possibly mirrored with existing version-control systems).

Recommendation 2: Fund a select number of research groups to fully implement reproducibility in their workflow and publications. This will allow a better understanding of what's required to enable reproducibility.

Recommendation 3: Provide leadership in encouraging the development of a set of common definitions permitting works to be marked according to their reproducibility status, including verified, verifiable, or inclusive of code or data.

Recommendation 4: Fund the creation of tools to better link code and data to publications, including the development of standardized unique identifiers and packages that allow the embedding of code and data within the publication (such as Sweave[11] or GenePattern[12]).

Recommendation 5: Fund the development of tools for data provenance and workflow sharing. It can often take researchers considerable time to prepare code and data for verification; provenance and workflow tracking tools could greatly assist in easing the transition to reproducibility. Examples include the UK-funded Taverna software package (www.mygrid.org.uk), the University of Southern California's Pegasus system (http://pegasus.isi.edu), Penn State University's Galaxy software (http://galaxy.psu.edu), and Microsoft's Trident Workbench for oceanography (http://research.microsoft.com/enus/collaboration/tools/trident.aspx).

The Journal Editor's Role

Journals are key to establishing reproducibility standards in their fields and have several options available to facilitate reproducibility.

Recommendation 1: Implement policies to encourage the provision of stable URLs for open data and code associated with published papers. (For an example, see Gary King's draft journal policy at http://gking.harvard.edu/repl.shtml.) Such URLs might be links to established repositories or to sites hosted by funding agencies or journals.

Recommendation 2: When scale permits, require the replication of computational results prior to publication, establishing a reproducibility review. To ease the burden on reviewers, publications could provide a server through which authors can upload their code and data to ensure code functionality before the results verification.

Recommendation 3: Require appropriate code and data citations through standardized citation mechanisms, such as Data Cite (http://thedata.org/citation/tech).

Several journals have implemented policies that advance sharing of the data and code underlying their computational publications. A prominent example is *Biostatistics*, which instituted an option in 2009 for authors to make their code and data available at publication time.[13] The journal itself hosts the associated data and code; code written in a standard format will also be verified for reproducibility, and the published articled is labeled accordingly. Authors can choose to release only the paper itself or to also release the code, the data, or both data and code (making the paper fully reproducible), indicated as C, D, or R, respectively, on the title pages. The policy is having an impact. Since it was implemented, three issues with a total of 43 papers have been published; of those, four papers have been marked with code availability, two with data availability, one with both, and two as fully reproducible.

In addition to traditional categories of manuscript (research, survey papers, and so on), the ACM journal *Transactions on Mathematical Software* has for many years let authors submit under a special "Algorithm" category (http://toms.acm.org). Submissions in this category include both a manuscript and software, which are evaluated together by referees. The software must conform to the *ACM Algorithms Policy*, which includes rules about completeness, portability, documentation, and structure designed "to make the fruits of software research accessible to as wide an audience as possible" (see www.cs.kent.ac.uk/projects/toms/AlgPolicy.html). If accepted, the manuscript component of an algorithm submission is published in the traditional fashion, but flagged prominently in the title as an algorithm, and the software becomes part of the AMC's collected algorithms, which are available for download and subject to the ACM Software Copyright and License Agreement. Although not appearing as frequently as traditional research papers, algorithm articles still make up a significant fraction of published articles in the journal despite the additional effort required of both authors and referees. In 2009, for example, seven out of 22 articles were in the algorithm category.

Geophysics, a prominent journal in the geosciences, created a special section on "Algorithms and Software" in 2004 (http://software.seg.org). Authors in this section must supply source code, which is reviewed by the journal to verify reproducibility of the results. The code is archived on the website. The journal *Bioinformatics* encourages the submission of code, which is actively reviewed, and an option is available for letting the journal archive the software (see www.biomedcentral.com/bmcbioinformatics/ifora/?txt_jou_id=1002&txt_mst_id=1009). *Nucleic Acids Research* publishes two dedicated issues annually: one entirely devoted to software and Web services useful to the biological community, and the other devoted to databases. The software is reviewed prior to publication and is expected to be well tested and functional prior to submission (www.oxfordjournals.org/our_journals/nar/for_authors/submission_webserver.html).

Unfortunately, archived code can become unusable—sometimes quickly—due to changes in software and platform dependencies, making published results irreproducible. One improvement here would be a system with a devoted scientific community that continues to test reproducibility after paper publication and maintains the code and the reproducibility status as necessary. When code is useful, there's an incentive to maintain it. Journals can facilitate this by letting authors post software updates and new versions.

Long-Term Goals

The roundtable participants also extended their discussion of recommendations beyond immediately available options. This section describes potential future developments, including ideal tools and practices that we might develop to facilitate reproducibility.

Goal 1: Develop version-control systems for data—particularly systems that can handle very large and rapidly changing data. Because many different research communities use computational tools, we should develop version-control systems for all aspects of research (papers, code, and data). Ideally, these would incorporate GUIs or Web-based tools to facilitate their use.

Goal 2: Publish code accompanied by software routines that permit testing of the software—test suites, including unit testing and/or regression tests, should be a standard component of reproducible publication. In addition, we should develop tools to facilitate code documentation. In the Python world, for example, the Sphinx machinery makes it possible to converge on a standard for documentation that produces consistent, high-quality documents in LaTeX, PDF, and HTML, with good math and graphics support that can be fully integrated in the development process (see http://sphinx.pocoo.org).

Goal 3: Develop tools to facilitate both routine and standardized citation of code, data, and contribution credits, including micro-contributions such as dataset labeling and code modifications, as well as to enable stable URL citations.

Goal 4: Develop tools for effective download tracking of code and data, especially from academic and established third-party websites, and use these data in researcher evaluation.

Goal 5: Mark reproducible published documents as such in an easily recognizable and accepted way.[9,12,13]

Goal 6: Require authors to describe their data using standardized terminology and ontologies. This will greatly streamline the running of various codes on data sets and a uniform interpretation of results.

Goal 7: That institutions, such as universities, take on research compendia archiving responsibilities as a regular part of their role in supporting science. This is already happening in several places, including Cornell University's DataStar project.[14,15]

Goal 8: Clarify ownership issues and rights over code and data, including university, author, and journal ownership. Develop a clear process to streamline agreements between parties with ownership to facilitate public code and data release.

Goal 9: Develop deeper communities that maintain code and data, ensure ongoing reproducibility, and perhaps offer tech support to users. Without maintenance, changes beyond individual's control (computer hardware, operating systems, libraries, programming languages, and so on) will break reproducibility. Reproducibility should become the responsibility of a scientific community, rather than rest on individual authors alone.

Novel contributions to scientific knowledge don't emerge solely from running published code on published data and checking the results, but the ability to do so can be an important component in scientific progress, easing the reconciliation of inconsistent results and providing a firmer foundation for future work.

Reproducible research is best facilitated through interlocking efforts in scientific practice, publication mechanisms, and university and funding agency policies occurring across the spectrum of computational scientific research. To ultimately succeed, however, reproducibility must be embraced at the cultural level within the computational science community.[16] Envisioning and developing tools and policies that encourage and facilitate code and data release among individuals is a crucial step in that direction.

References

1. R. Stevens, T. Zacharia, and H. Simon, *Modeling and Simulation at the Exascale for Energy and the Environment*, report, US Dept. Energy Office of Advance Scientific Computing Research, 2008; www.sc.doe.gov/ascr/ProgramDocuments/Docs/TownHall.pdf.
2. D. Donoho et al., "Reproducible Research in Computational Harmonic Analysis," *Computing in Science & Eng.*, vol. 11, no. 1, 2009, pp. 8–18.
3. "The Clouds of Unknowing," *The Economist*, 18 Mar. 2010; www.economist.com/displaystory.cfm?story_id=15719298.
4. K. Baggerly and K. Coombes, "Deriving Chemosensitivity from Cell Lines: Forensic Bioinformatics and Reproducible Research in High-Throughput Biology," *Annals Applied Statistics*, vol. 3, no. 4, 2009, pp. 1309–1334.
5. B. Alberts, "Editorial Expression of Concern," *Science*, vol. 327, no. 5962, 2010, p. 144; www.sciencemag.org/cgi/content/full/327/5962/144-a.
6. G. Chang et al., "Retraction," *Science*, vol. 314, no. 5807, 2006, p. 1875; www.sciencemag.org/cgi/content/full/314/5807/1875b.
7. Toronto International Data Release Workshop, "Prepublication Data Sharing," *Nature*, vol. 461, pp. 168–170; www.nature.com/nature/journal/v461/n7261/full/461168a.html.
8. M. Altman and G. King, "A Proposed Standard for the Scholarly Citation of Quantitative Data," *D-Lib Magazine*, vol. 13, nos. 3-4, 2007; www.dlib.org/dlib/march07/altman/03altman.html.
9. V. Stodden, "Enabling Reproducible Research: Licensing for Scientific Innovation," *Int'l J. Comm. Law & Policy*, vol. 13, Jan. 2009; www.ijclp.net/issue_13.html.
10. V. Stodden, "The Legal Framework for Reproducible Scientific Research: Licensing and Copyright," *Computing in Science & Eng.*, vol. 11, no. 1, 2009, pp. 35–40.
11. F. Leisch, "Sweave and Beyond: Computations on Text Documents," *Proc. 3rd Int'l Workshop on Distributed Statistical Computing*, 2003; www.ci.tuwien.ac.at/Conferences/DSC-2003/Proceedings/Leisch.pdf.
12. J. Mesirov, "Accessible Reproducible Research," *Science*, vol. 327, no. 5964, 2010, pp. 415–416.
13. R. Peng, "Reproducible Research and *Biostatistics*," *Biostatistics*, vol. 10, no. 3, 2009, pp. 405–408.
14. G. Steinhart, D. Dietrich, and A. Green, "Establishing Trust in a Chain of Preservation: The TRAC Checklist Applied to a Data Staging Repository (DataStaR)," *D-Lib Magazine*, vol. 15, nos. 9-10, 2009.
15. G. Steinhart, "DataStar: An Institutional Approach to Research Data Curation," *IAssist Quarterly*, vol. 31, no. 3-4, 2007, pp. 34–39.
16. V. Stodden, *The Scientific Method in Practice: Reproducibility in the Computational Sciences*, paper no. 4773-10, MIT Sloan Research, 9 Feb. 2010; http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1550193.

IEEE computer society

PURPOSE: The IEEE Computer Society is the world's largest association of computing professionals and is the leading provider of technical information in the field.
MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.
COMPUTER SOCIETY WEB SITE: www.computer.org
OMBUDSMAN: To check membership status or report a change of address, call the IEEE Member Services toll-free number, +1 800 678 4333 (US) or +1 732 981 0060 (international). Direct all other Computer Society-related questions—magazine delivery or unresolved complaints—to help@computer.org.
CHAPTERS: Regular and student chapters worldwide provide the opportunity to interact with colleagues, hear technical experts, and serve the local professional community.
AVAILABLE INFORMATION: To obtain more information on any of the following, contact Customer Service at +1 714 821 8380 or +1 800 272 6657:

- Membership applications
- Publications catalog
- Draft standards and order forms
- Technical committee list
- Technical committee application
- Chapter start-up procedures
- Student scholarship information
- Volunteer leaders/staff directory
- IEEE senior member grade application (requires 10 years practice and significant performance in five of those 10)

PUBLICATIONS AND ACTIVITIES

Computer: The flagship publication of the IEEE Computer Society, *Computer*, publishes peer-reviewed technical content that covers all aspects of computer science, computer engineering, technology, and applications.
Periodicals: The society publishes 13 magazines, 18 transactions, and one letters. Refer to membership application or request information as noted above.
Conference Proceedings & Books: Conference Publishing Services publishes more than 175 titles every year. CS Press publishes books in partnership with John Wiley & Sons.
Standards Working Groups: More than 150 groups produce IEEE standards used throughout the world.
Technical Committees: TCs provide professional interaction in more than 45 technical areas and directly influence computer engineering conferences and publications.
Conferences/Education: The society holds about 200 conferences each year and sponsors many educational activities, including computing science accreditation.
Certifications: The society offers two software developer credentials.
For more information, visit www.computer.org/certification.

IEEE

revised 17 Jun. 2010

COMPUTER SOCIETY OFFICES

Washington, D.C.: 2001 L St., Ste. 700, Washington, D.C. 20036
Phone: +1 202 371 0101 • **Fax:** +1 202 728 9614
Email: hq.ofc@computer.org
Los Alamitos: 10662 Los Vaqueros Circle, Los Alamitos, CA 90720-1314
Phone: +1 714 821 8380
Email: help@computer.org
Membership & Publication Orders:
Phone: +1 800 272 6657 • **Fax:** +1 714 821 4641
Email: help@computer.org
Asia/Pacific: Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan
Phone: +81 3 3408 3118 • **Fax:** +81 3 3408 3553
Email: tokyo.ofc@computer.org

Next Board Meeting:
15–16 Nov. 2010, New Brunswick, NJ, USA