Reproducible Computing

Patrick Wessa
KU Leuven, Belgium

Ian E. Holliday
Aston University, UK

INTRODUCTION

Is it easy for reviewers of scientific journals to reproduce and verify the statistical analysis that is presented in submitted articles? Do published science articles provide adequate information for readers to reproduce the research results quickly and without much effort? Are students able to reproduce or re-use the computations in course texts effectively and without making errors? Is it easy to assess whether the empirical research in student papers is original and correct?

If your answer to these questions is no, you might consider Reproducible Computing (RC) as a generic, technological solution which is freely available and easy to use. By definition, RC allows any (non-expert) writer to create and publish electronic documents which allow any (non-expert) reader to reproduce and re-use the computations that are presented, without the need to download or install anything on the client machine. The reader only needs to click on a table or picture to open a web application which provides instant and free access to the underlying data and software.

Even though RC is still in its infancy, remarkable progress has been made in recent years. This article discusses how this is achieved and what the academic community can do to successfully implement RC for the purpose of academic teaching, scientific research, and publishing. The focus is on the concepts, consequences, and challenges that are related to RC from the Information Science point of view. First we define and describe the most important concepts based on literature. The second section discusses the roles and effects of important concepts based on Levels of Reproducibility, real life implementations, and our recommendations. The third section briefly discusses some of the challenging ideas for further research.

BACKGROUND

The problem of irreproducible research received a great deal of attention within the statistical computing and bioinformatics communities. Arguably, the most famous quote about this problem is called Claerbout's principle (de Leeuw, 2001): "An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and that complete set of instructions that generated the figures.".

Other scientists have extended Claerbout's Principle and specified additional requirements for Reproducible Research (de Leeuw, 2001): "First, there is no reason to single out figures. The same 'Principle' obviously applies to tables, standard errors, and so on. The fact that figures often happen to be easier to reproduce, does not preclude that we should apply the same rule to any form of computer-generated output. Second, there is no reason to limit the Claerbout's Principle to published articles. We can make exactly the same statement about our lectures and teaching, certainly in the context of graduate teaching. We must be able to give our students our code and our graphics files, so that they can display and study them on their own computers (and not only on our workstations, or in crowded university labs). And third, and perhaps most importantly, it is not clearly defined what a 'software environment' is. Buckheit and Donoho apply the principle in such a way that everybody who wants to check their results is forced to buy MatLab®. Not Mathematica®, Macsyma®, or S-plus®. Those you may need to buy for other articles. This violates the Freeware Principle... ".

Several solutions have been proposed but the most prominent one is based on the concept of Literate Programming (Knuth, 1984) and has been implemented in

DOI: 10.4018/978-1-4666-5888-2.ch647

an R package called "Sweave" (Leisch, 2003) where the concept of the so-called Compendium plays a fundamental role. The (traditional) Compendium is defined as an integrated collection of text (written in LaTeX), statistical code (written in R), and data that allows the presented science to be reproduced. All the necessary documents that are needed to create the article are contained in an archive file (preferably in tar.gz or zip format).

While the traditional Compendium, based on Sweave-like technology, seems to solve the problem of irreproducible research there are several shortcomings and remaining problems:

- Readers are required to download, install, and execute software (i.e. R interpreter, R scripts, and all necessary libraries) on their client machines which requires technical knowledge and may lead to a variety of compatibility and security issues.
- The reader is assumed to have a working knowledge of LaTeX and R code.
- If the reader re-uses the archived computation in derived work then there is no easy way to ensure dissemination (a new Compendium must be created first and sent to all users who might be potentially interested).
- The use of Compendiums cannot be measured or monitored which implies that we cannot research the related educational processes.

Even if we ignore the impracticalities of the proposed solutions in the literature, one may conclude that ordinary data sharing is insufficient to solve the irreproducibility problem – there seems to be a consensus that a solution must encompass the entire software environment and the text that is disseminated. As far as the software environment is concerned, it should be noted that this includes several components: statistical R code, R interpreter, R libraries, operating system, and the hardware. All of these components are necessary to reproduce statistical analyses – therefore, all should be included in any Reproducibility solution that is proposed. Since it is reasonable to argue that we cannot distribute the hardware in a Compendium, we have to treat this differently from the distributable code that represents the analytical logic. This separation, however, is problematic because the statistical software that interprets the logic is dependent on the hardware. In other words, we have to separate the Analytical Code layer (i.e. program logic written as R Code, the source code of the R language and R libraries) from the Computing System layer (i.e. the hardware, the operating system, and the executable statistical software).

We have opted to introduce an additional separation for conceptual and practical reasons: the methodology and accompanying Model/Parameter Specification layer is treated differently from the Analytical Code layer. The use of Models and Parameters can be described formally or in words (e.g. a regression equation) within the text that is disseminated – often this is not accompanied by the actual code or script that is used to estimate the parameters in the statistical software. Sometimes, researchers only include a reference of the model (e.g. when the model has been well-established in the literature). On the other hand, the Analytical Code provides the reader with explicit information of the commands that are needed to compute the analysis – these commands may refer to high-level instructions to compute some standard routines that have been made available (e.g. in R libraries) or they may include detailed logic to run a custom made algorithm.

The Analytical Code can be subject to different types of software licenses. When the software is proprietary (as is the case in many popular statistical software products) then the user has no information about what is actually computed – the user has to believe the claims that are made by the creator/owner of the software. In open source systems there is, at least, the opportunity to read the source code that is used in the statistical computations – this however, does not automatically include the right to make changes and redistribute the software. Only Free Software comes with the guarantee of the essential freedom to reproduce and re-use (Free Software Foundation, 2013):

- "The freedom to run the program, for any purpose (freedom 0).
- The freedom to study how the program works, and change it so it does your computing as you wish (freedom 1). Access to the source code is a precondition for this.
- The freedom to redistribute copies so you can help your neighbor (freedom 2).