

Seven Years of Reproducible Research: From R / Sweave to Org

Christophe Pouzat

Mathématiques Appliquées à Paris 5 (MAP5)
Université Paris-Descartes and CNRS UMR 8145
`christophe.pouzat@parisdescartes.fr`

SimGrid User Days

Lyon, June 14 2012

Outline

Introduction for the reader

Introduction

The context

Tools: R and Sweave

Tools: emacs, Org and Babel

Links and References

A note to the reader

I've tried to put hyperlinks to websites and papers for most of the technical terms appearing (in red) in this document. I have also added references and links at the end of the document following the "Acknowledgments".

What is “Reproducible Research”?

Reproducible Research (*RR*) or **reproducible data analysis** is an approach aiming at complementing classical printed scientific articles with **everything** required to independently reproduce the results they present.

What is “Reproducible Research”?

Reproducible Research (RR) or **reproducible data analysis** is an approach aiming at complementing classical printed scientific articles with **everything** required to independently reproduce the results they present.

“Everything” covers here:

- ▶ the data,
- ▶ the computer codes,
- ▶ a precise description of how the code was applied to the data.

What is “Reproducible Research”?

Reproducible Research (RR) or **reproducible data analysis** is an approach aiming at complementing classical printed scientific articles with **everything** required to independently reproduce the results they present.

“Everything” covers here:

- ▶ the data,
- ▶ the computer codes,
- ▶ a precise description of how the code was applied to the data.

The “movement” started with what economists have been calling **replication** since the early eighties to end with what is now called **reproducible research** in computational data analysis oriented fields like statistics and signal processing.

What Am I Going to Talk About?

- ▶ I have been implementing RR approaches for seven years now in a context that I will briefly describe.
- ▶ The idea of RR seems nice (to me) but proper tools turned out to be extremely important to go from ideas to practice.
- ▶ I started using the **R** software together with its **Sweave** function. I'm still using that tool and I will describe some of its features, but...
- ▶ I'm now using mostly the **emacs** / **Org** / **Babel** combination that I will describe at the end of the talk.

The context

I'm mainly working with neurophysiological data:

- ▶ Extracellular recordings performed with multi-electrodes arrays (MEA).
- ▶ Calcium concentration measurements with fluorescent indicators.

I'm also now frequently helping colleagues in Biology analyzing their data sets.

Data analysis

I can broadly classify the types of data analysis I perform in three categories:

1. Do a “standard analysis” on a new data set.
2. Develop short codes—10 to 100 lines in a “high level” languages like R—for a new type of analysis.
3. Implement a new method through the development of large codes—more than 1000 lines—in a compiled language (C and, more and more, Common Lisp).

Reproducibility

In order to make an analysis reproducible and **open to criticism** we have to:

- ▶ describe our hardware environment: CPU (`cat /proc/cpuinfo`), memory (`cat /proc/meminfo`), etc.
- ▶ describe our software environment: kernel, compiler versions, etc; **working with open source software facilitates clearly this part.**
- ▶ **document our codes.**
- ▶ **document every non automatic decision made during the analysis.**

One Remark

Results reproducibility in an experimental science implies that:

1. **the data are reproducible,**
2. as well as the analysis.

Strictly speaking, only the second point is going to be discussed here; but a reproducible analysis **implies raw data access**. Accessible raw data become open to criticism and comparable. **That should help progress on the first point.**

From Theory to Practice: the tools (1)

Once convinced by the *RR* paradigm as are usually students and young researchers—established ones in my field at least tend to be less convinced—we must find the tools to go from theory to practice:

- ▶ Tools are critical for a systematic implementation of the *RR*.
- ▶ The first approaches like the **Stanford Exploration Project** (Claerbout et Karrenbach, 1992) one, based on **L^AT_EX**, **C** and **Fortran** codes and a compilation controlled by **make**, are too heavy for *my* daily work made of a lot of exploratory analysis.

From Theory to Practice: the tools (2)

- ▶ I have to work mainly with “high level” software / programming languages (Matlab, GNU Octave, R, python).
- ▶ I generate a lot of figures.
- ▶ Having a minimal number of files to maintain helps a lot in an RR context.
- ▶ Having files that can be opened by my biologist collaborators in a **language intelligible to them** is a big plus.

I will now describe briefly two tools satisfying part or all of these criteria.

R

R is a language distributed under a *GPL* license described as follows on the [FAQ](#) pages:

R is a system for statistical computation and graphics. It consists of a language plus a run-time environment with graphics, a debugger, access to certain system functions, and the ability to run programs stored in script files.

For the programmers: R was inspired by [Scheme](#)—from which it kept [lexical scoping](#)—but has an [ALGOL](#) like syntax: we write `2+2` and not `(+ 2 2)`.

Sweave

- ▶ **Sweave** is an R function.
- ▶ Sweave processes **text** files where “prose” written in \LaTeX or HTML is mixed with code written in R.
- ▶ Sweave makes a verbatim copy of the prose part, **executes the code part** and puts the result (table, figure) in a new \LaTeX or HTML file.
- ▶ Sweave derives from the **literate programming** approach advocated by Don Knuth.
- ▶ Sweave syntax is very much alike the **noweb** one.

With Sweave, one stores the code generating the tables and figures of a paper instead of the tables and figures themselves.

Example

A Sweave file part looks like:

```
\subsection{Data Summary}
The \textit{5 numbers summary} of  $\alpha$  data set
is :
<<summary-set-alpha>>=
summary( $\alpha$ )
@
We see \textbf{no sign of amplifier saturation
during the acquisition}...
```

Extensions

- ▶ A fine control of “code chunks” output is possible (*i.e.*, we keep the result, the result and the code, etc).
- ▶ Code chunks evaluation results can be “cached” (written to disk) avoiding re-evaluation when the document is regenerated.
- ▶ Raw data can be distributed with the Sweave file together with custom developed C or R libraries as an R package; this is the *compendium* approach advocated by **Gentleman and Temple-Lang (2007)**.

Shortcomings

- ▶ Obviously one must know R and \LaTeX (or HTML).
- ▶ For me \LaTeX is perfect to write scientific papers and tutorials but it is somewhat heavy for my “lab book” (*i.e.*, for notes taking).
- ▶ Conversion from one output format (PDF) to another (HTML) must be done with external tools like **TeX4ht** (or the source file has to be rewritten).
- ▶ Classical literate programming (the Knuth’s way) cannot be implemented.

Org

- ▶ **Org** is a **mode** of **emacs**, the editors' "Swiss army knife".
- ▶ Org generates PDF—with \LaTeX or $X_{\text{E}}\text{TeX}$ or LuaTeX —, HTML, DocBook, ODT— *OpenDocument Text*, OpenOffice / LibreOffice format— outputs from a **single** file written with a simplified syntax like **Markdown**.
- ▶ `"* My section"` signals the beginning of a section entitled "My section".
- ▶ `"** My sub-section"` signals the beginning of a sub-section entitled "My sub-section".
- ▶ `*bold*` appears **bold**, `/italic/` appears *italic*, `_underline_` appears underline.

I'm just scratching the surface of Org here...

Babel

Babel is an Org extension whose name comes from:

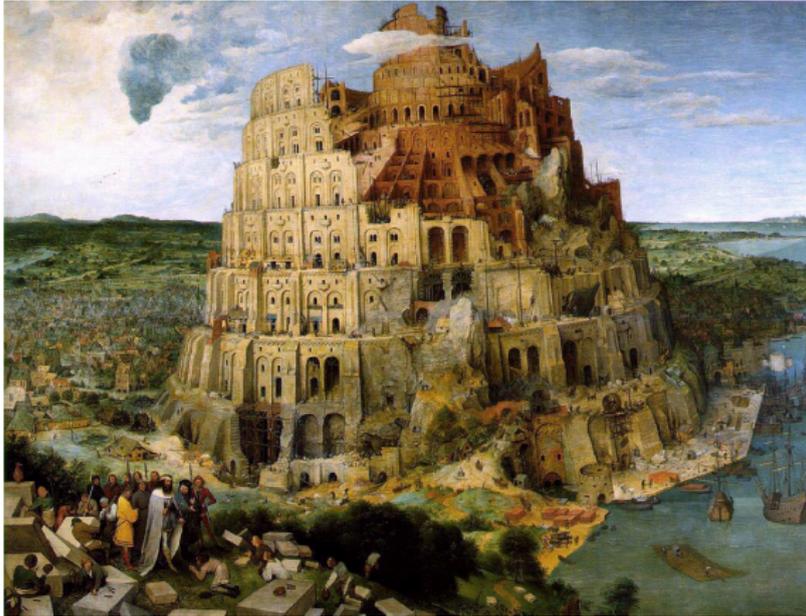


Figure : The programming and scripting languages are referred to here.

Babel

- ▶ With Babel we can include “code blocks” in Org files.
- ▶ 39 “programming” languages including R, Python, Perl, C, C++, Matlab, Maxima, SQL, Java, Gnuplot, are currently supported. They can be used **in the same Org file**.
- ▶ Babel makes **meta-programming** possible, that is, the output of a perl code block can be used as the input of a Gnuplot one.
- ▶ With Babel both RR and literate programming can be implemented.
- ▶ Everything can be kept in a **single file**.

Example

The previous example gives:

```
** Data summary
```

```
The /5 numbers summary/ of data set  
 $\alpha$  is:
```

```
#+BEGIN_SRC R :exports results
```

```
summary( $\alpha$ )
```

```
#+END_SRC
```

```
We see *no sign of amplifier saturation  
during the acquisition*...
```

Some remarks in guise of conclusion

- ▶ Org is **very simple to learn**.
- ▶ People coding in Matlab, Python R, etc. with emacs can immediately switch to Org + Babel.
- ▶ A single source file gives multiple outputs.
- ▶ With its simplified syntax and its use of UTF-8, even people who do not know emacs and who **do not use it** can work on Org source files without “screwing up”. For me this is an extremely useful feature in my collaborations with biologists.
- ▶ Org is presently evolving from a “simple” emacs mode towards a **format** and soon emacs won't be necessary anymore to process Org files.

Acknowledgments

I want to thank:

- ▶ the organizers of these “SimGrid User Days”,
- ▶ the developers of R, \LaTeX , emacs,... (the list is too long) and, especially, Org and Babel,
- ▶ my students and postdocs (M. Delecluse, T. Lieury, S. Joucla, R. Franconville) who were “strongly encouraged” to adopt the RR paradigm,
- ▶ Bastien Guerry for suggestion on a previous version of this talk (in French),
- ▶ the CNRS for allowing me to do all that,
- ▶ you for listening.

Notes (1)

Readers can find on the following slides additional material related to the presentation:

- ▶ In contrast to Matlab, R is a “proper” language whose definition can be found at:
<http://cran.r-project.org/doc/manuals/R-lang.html>.
- ▶ When Ross Ihaka and Robert Gentleman developed R **deliberate choices were made**. These choices are discussed in a paper available at:
<http://www.stat.auckland.ac.nz/%7Eihaka/downloads/R-paper.pdf>.
- ▶ Other choices could have been made and now seem, with hindsight, better as discussed in:
<http://www.stat.auckland.ac.nz/%7Eihaka/downloads/Compstat-2008.pdf>.

Notes (2)

- ▶ One of the seminal papers on emacs by Richard Stallman can be downloaded at:
<ftp://publications.ai.mit.edu/ai-publications/pdf/AIM-519A.pdf>.
- ▶ After the tutorial and the guided tour (<http://www.gnu.org/software/emacs/tour/>), Phil Sung's lectures (<http://web.psung.name/emacs/>) are a must.
- ▶ To learn using R with emacs, take a look at Stephen Eglen's tutorials:
<http://www.damp.cam.ac.uk/user/sje30/ess11/index.html>.

Notes (3)

- ▶ The Org site: <http://orgmode.org/>.
- ▶ To learn the numerous features (and there are a lot of them!) of Org, check Worg (<http://orgmode.org/worg/>) and its tutorials.
- ▶ The Babe1 site: <http://orgmode.org/worg/org-contrib/babel/>.
- ▶ To start digging into Babe1 look at the following paper: Eric Schulte, Dan Davison, Thomas Dye et Carsten Dominik (2012) A Multi-Language Computing Environment for Literate Programming and Reproducible Research *The Journal of Statistical Software* **46** (3) : <http://www.jstatsoft.org/v46/i03>.

Notes (4)

I conclude shamelessly with some self-advertisement:

- ▶ A paper on RR with M. Delescluse, S. Joucla, R. Franconville and T. Lieury: <http://hal.archives-ouvertes.fr/hal-00591455>.
- ▶ A web page: http://www.biomedicale.univ-paris5.fr/phycserv/C_Pouzat/ReproducibleDataAnalysis/ReproducibleD

This document was prepared with emacs, Org and X₃T₃E_X.