Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 4 (2011) 618–626

Procedia Computer Science

International Conference on Computational Science, ICCS 2011

Executable Papers for the R Community: The R^2 Platform for Reproducible Research

Friedrich Leisch

Institute of Applied Statistics and Computing, University of Natural resources and Life Sciences
Gregor-Mendel-Straße 33, 1180 Vienna, Austria

Manuel Eugster, Torsten Hothorn

Department of Statistics, Ludwig-Maximilians-Universität München
Ludwigstraße 33, 80539 Munich, Germany

Abstract

Reviewing the computational part of scientific papers puts a lot of effort on referees: even if authors provide their data and code the referee often needs to install additional software on his machine and figure out which parts of the code belong to which part of the manuscript. As a result, computational results or often not reviewed at all. We propose a new web service which outsources validation of computational results in executable papers to an independent third party. Our system adapts the well-tested toolbox currently checking R extension packages in software repositories like CRAN to check manuscripts in paper repositories. In addition, paper packages can easily be downloaded from the server and installed to replicate results locally by anyone wishing to do so.

Keywords: reproducible research, statistical computing, R, Sweave

1. Introduction

One of the cornerstones of modern science is that results need verification by peers in order to be accepted. Physicists must describe their experiments with enough details such that others can replicate them, mathematicians publish proofs of theorems, and medical studies are routinely replicated by other research teams. Papers submitted to scientific journals are subject to a peer review process where usually two to three experts check the manuscripts.

For a growing number of scientific fields complicated numerical computations and simulation-based experiments play an essential part of research. While publication of research papers is based on the verification or proper referencing of proofs for every theorem, there is a tendency to accept seemingly realistic computational results, as presented by figures and tables, without any proof of correctness. Yet, these results are critical for justifying the proposed methods and represent a substantial percentage of the content in many journal articles.

Email addresses: Friedrich.Leisch@boku.ac.at (Friedrich Leisch), Firstname.Lastname@stat.uni-muenchen.de (Manuel Eugster, Torsten Hothorn)

URL: http://statedv.boku.ac.at (Friedrich Leisch), http://www.stat.uni-muenchen.de (Manuel Eugster, Torsten Hothorn)

doi:10.1016/j.procs.2011.04.065

```
Most groups working on the task figured out the
corresponding R code after some time:
<<keep.source=TRUE,echo=TRUE>>=
BULIMIA <- read.csv("bulimia.csv")
BULIMIA$GROUP <- factor(BULIMIA$GROUP,
                        levels=c("self-help", "group"))
BE <- paste("BE4WV", 2:4, sep="")
BEDATA <- reshape(BULIMIA[,c(BE,"GROUP")], varying=list(BE),
                  direction="long", times=2:4,  v.names="y")
summary(BEDATA)
@
The other groups were given this information in
the middle of the session such that they could also
try their luck with the mixed effects models.

<<fig=TRUE,echo=FALSE,width=5,height=3>>=
library("lattice")
print(bwplot(y~ordered(time)|GROUP, data=BEDATA))
@
```

Figure 1: Parts of the Sweave files generating [3].

Computations can be proofed. Correctness can be verified by delivering appropriately documented and functional code, along with corresponding inputs and data, *for all results* along with the paper. This is not new, [1] define what [2] calls Claerbout's Principle:

> *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 the complete set of instructions which generated the figures.*

Almost two decades later computational results are still routinely published without delivering any code at all [3]. In this project we propose software tools and web services to support authors providing code for papers and reviewers in checking results.

2. R, Packages and Sweave

We focus on papers where the R environment for statistical computing and graphics [4] has been used to analyze data or propose new methods. But many of the concepts are generic and can easily be adapted to other software environments. One of the key factors for the immense success of R over the last decade has been the package system and the Comprehensive R Archive Network CRAN [5, 6], which allow hundreds of volunteers to contribute to the project. R has a very advanced system to check extension packages:

- Documentation and code are checked for consistency.
- All examples are executed.
- Test code is executed and output automatically compared to pre-computed results.
- ...

This `R CMD check` procedure is available for developers to improve their code. But it is also used to guarantee that all of the 2500+ packages on CRAN actually work for all current versions of R. All packages are checked daily for several versions of R (release, patch and devel branch) and computer platforms (Linux, MacOS, Solaris, Windows) and check results are published on the web. If changes in the R base distribution break code in contributed packages,

this will be detected at latest 48 hours after making the change. A package author developing on a Linux machine automatically gets his package checked on all other platforms (things working on Linux may not work for Windows).

R also features its own "executable paper" format named Sweave [7]: R Code can be directly embedded into LaTeX documents. When run on the source file, Sweave replaces all code by the corresponding results, which can be text, tables, figures, etc.. If data or analysis change all figures and results can easily be recreated on the fly. Figure 1 shows a small part from the Sweave files generating [3]. Sweave uses syntax and concepts similar to literate programming tools [8, 9]. The `<<...>>=` starts a code chunk, options in the middle control how R input and output are rendered in the final document. The code in the example reads a data set into R, reshapes it and the summarizes the result. Figure 2 shows the result of running Sweave: Because option `echo=TRUE` is used both R input and output are shown and marked in different colors for simple identification. With `echo=FALSE` only the red output would have been shown. This works not only for numerical output of R but also for figures, see the second code chunk in the example.

R package manuals written in Sweave format are called package vignettes and their code is also subject to the rigorous checking process described above. Sweave is used in many different application areas and has been recommended for reproducible research in biostatistics and bioinformatics [3, 10], data mining [11], econometrics [12], or signal processing [13], to name a few.

Another example of Sweave usage is the detailed 'forensic' analysis of a published microarray study [14]. It contains detailed case studies of problems when reproducing published results by others, which even led to the suspension of the authors of the original manuscript (`http://cancerletter.com/articles/20100902`). They make their own paper reproducible by providing extended supplementary electronic material created with Sweave. Note that such such a 'forensic' analysis may be the starting point of a scientific discussion: [15] complain about lack of reproducibility of the results in [16] backed up again by supplementary electronic material. [17] do forensic on the forensic and show that the truth is somewhere in the middle, arguments supported by package `dressCheck` available from `http://www.bioconductor.org`.

There are also Sweave versions using OpenOffice or HTML for text rendering, and StatWeave which extends the approach to other statistical software systems like SAS. For writing Sweave documents there are utilities like "Emacs speaks statistics" [18] which connects the source file to a running R process for simple execution of code chunks.

3. The R^2 Platform

R packages can contain code in various programming languages, documentation, data sets and optionally executable papers in Sweave format. The `R CMD check` utility makes them an ideal tool for reproducing results in executable papers: an R^2 paper package may contain the complete paper or only the computable parts like figures, tables, electronic supplementary material, etc.. The DESCRIPTION file describes dependencies using the regular R package management system (which in turn was modeled after Linux package managers). The data directory contains all needed data either by a copy of the data set itself or R code to download data from the Internet.

The new service we propose on top of this already existing and well-tested tools is a web server that automatically checks such executable papers for authors and reviewers. Figure 3 shows the process model (using the Business Process Modeling Notation [19]) of the R^2 platform for the submission of a paper package. All related actors, tasks, objects, and connections are presented. Note that the message flow between author and reviewer is shortened; the journal actor is not modeled.

Each paper is submitted as a paper package and the submission to the R^2 server farm is, e.g, per FTP or an HTML form, for details see Section 3.1 below. We then use a CRAN-like package building and checking system:

- When the paper is built, the resulting R output, figures and the complete R workspace are saved.
- Compiling the paper on various platforms (Linux, Windows, R versions, ...) allows to show possible numerical differences by comparing the individual workspaces.
- The final paper (PDF) contains all computed parts highlighted by font and/or background color to differentiate them from static content, see Figure 2.
- The author approves the final paper and receives a validation link.

Most groups working on the task figured out the corresponding R code after some time:

```
> BULIMIA <- read.csv("bulimia.csv")
> BULIMIA$GROUP <- factor(BULIMIA$GROUP,
+                         levels=c("self-help", "group"))
> BE <- paste("BE4WV", 2:4, sep="")
> BEDATA <- reshape(BULIMIA[,c(BE,"GROUP")], varying=list(BE),
+                   direction="long", times=2:4,  v.names="y")
> summary(BEDATA)
```

```
      GROUP          time         y                id
 self-help:120   Min.   :2   Min.   :  0.00   Min.   : 1
 group    :123   1st Qu.:2   1st Qu.:  3.00   1st Qu.:21
                 Median :3   Median : 12.00   Median :41
                 Mean   :3   Mean   : 18.79   Mean   :41
                 3rd Qu.:4   3rd Qu.: 24.25   3rd Qu.:61
                 Max.   :4   Max.   :110.00   Max.   :81
                             NA's   : 59.00
```

The other groups were given this information in the middle of the session such that they could also try their luck with the mixed effects models.

Figure 2: Output of Sweave on Figure 1 and shown in [3].

This validation link points to a website where both the check results and the final paper are available. The author can send the validation link along with the journal paper submission. Any referee can go to this website and can approve that each data-based part in the paper he has to review conforms with the listed parts on the validation website. The R^2 server will be run by a trusted and independent entity like the R Foundation for Statistical Computing. In case the referee wants to check results on his own machine, he can download the paper package like any other R package and do so.

The obvious advantage of the R^2 platform is that reviewers do not need to check code on their own machines, this is outsourced to a (hopefully) trusted entity which is independent from paper author, journal publisher and referee. In addition, it makes a lot of sense to create different paper checkers for different software systems rather than, say, journals or scientific communities. The computational challenges to check a paper which uses R to analyze, e.g., marketing data or genomic data are very similar, although the application areas are completely different.

Many components of our framework are well-known and widely used:

R: the lingua franca of statistics and data analysis

Sweave: the most popular format for executable papers in the R community

CRAN: package building and checking system has been developed for more then a decade and copes successfully with the exponential growth of the number of packages

In the following we describe three steps of the complete process in more detail: How to generate a data package in the first place, what the checking process on the R^2 server does and how reviewers and other peers can reproduce the results on their own machines.

3.1. Paper Package

There are two ways to create an R^2 paper package for submission to the R^2 server:

1. Prepare the complete package manually on your own machine and upload as a `.tar.gz` file.
2. Upload either an Sweave file or R code plus data sets (or URLs for data sets) to a web form which will generate the package directly on the server.

In addition publishers can interface both submission methods directly from within their editorial management online portals: If the electronic supplementary material to an article contains R code and data sets, a "Submit to R^2" button could automatically transmit the data to our server such that the author has to register in only one place. In this case reviewers can be directly provided with URLs for check results.

An R^2 paper package is very similar to a regular R package, see [20] for details. It is a `tar` archive of a directory containing a `DESCRIPTION` file with meta-data (author, version, title, etc.), and the subdirectories `R` (R functions), `data` (data sets or code to download data over the Internet), `inst` (Sweave files), and `tests` (R code reproducing the results in the paper). Some of these subdirectories can be missing in case they are not needed.

3.2. Check Paper Package and Generate Results

The R^2 server runs a modified version of the usual `R CMD check` procedure [20]. This includes only the tests that are necessary for paper packages, and some new ones which are especially necessary for them. `R CMD check` itself is very modular, checks can easily be newly combined for new situations. Our checks include

1. The package is installed (will warn about missing pieces). The file names are checked to be valid across file systems and supported operating system platforms. The `DESCRIPTION` file is checked for completeness, and some of its entries for correctness.
2. The R files are checked for syntax errors.
3. All R code in `tests` is executed and compared with target output files if available. This will be repeated on all major computing platforms and results compared.
4. The code in Sweave files is executed, and the PDFs recreated from the sources. Recomputed numerical results and figures are highlighted in color.

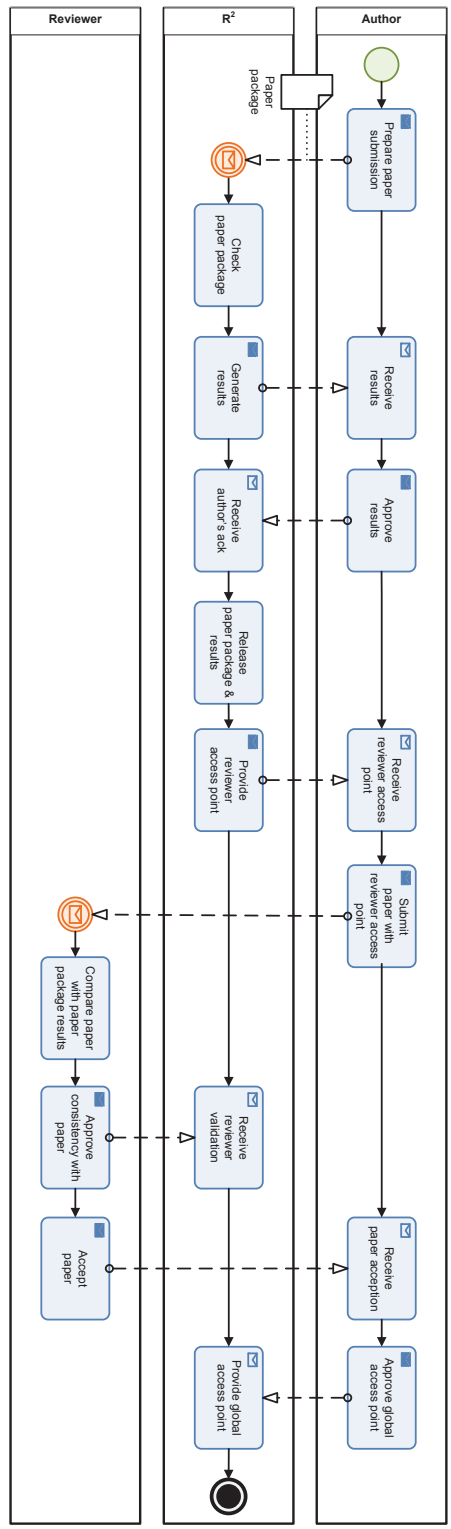

Figure 3: Process model for the submission of a paper package.

```
* using R version 2.12.2 (2011-02-25)
* using platform: x86_64-pc-linux-gnu (64-bit)
* using session charset: UTF-8
* checking for file bioinfrepro/DESCRIPTION ... OK
* checking extension type ... R2 Paper Package
* checking package dependencies ... OK
* checking for portable file names ... OK
* checking for sufficient/correct file permissions ... OK
* checking DESCRIPTION meta-information ... OK
* checking top-level files ... OK
* checking package subdirectories ... OK
* checking R files for non-ASCII characters ... OK
* checking R files for syntax errors ... OK
* checking R code for possible problems ... OK
* checking contents of 'data' directory ... OK
* checking data for non-ASCII characters ... OK
* checking data for ASCII and uncompressed saves ... OK
* checking for unstated dependencies in tests ... OK
* checking tests ...
  Running exp-bioinf.R
  Running exp-hoehenried-data.R
  Running exp-hoehenried.R
  Comparing exp-hoehenried.Rout to exp-hoehenried.Rout.save ... OK
 OK
* checking package vignettes in inst/doc ...
  Creating bioinf-repro.tex from bioinf-repro.Rnw
  Comparing bioinf-repro.tex to bioinf-repro.tex.save ... OK
* collecting all results into bioinfrepro-check.pdf ... OK
 OK
* elapsed time (check, wall clock): 1:28
```

Figure 4: Logfile of running the *R*2 checks on the paper package for [3].

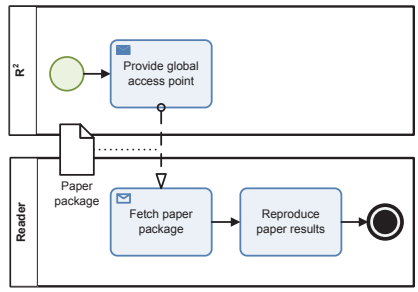

Figure 5: Process model of locally reproducing the results of a paper by a reader.

see [20] for details. At the end, all results are automatically combined into a single PDF file which can easily be downloaded from our server.

Figure 4 shows the logfile for checking the paper package of manuscript [3]. The first part checks that the paper package ist structurally valid and can be used on all major computing platforms without problems (no special characters in file names etc.). The main part is running the tests, i.e., R code that claims to reproduce the results in the paper. All output (figures, tables) is automatically merged into a single PDF document with corresponding subsections. For one example file (`exp-hoehenried.R`) there is also pre-computed output. The check compares the new output and reports any differences to previous runs. Once the pre-computed output is validated against the manuscript differences in future computations can be detected automatically.

In this case we also have the Sweave version of the manuscript which also checks OK. In most cases one would submit either the Sweave vignette *or* R code for tests. But of course it is possible to have both, e.g., the Sweave file for the paper itself, and R code for intermediate computational results not shown directly in the paper (simulations, etc.).

3.3. Local Reproduction of Results

There are many situations when a referee or reader wants to reproduce results on his or her own machine, e.g., to try a new method on ones' own data, or to learn about it by replicating results from others. Figure 5 shows the process model of locally reproducing the results of a paper by a reader (or reviewer). Building the R^2 framework on top of the R packaging system makes this process simple and elegant for the reader; in fact, it is equivalent to the common process of installing an arbitrary R package and reading its vignette:

```
> install.packages("papername", repos = "R2url")
> edit(vignette("papername"))
```

An obvious (future) extension of this process is to allow interaction between the reader and the R^2 platform. This enables, among other things, the reader to comment and to rate articles – which gives the readership (who are in fact the most qualified) a forum to judge about the importance of any particular paper. Similar procedures are already implemented by some journals.

4. Summary

R already has a widely used format for executable papers named Sweave, which in its original version embeds R code into LaTeX documents. The R^2 platform extends this in several ways and provides web services which are very easy to use for beginners. Sweave files usually depend on other files like data sets or R code. R^2 paper packages collect all this pieces and arrange them in a structured way such that we can do automated computations on them. The package can either be created manually like a regular R package, or by uploading the pieces to our web server (similar to uploading files into an editorial management system) which then creates the package. In the simplest case only R code and data are uploaded, the final PDF document is then created using templates on the server.

The R^2 platform offers advantages for authors, reviewers and readers of scientific manuscripts. Authors who submit a paper package can see if their results are reproducible on a system independent from their own computing environment. This also allows to pass on software in double blind review processes. Meta-information in the

DESCRIPTION file controls which parts can downloaded by whom during the review process. Reviewers need not execute code that has been verified by the R^2 server, but they can easily do so in case they want to. And finally readers of a manuscript have direct and convenient access to all electronic supplementary material for a paper as an R package.

[1] J. Buckheit, D. Donoho, WaveLab and reproducible research, statistics Department, Stanford University, CA, USA (1995).
URL http://www-stat.stanford.edu/ donoho/

[2] J. de Leeuw, Reproducible research: the bottom line, statistics Program, Unniversity of California, Los Angeles, CA, USA (2001).
URL http://preprints.stat.ucla.edu/

[3] T. Hothorn, F. Leisch, Case studies in reproducibility, Briefings in BioinformaticsAccepted for publication on 2010-12-01. doi:10.1093/bib/bbq084.

[4] R Development Core Team, R: A language and environment for statistical computing, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-07-0 (2010).
URL http://www.R-project.org

[5] K. Hornik, F. Leisch, Vienna and R: Love, marriage and the future, in: R. Dutter (Ed.), Festschrift 50 Jahre Österreichische Statistische Gesellschaft, Österreichische Statistische Gesellschaft, 2002, pp. 61–70, iSSN 1026-597X.

[6] S. Theußl, U. Ligges, K. Hornik, Prospects and challenges in R package development, Computational StatisticsAccepted for publication. doi:10.1007/s00180-010-0205-5.
URL http://dx.doi.org/10.1007/s00180-010-0205-5

[7] F. Leisch, Sweave: Dynamic generation of statistical reports using literate data analysis, in: W. Härdle, B. Rönz (Eds.), Compstat 2002 — Proceedings in Computational Statistics, Physica Verlag, Heidelberg, 2002, pp. 575–580, iSBN 3-7908-1517-9.
URL http://www.stat.uni-muenchen.de/ leisch/Sweave

[8] D. E. Knuth, Literate programming, The Computer Journal 27 (2) (1984) 97111.

[9] F. Leisch, A. J. Rossini, Reproducible statistical research, Chance 16 (2) (2003) 46–50.

[10] R. C. Gentleman, V. J. Carey, D. M. Bates, B. Bolstad, M. Dettling, S. Dudoit, B. Ellis, L. Gautier, Y. Ge, J. Gentry, K. Hornik, T. Hothorn, W. Huber, S. Iacus, R. Irizarry, F. Leisch, C. Li, M. Maechler, A. J. Rossini, G. Sawitzki, C. Smith, G. Smyth, L. Tierney, J. Y. Yang, J. Zhang, Bioconductor: Open software development for computational biology and bioinformatics, Genome Biology 5 (10) (2004) R80.1–16.
URL http://genomebiology.com/2004/5/10/R80

[11] G. J. Williams, Rattle: A Data Mining GUI for R, The R Journal 1 (2) (2009) 45–55.

[12] R. Koenker, A. Zeileis, On reproducible econometric research, Journal of Applied Econometrics 24 (5) (2009) 833–847. doi:10.1002/jae.1083.

[13] P. Vandewalle, J. Kovacevic, M. Vetterli, Reproducible research in signal processing [what, why, and how], IEEE Signal Processing Magazine (2009) 37–47.

[14] K. A. Baggerly, K. R. Coombes, Deriving chemosensitivity from cell lines: Forensic bioinformatics and reproducible research in high-throughput biology, The Annals of Applied Statistics 3 (4) (2009) 1309–1334. doi:10.1214/09-AOAS291.

[15] K. A. Baggerly, K. R. Coombes, E. S. Neeley, Run Batch Effects Potentially Compromise the Usefulness of Genomic Signatures for Ovarian Cancer, Journal of Clinical Oncology 26 (7) (2008) 1186–1187. doi:10.1200/JCO.2007.15.1951.

[16] H. K. Dressman, A. Berchuck, G. Chan, J. Zhai, A. Bild, R. Sayer, J. Cragun, J. Clarke, R. S. Whitaker, L. Li, J. Gray, J. Marks, G. S. Ginsburg, A. Potti, M. West, J. R. Nevins, J. M. Lancaster, An Integrated Genomic-Based Approach to Individualized Treatment of Patients With Advanced-Stage Ovarian Cancer, Journal of Clinical Oncology 25 (5) (2007) 517–525. doi:10.1200/JCO.2006.06.3743.

[17] V. J. Carey, V. Stodden, Reproducible research concepts and tools for cancer bioinformatics, in: M. F. Ochs, J. T. Casagrande, R. V. Davuluri (Eds.), Biomedical Informatics for Cancer Research, Springer, 2010, pp. 149–175.

[18] A. J. Rossini, R. M. Heiberger, R. Sparapani, M. Mächler, K. Hornik, Emacs speaks statistics: A multiplatform, multipackage development environment for statistical analysis, Journal of Computational and Graphical Statistics 13 (1) (2004) 247–261.

[19] Object Management Group, Inc., Business Process Modeling Notation, v2.0 (January 2011).
URL http://www.bpmn.org

[20] R Development Core Team, Writing R Extensions, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-11-9 (2011).
URL http://www.R-project.org