Creating statistical reports in the past, present and future

PhD candidate Gergely DARÓCZI (daroczig@rapporter.net)
BCE, Hungary

ABSTRACT

The paper summarizes the most important milestones in the recent history of computer-aided data analysis, then suggests an alternative reporting workflow to the traditional statistical software methods by the means of an R package implementing statistical report templates with annotations in plain English.

Keywords: *R, reports, reproducible research, literate programming*

This paper first provides a brief historical overview of a variety of statistical and reporting tools actively used by practicing data analysts in the past 100 years (Daróczi-Tóth, 2013). That time covered lots of changes both in methodology and in tools: existing methods were improved and new ones were also discovered, on the other hand mainframes, personal computers and nowadays cloud computing became the standard source of processed data instead of statistical tables and the slide rule.

This also changed the way how statistics and data analysis are used by an ever growing number of experts, laymen and industries: it is no surprise nowadays to do e.g. customer segmentation without any deep theoretical knowledge on how k-means cluster or latent-class analysis really works.

Doing data analysis means something different compared to what Karl Pearson did in the past: in our days, statistical wizards and data-driven decision-making tools can help us to run valid analysis on live databases – even on mobile platforms. For this end, the paper also proposes a way on how to create annotated reproducible statistical templates and reports in R.

DEVELOPMENT OF COMPUTATIONAL HARDWARE RESOURCES

After the appearance of mechanical computers and the first programmable devices, the development of the first electronic computer, the Atanasoff-Berry Computer (ABC), began at the University of Iowa (1937). Sadly, the long lasting process failed to produce the expected results, and the ABC wasn't able to complete all the scheduled tasks at the end of the project (1942).

But not much later (1946), the first general purpose digital computer, the Electronic Numerical Integrator And Computer (ENIAC), was successfully built at The University of Pennsylvania for the United States Army. This was the first working programmable digital computer in human history, and although the machine weighted 30 tons, it was a real state of the art technology in the 1940's: the performance was even better than the Mark II's (developed at Harvard, funded by the United States Navy) that was not capable of storing programs internally.

The successor of the ENIAC was built under the supervision of János Neumann (1949). The Electronic Discrete Variable Automatic Computer (EDVAC) was already equipped with a central controlling unit and internal memory, which is an important milestone in the history of computer science. Particularly because Neumann (1993) had already published his results before the time of the presentation of the EDVAC, there were already multiple devices working on the basis of the same concept – including the first computer storing programs, the Electronic Delay Storage Automatic Calculator (EDSAC), which was built at the Cambridge University.

At this time, there were already a number of universities around the world working on similar projects, and even the first devices with business purposes had appeared. The first commercially sold universal computer was the UNIVAC (UNIVersal Automatic Calculator), and it was actually used at the US Bureau of Statistics from 1951 (Stern, 1981), where they had already been using electronic devices before.

Another historically significant tool was the punched card reader, which was first used for tabulating the 1890 census, developed by Herman Hollerith and the Tabulating Machine Company, which is considered as the precursor of IBM (Truesdell, 1965). The punched card reader was a major success both in the United States and in Europe, however because of a strong rise in rental prices, the US Census Bureau started to look for an alternative solution.

First, they tried to cooperate with Simon North, however that didn't bring any success, so after the request from James Powers and John Mauchly, the Bureau started to support the development of UNIVAC. It's well known, that after its launch (1951), the device had great success: besides the statistical bureau and many institutions of the US Army, multiple business companies started to also use these computers, including e.g. ACNielsen.

Next to the UNIVAC, another machine worth mentioning on the market at that time was IBM's "mainframe" product family. Although it was launched in the 1950s, the real success of mainframes is connected to the second generation of the products in the 1960s (Renfro, 2004). These

machines (IBM 7090/7094) were increasingly stable, due to the air cooling system instead of the old oil cooling one, and were used e.g. by the NASA in the Apollo programs.

IBM announced the "System/360" model in 1964. The strongest version was capable of completing tens of thousands of tasks per second, and it's relatively large memory (8 MB) brought great success for its designers. The business success was partially based on the elimination of the software compatibility problems, the programs became portable, and even more: those programs still work today on any of the IBM servers from the zSeries product family.

Compared to the first vacuum tube computers and second generational transistor computers mentioned earlier, the invention (1958) and mass production (1960s) of integrated circuit was a fundamental change, and was crucial for the appearance of the third generational devices with increased computational capacity. The fourth generation of computers was even more integrated, and the performance was sometimes a hundred times better compared to the IBM machines' from the 1960s.

The invention of microprocessor, the possibility to transmit data between computers and the decreasing hardware prices, all led to the appearance of personal computers. IBM released the first PCs in the early 1980s and multiple companies (like Xerox, Hewlett Packard, Apple or Commodore) followed the success story.

The history of computer hardware from this point is familiar to most of us, as PCs and related technologies have become part of our everyday life. As computers are getting smaller and smaller, we tend to require those even for the most common tasks, and their role in statistics is obviously crucial. Nowadays, we use notebooks or laptops, smart phones, PDAs and most recently tablets for everyday actions, usually with direct internet access.

But these new devices have limited computational resources due to increased mobility, which results in limited support for the most recent statistical methods. A possible solution for this technical problem is to reassign the tasks that require more computational resources to server computers, which led to the birth of cloud technology and online services. The basic concept behind this is to store the data and algorithms on a secure server, while the connecting clients can easily run the queries on mobile devices without any local unnecessary load. This paper will present an example of such infrastructure beside e.g. Shiny (RStudio, 2014) or OpenCPU (Ooms, 2014).

DEVELOPMENT OF STATISTICAL SOFTWARE

Not only did the increased computational power changed how data analysis and reporting works nowadays, but ever since the beginnings of probability theory or e.g. the creation of the least squares method, more and more robust, multivariate methods have been created in the recent few hundred years' history of demography and statistics. E.g. instead of the analytical explanation and deduction, now we usually apply more practical methods with simulations. To understand this significant change in statistical theory, this paper also provides a brief overview on the evolution of statistical software packages.

The first econometric software was developed at Cambridge (Renfro, 2004). Although the EDSAC was already able to run econometric programs in 1953, but it was only used to execute basic operations until the late 1950s, so the general use of statistical software only began later.

All the software created at this time were designed to accomplish a specific task, and the users had hard time migrating data from a software to another, because those were not prepared to be compatible with each other. To present this problem with a simplified example: an ANOVA program written in machine language may have required a completely different structure for input data, than a program creating e.g. cross tables – even if both were written for the same computer, in the same programming language.

The first generation of statistical software packages only appeared in the mid 1960s, and they overcame the earlier problems by providing a complex solution: the user could apply a variety of statistical methods on the same structured data (Leeuw, 2011).

The more than 30 years long career of BMD and later the BMDP (BioMeDical Package) started in 1965 at the UCLA Department of Medicine. This originally free statistical program was created for calculations regarding health sciences. Later it became a proprietary product until acquired by SPSS Inc. (1996), when the development of the software was discontinued.

The SPSS (Statistical Package for the Social Sciences) software is well known amongst social scientists, which was released in 1968 by the University of Chicago. The success can be measured by the fact that Wellmann (1998) noted the user manual of SPSS as one of the most influential books (Nie, 1970). At that time, SPSS was exclusively focused on social sciences, and it started to expand towards other fields only later, when acquired by IBM (2009). Since then, SPSS refers to "Statistical Product and Service Solutions" along with PASW (Predictive Analytics SoftWare). The program was originally created to process punched cards and to work only from command line, but it has later

established its own file structure (sav), a graphical user interface (1985), then a Java based, platform independent version (2007). Nowadays, there are a variety of add-ons besides the "Base" package to help the users accomplishing a wide range of statistical tasks – let it be the creation of questionnaire, designing samples or a summary of the results.

The SAS (Statistical Analysis Software) package is similarly wide spread and well known; however it's mainly focused on business processes. It was first released at the North Carolina State University (1968), and by now it's one of the biggest service providers in the business intelligence industry beside MicroStrategy, IBM Cognos, Oracle Hyperion, Microsoft BI and SPSS Modeler. The foundations of SAS were laid down by a former student of NCSU, who began to develop a framing structure after the implementation of ANOVA and multivariate linear regression (1966). The popularity of the package was partly due to the fact that the developing team managed to efficiently deal with missing data. There were several important milestones in the development of SAS, like the first platform independent release in the early 1980s, supporting a variety of mini (not mainframe) computers, then the change-over from the PL/I, FORTRAN and machine languages to the C programming language. By now, SAS also provides a server hosted, so called "on-demand" service.

Leeuw (2011) dates the appearance of the second generation of statistical software packages to 1985, when a graphical user interface was added to all above mentioned software, and apart from those, new ones also appeared on the market, focusing mainly on fine-tuning the graphical interface and user experience.

Data Desk was released in 1986 for Macintosh computers with the primary goal of helping exploratory data analysis. The main advantage was the user friendly and interactive user interface, which allowed even the non-experts to achieve spectacular results. Since 1997, it has been available for Windows as well; however the development has been discontinued after all.

Not much later (1989), JMP (jump) was also released for Macintosh devices by one of the co-founders of SAS. The developers focused on improving the graphical user interface, which resulted in interactive plots and graphics for exploratory data analysis.

A significant part of the success of STATA (1985) is due to its community and the user activity, as STATA made it possible and easy to use and reference STATA codes uploaded to the internet as user contributed code in "ado" format. This community and user base is relatively large, and e.g. the STATA mailing list has an extraordinary traffic (more than a thousand e-mails per month) compared to the software mentioned earlier or to any other

commercial statistical software. STATA is still under active development, and it also has a graphical user interface since 2003.

The S programming language was started by John Chambers, and it was actively used as early as the late 1970s in the internal network of Bell Laboratories. The great advantage of S was, compared to the earlier FORTRAN programs written for specific tasks, that it used standard commands to do interactive data analysis a statistical modeling, and these functions and statistical methods were also easily accessible for the developers. The computer program was later ported UNIX (1980) from the General Comprehensive Operating System designed for mainframe computers, and the release of the program (1981) and later the source code (1984) also guaranteed success for its successors, like R. The "New S" language was released at the late 1980s with real support for functions instead of the macros used before, new graphical devices (X11 and PostScript) became available, and the "formula-notation", S3 and later the S4 methods were also introduced at that time that are used even today.

Although S is still available today, some alternative implementations became far more popular among the users. For example according to the TIOBE-index measuring the popularity of different programming languages, R is amongst the top 30 most used programming languages, and the commercial version of S (S-PLUS) has been also in the top 100 multiple times.

The R language was developed based on the SCHEME language created by Gerald Jay Sussman and on the results of S (Hornik, 2012). The rewriting of SCHEME functions and features began in 1993 at the University of Auckland with the leadership of Ross Ihaka and Robert Gentleman. The fact that John Chambers, the author behind the original idea and development of S, is also in the R Development Core Team, probably indicates the significance of the success of R as well.

This is open source software: free to use, distribute and modify under the GPL v2 license and it is also supported by the Free Software Foundation as being part of GNU. The core programming environment available for multiple operating systems (Windows, Macintosh and Linux) for free, and what's more, by now, many different types of graphical user interface and front-ends help everyday's work of R users. Integrated development environments (like Eclipse/StatET, Emacs/ESS, Rstudio, TextMate, Notepad++, etc.) are also available.

Besides the fact that it is free, the great success of R also due to CRAN (Comprehensive R Archive Network), which is a central repository of user contributed packages. By now, there are over 5000 libraries on CRAN, and they mostly cover all the currently available theoretical statistical methods.

Although anybody can upload new packages to CRAN, and the operators of the network only run automated tests on those, the great number of users, the active community (GitHub, StackOverflow, and e.g. the [R-help] mailing list with more than 3000 messages per month etc.) and their constant feedback guarantees the maintenance and further development of the software. The R Core Development Team also officially supports the stable operation of base libraries, for example R has became a standard and certified statistical tool for clinical trials (The R foundation, 2012).

Besides those presented above, there are also many other business software packages (MATLAB, Mathematica, Statistica etc.) available on the market, however as they are not too significant considering the subject of this paper, we won't discuss them in further detail.

LITERATE PROGRAMMING

Similarly to how the development of general statistical software and environments supplanted the use of custom computer programs created for specific tasks, the workflow of creating reports had also changed. For example, reproducibility of research results became more and more important also in programming, just like in natural sciences. First, this resulted in detailed comments in source code, later source code started to appear in research paper, which later ended up in mixed texts including e.g. statistical commands in so-called chunks inside of papers, which were evaluated and replaced by the results while compiling the document. We believe that this method resulted in significant change in writing reports, as there is no further need to do data analysis and formatting texts etc. with different tools, but all these tasks can be managed within a single piece of software, so that the user can concentrate on the real scientific tasks, not on the software environment.

The first implementation of such literate programming tool (Noweb) was released in 1994 (Johnson, 1997), and soon an R implementation was also published, called Sweave (Leisch, 2002). *Sweave* is a great tool, but it only supports the pdf file format requiring LaTeX knowledge with a rather steep learning curve, so several similar alternatives also appeared on CRAN later. There are packages provide support for HTML, Open/LibreOffice, MS Word etc. document formats; an almost complete list can be found on the dedicated CRAN Task View (Zeileis, 2005). Despite the considerable amount of already existing similar tools, in 2011, two independent R developers and teams decided to build new packages to provide an alternate way for literate programming.

The main motive of Yihui Xie (2012) with *knitr* was to replace *Sweave* with increased functionality (like caching) and support for multiple file formats.

This later feature and the general ease of use gained a huge interest for *knitr* in the R community, which became one of the most trending R package.

Gergely Daróczi and Aleksandar Blagotić (2012) started to work on a similar alternative in 2011, which development resulted in the *pander*, *rapport* and *rapportools* packages. Unlike *knitr*, the original vision of these packages was not to provide an alternative tool for literate programming, but we decided to create an environment that supports textual statistical templates in plain English.

These templates can be considered similar to R functions that can be applied to any dataset in R, but the results are markdown-formatted textual reports instead of R objects. For example the already existing ANOVA template can produce the usual tables and graphs with annotations in plain English, so that any student could understand the results of the applied analysis run in the background.

REPRODUCIBLE STATISTICAL TEMPLATES

From a technical point of view, all these templates are made of two important parts. The first one ("header") defines meta-data of the statistical template, where the author can provide details about the goal and the output of the template, the list of required R packages and a few examples of usage, similarly to the R documentation, and the latter part ("body") contains all the plain text in English to be shown to the user along with the R expressions to be evaluated.

The header also includes the optional or required inputs, which are to be provided by the user on call. One may consider these inputs as R function arguments, e.g. the user can pass data frames or vectors, numbers and string, also any number of similar options to the template to be processed later. The inputs may be any R class from character, complex, factor, integer, logical, numeric or raw, and several attributes might also be defined for those. A simple example for such header part:

```
<!--head
meta:
  title: Rapport demo
   description: This is POC demo on the usage of rapport
templates
  packages:
  - ggplot2
 - pander
inputs:
- name: v
  label: Variable to analyse
  required: yes
  class: numeric
  length:
    min: 1.0
    max: 1.0
- name: color
  label: Color of the histogram
  standalone: yes
  value: red
  class: character
head-->
```

This header was written in YAML syntax, which is a human-readable format to represent hierarchical list data. Beside the title and description fields, we have listed two required R packages for this template, then listed two inputs from which only one is compulsory. This prior input requires a numeric vector to be passed, while the latter is an optional string which defaults to "red" if omitted. Anything written after the closing "`head-->`" tag belongs to the body of the template:

```
# A quick analyis on <%= v.name %>

The mean of <%= v.name %> is <%= mean(v) %> and the
standard deviation is <%= sd(v) %>. Let us also
check the frequency table:

<%= table(v) %>

## Tables are boring!

<%=
set.caption(paste('Histogram of', v.name))
hist(v, xlab = v, col = color, main = '')
%>
```

This part starts with a markdown formatted, Atx-style header which also includes an R expression: to return the name of the variable passed as "`v`", which was defined in the header above. Please note that there is no need to `cat`, `print` or e.g. `xtable` the R objects we generate, as all R code chunks are automatically passed to the `pander` S3 method, which turns any R object to human-readable Pandoc's markdown (MacFarlane, 2012) that can be easily exported to various document formats. The rest of the body follows a similar syntax: plain text in regular English is mixed with some R expressions in code chunks.

Now let's see the results of this very basic report template applied on the transmission variable in the `mtcars` bundled dataset of R (Henderson–Velleman, 1981):

```
> library(rapport)
> rapport('demo.rapport', data = mtcars, v = 'am')

# A quick analyis on am

The mean of am is _0.4062_ and the standard
deviation is _0.499_. Let us also check the
frequency table:
-------
 0   1
--- ---
19  13
-------
```

```
## Tables are boring!

![Histogram of am](plots/rapport-demo-rapport-1-1.png)
```

So after loading the package, the `rapport` function takes a few arguments: the first one refers to the template to be used, then a data frame is passed. The `v` parameter stands for the first input defined in the header, and we omit the second input, which will default to "red". All the R chunks were processed by `rapport`, the returning R objects were transformed to markdown, and the histogram was saved to disk, which can be easily included in other document formats like MS Word docx, pdf or HTML. To automatically create and open such documents at one go, use e.g. `rapport.docx`, `rapport.pdf` or `rapport.html` instead of `rapport`. For more details, please check the documentation.

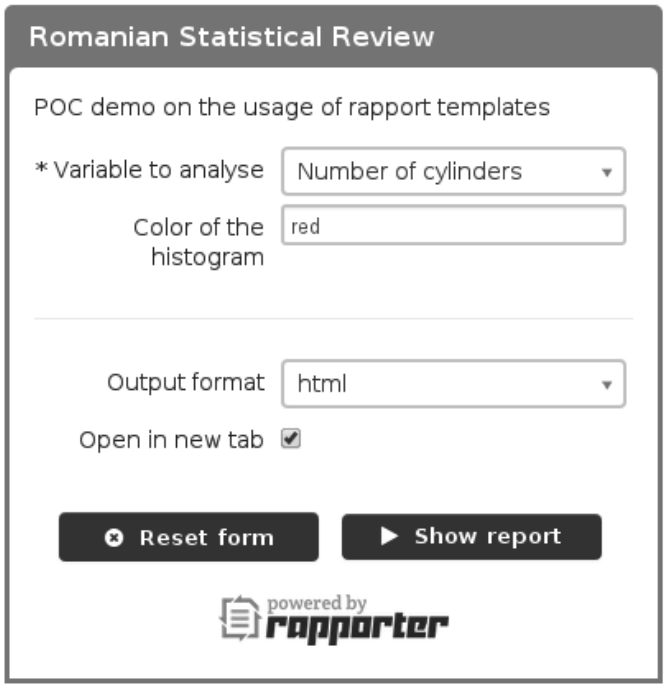
REPORTING IN THE CLOUD

Although running a *rapport* template is as easy as seen above, this process still requires a working R and Pandoc installation on the user's computer beside some R knowledge to load the data and to run the above few commands. For this end, we have created a cloud environment hosting such report templates that can be evaluated in any Internet browser even on mobile devices and without any prior R or statistical knowledge. A proof of concept demo front-end of the above defined template can be seen on Figure 1, which can be also accessed online at **http://bit.ly/1kWpqMl.**

This tool provides a simple way to create similar front-ends and user interfaces for R developers, and it also let non-experts use the reporting templates integrated into any web- or mobile application – which is already trending in the data analysis sector.

Web application front-end to the statistical report template

Figure 1



References

1. Chambers, J. M. [1980]: Statistical Computing: History and Trends. The American Statistician. 34(4): 238–243.
2. Daróczi, G. [2012]: sandboxR: filtering malicious R calls. https://github.com/Rapporter/sandboxR
3. Daróczi, G. [2013]: pander: an R Pandoc Writer. CRAN.
4. Daróczi, G. – Blagotić, A. [2013]: rapport: an R Templating System. CRAN.
5. Daróczi, G. – Tóth, G. [2013]: Felhőtlen statisztika a felhőben. Hungarian Statistical Review. 91(11): 1118–1142.
6. Daróczi, G. [2014]: rapportools: Miscellaneous (stats) helper functions with sane defaults for reporting. CRAN.
7. Francis, I. [1981]: Statistical Software: A Comparative Review. Elsiever. New York.
8. Leisch, F. [2002]: Sweave: Dynamic Generation of Statistical Reports Using Literate Data Analysis. In: Proceedings in Computational Statistics. Physica Verlag. Heidelberg. 575–580.

9. Henderson, V. [1981]: Building Multiple Regression Models Interactively. Biometrics. 37(2): 391–411.
10. Hornik, K. [2012]: The R FAQ. CRAN.
11. Johnson, A. L. – Johnson B. C. [1997]: Literate Programming Using Noweb. Linux Journal. 42: 64–69.
12. Jong, V. J. de. [1989]: A Specification System for Statistical Software. Centrum voor Wiskunde en Informatica. Amsterdam.
13. Leeuw, J. [2011]: Statistical Software: An Overview. In: Lovric, M. (ed.): International Encyclopedia of Statistical Science. Springer. Berlin. pp. 1470–1473.
14. MacFarlane, J. [2012]: Pandoc: A Universal Document Converter. http://johnmacfarlane.net/pandoc/
15. Nie, N. H. – Bent, D. H. – Hull, C. H. [1970]: SPSS: Statistical Package for the Social Sciences. McGraw-Hill. New York.
16. Ooms, J. [2013]: The RAppArmor Package: Enforcing Security Policies in R Using Dynamic Sandboxing on Linux. Journal of Statistical Software. 55(7): 1-34.
17. Ooms, J. [2014]: opencpu: OpenCPU framework for embedded statistical computation and reproducible research. CRAN.
18. R Development Core Team [2014]: R: A Language and Environment for Statistical Computing. R Foundation for Statistical Computing. Vienna. http://www.r-project.org/
19. R Foundation for Statistical Computing [2012]: R: Regulatory Compliance and Validation Issues. A Guidance Document for the Use of R in Regulated Clinical Trial Environments. http://www.r-project.org/doc/R-FDA.pdf
20. Renfro, C. G. [2004]: Computational Econometrics: Its Impact on the Development of Quantitative Economics. IOS Press. Amsterdam.
21. Renfro, C. G. [2009]: The Practice of Econometric Theory: An Examination of the Characteristics of Econometric Computation. Springer. Berlin.
22. RStudio (2014). shiny: Web Application Framework for R. CRAN.
23. Routh, D. A. [2007]: Statistical Software Review. British Journal of Mathematical and Statistical Psychology. 60(2): 429–432.
24. Stern, N. B. [1981]: From Eniac to Univac: Appraisal of the Eckert-Mauchly Computers. Digital Press. Bedford.
25. Valero-Mora, P. M. – Ledesma, R. [2012]: Graphical User Interfaces for R. Journal of Statistical Software. 49(1): 1–8.
26. Von Neumann, J. [1993]: First Draft of a Report on the EDVAC. IEEE Annals of the History of Computing. 15(4): 27–75.
27. Wellman, B. [1998]: Doing It Ourselves: The SPSS Manual as Sociology's Most Influential Recent Book. In: Clawson, D. (ed.): Required Reading: Sociology's Most Influential Books. Amherst: University of Massachusetts Press. Amherst. 71–78.
28. Xie, Y. [2012]: knitr: A General-Purpose Package for Dynamic Report Generation in R. CRAN.
29. Zeileis, A. [2005]: CRAN Task Views. R News. 5(1): 39–40.