Preamble and installation

Antonio Gasparrini

01 November 2023

Contents

This chapter introduces first provides an introduction to the R software, then describes its origin as a language and then its development as a fully-fledged environment for statistical computing and graphics, and illustrates the advantages of R and why you should use it. The chapter then provides an overview of the course, before describing how to install it and the RStudio software, and integrated systems for R. Finally, the chapter descrived the graphical user interfaces (GUIs) of both R and Rstudio.

What is R?

R is often described as an *open source and free software environment for statistical computing and graphics*. This definition intends to characterize R as something more than a statistical program. In fact, R can be defined as a fully planned, integrated and coherent system, more than an inflexible collection of tools for data analysis. Also, R is considered a proper programming language itself, although much of the source code of the software is written in basic languages, such as C.

The R language and software is a collaborative effort with contributions from all over the world. Its first version was written by Robert Gentleman and Ross Ihaka, two researchers at the University of Auckland, New Zealand. R is considered a dialect of the S language, which has been principally developed by Rick Becker, John Chambers and Allan Wilks while at Bell Laboratories. The S language has been commercially implemented in the software S-plus. R was named partly after the idea of a different letter of the alphabet for a development of S, and partly after the initials of the first two authors.

From 1997, R has been developed and maintained by the R Development Core Team, and from 2003 supported by the R Foundation, a not for profit organization working in the public interest. The Development Core Team and Foundation take care of distributing and documenting the software, promoting its use, and protecting its copyright. The R project is also supported by a vast community of users, which plays an important part in the development of the software, in particular regarding the implementation of novel statistical methodologies, and also contributes to producing documentation or organising emailing lists or forums.

The software is part of the GNU Project, and released under the GNU General Public Licence. This license guarantees the free distribution of the software and ensures that derived works are distributed under the same license terms. The open source nature of the software allows everybody to copy, change or extend it, providing a reference to the original implementation and that the resulted work is kept as open source. In particular, R is distributed free of charge, mainly from the Comprehensive R Archive Network (CRAN), directly through the main web site (http://www.r-project.org/) It is a multi-platform software, with distributions for Windows, macOS and Unix/Linux.

The R software is structured in *packages*, namely libraries including code, data and documentation. Packages are usually organized as collections of tools for specific statistical tasks, although packages for other purposes, such as mapping, internal calls to other software or merely data storage, are also available. In addition to the standard packages included in the main distribution, thousands of other packages have been written and can be downloaded through R from the CRAN, or other sources.

The computing philosophy of R, inherited from S, is rather different from that of other statistical programs and is based on the definition of *objects*. While most of the computations in other systems such as Stata or SPSS focus on a single dataset, in R multiple datasets can be simultaneously loaded or created in the session, together with other data structures (*e.g.*, vectors or lists). Also, R is an *interpreted* language, meaning that the computations are carried out through a series of progressive steps, where commands are individually executed and their results saved in new objects. This approach is different than in *compiled* languages, such as that adopted by SAS, where a full program needs to be written and executed. In R, intermediate results, saved in objects, can be accessed and displayed, and then used for further computations.

Why using R?

The most obvious reason you may want to use R is, of course, because it is free, although this is not the only important advantage. Being free means that you can install or update the software on all your laptops or PCs, without any concern about buying licenses. This feature can be important for researchers in developing countries or in small academic institutions or companies, for whom the price to be paid for commercial software is substantial. Even more importantly, other than saving money, using freely-available software guarantees that your analysis and results can be openly shared with others.

The open-access and free nature of R provides other significant benefits. While the development and support for commercial software are configured through a seller-customer relationship, a peer-to-peer scheme among users is preferred and applied in R. The added value comes from the existence of a community, where users are actively engaged in extending the software and documentation, participating to help with emailing lists and forums, and in testing the implementation of established or novel statistical and non-statistical tools. At the time of writing, there are 19,646 packages available on CRAN, written by users from all over the world. If you plan to work on an innovative statistical technique, it might be the case that someone has already provided functions or entire packages about it. Also, coherently with the open-access philosophy, most of the documentation is freely available, together with examples of code for different statistical methods.

The fact that the analysis is performed by creating and managing different objects is often reported as a limitation in using R. The language is based on a syntax that largely prevents the use of menu-based commands or dialogue boxes, as in other statistical programs (*e.g.*, Stata or SPSS), and forces the user to input expressions in the command-line interface. Also, the user is expected to define the steps for producing the desired computation and, especially for basic tasks such as data management, some of the computations are not always pre-determined by the use of specific functions, but sometimes need to be programmed. This implies a steeper learning curve for beginners.

However, such initial complexity is compensated by extended flexibility in the use of the software. The same computation can be often produced by several alternative ways, calibrated on your expertise, length or clarity of the code, and computing time. If you do not remember or even know the function to perform a specific computation, you can find your own way to produce the same result through a set of alternative

steps. As explained earlier, intermediate results are stored in objects and can be checked, accessed or displayed during the analysis, and re-used later. It is much simpler to spot errors in your code if compared to other statistical programs. The output is usually minimal and tailored to the specific aim of the analysis.

The features of R detailed above makes programming in R a simple task, also for beginners. Indeed, the distinction between standard computing and programming is blurred in R, given the same simple and intuitive syntax is used in both cases. Writing new functions to simplify the code, speed up the analysis or perform sophisticated computations does not require any additional knowledge.

In the last few years, several tools have been developed to facilitate the use of R, in particular for inexperienced users. A useful addition is the availability of graphical user interfaces (GUIs), which improve and extend the default R's GUI. Among other choices, a well-developed GUI is provided by another freely-available and open-source software, RStudio (http://rstudio.org/) RStudio is an integrated development environment for R, which includes a wide range of productivity-enhancing features. In particular, several tasks not easily accomplished with a command-line interface can be performed through menus and dialogue boxes provided by the RStudio's GUI. Similarly to R, RStudio runs on all major platforms (Windows, macOS, Unix/Linux).

About these notes

These notes are designed for an introductory course on the R software. However, you can easily use them as a help to learn R on your own, or as a reference for further study.

While the standard R documentation is fairly technical, these notes are written following a more practical perspective, while at the same time describing the basic concepts underlying the architecture and philosophy of the R software and language. The notes include several examples including R code, embedded in the text using Rmarkdown, a typesetting program built to integrate software input and output with text.

The main aims of these notes are:

- to progressively introduce you to the main concepts and functionalities of the R software and language;
- to show you how to interact with R and run simple and more sophisticated computations;
- to illustrate tools for helping you progress independently in learning R.

These notes can be seen as a preliminary reading, providing the basic information to overcome the initial steep learning curve typical of software and programming languages. Several important topics in data analysis are intentionally omitted, focusing instead on the basic usage of R.

It is assumed that you are running R in a Windows operating system. Most of the information identically applies to other systems, such as macOS or Unix/Linux, in particular regarding the examples of R code. However, practical issues about the program, such as installing or managing packages, may change depending on the specific operating system. Also, the description focuses on running R through RStudio. However, the use of the standard R or other GUIs only requires minor changes to the information provided here.

Please send comments and suggestions, or report errors and problems to antonio.gasparrini@lshtm.ac.uk.

Installing R and RStudio

The pre-compiled binaries of the R software can be installed for free from the web site at http://www.r-project.org. You can follow the link DOWNLOAD R, or click on CRAN in the menu on the left. You will be asked to choose a mirror, and then the operating system (Windows, Mac, Linux). Under Windows, click on the link BASE and follow the instructions of the installation wizard to install the pre-compiled binary distribution. Other links in these web pages provide additional information. Also, see the Manual "R Installation and

Administration", available at http://cran.r-project.org/manuals.html, for a thorough but quite technical overview about installation, maintenance and update.

RStudio can be easily installed from the web site http://rstudio.org/, clicking on the link for downloading and then following the instruction of the installation wizard. During the installation, it automatically links and integrates with R.

The R and RStudio GUI

After the installation, you can launch R by running the executable from the Start menu or double-clicking on the R icon on the desktop. What you see now is the default graphical user interface (GUI) of R, as shown below.

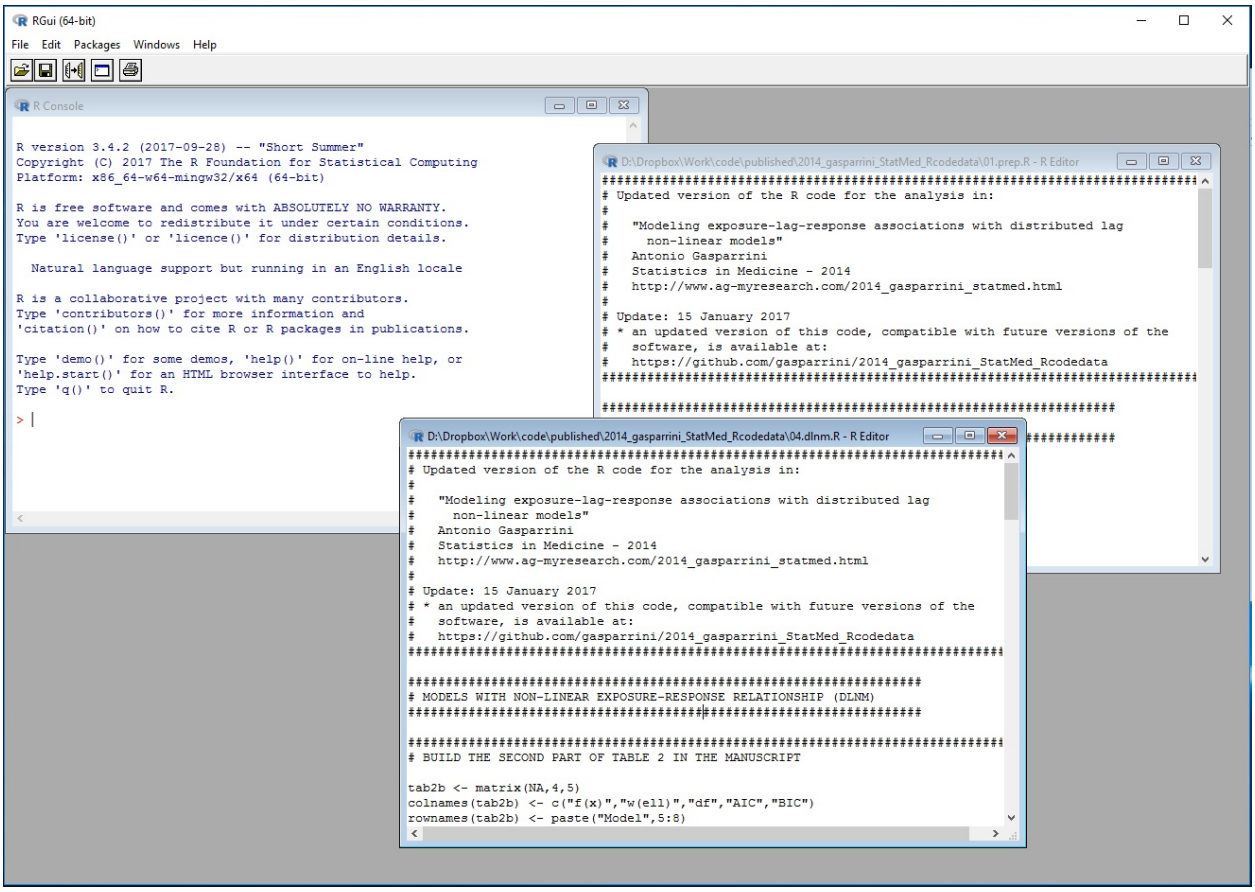

Figure 1: The R graphical user interface (GUI) in Windows

The interface of R is quite bare if compared to other software. The main part is constituted by the Console sub-window (top left in the figure above), where the commands can be inputed, and the results are shown. Also, you can pursue a few tasks using the menu on the top, for example, loading or saving sessions or accessing the help tools. The two other sub-windows shown in the figure above are R *scripts*, a text file where lines of commands can be saved and run. More script sub-windows can be opened simultaneously in the interface. Graphical devices, where plots produced by R are shown, are opened in the main window as well (not shown).

The default GUI allows full control of the capabilities of R. However, the interaction with the program is almost exclusively performed through written commands inputted in the console or via scripts. Other software,

such as Stata or SPSS, offers a structured set of menus and dialogue boxes which simplifies the life of users. The computing language of R is more flexible than in other programs: this feature provides a great potential for creative and efficient computing and programming, but nonetheless prevents the development of comprehensive menu-driven tools, as mentioned earlier.

The GUI of RStudio provides a set of facilities to address these limitations. Among other advantages, the user can exploit menus and dialogue boxes for saving R files or graphs, handling packages or accessing the documentation. Also, a better editor is provided, with colours identifying different parts of scripts or results, and parenthesis and quote matching. RStudio provides an integrated and structured environment for R, with several other facilities, such as tools for creating R packages or writing R documentation pages and LaTeX documents with embedded R code (incidentally, these notes have been entirely written in RStudio). The RStudio interface is reproduced in the figure below.

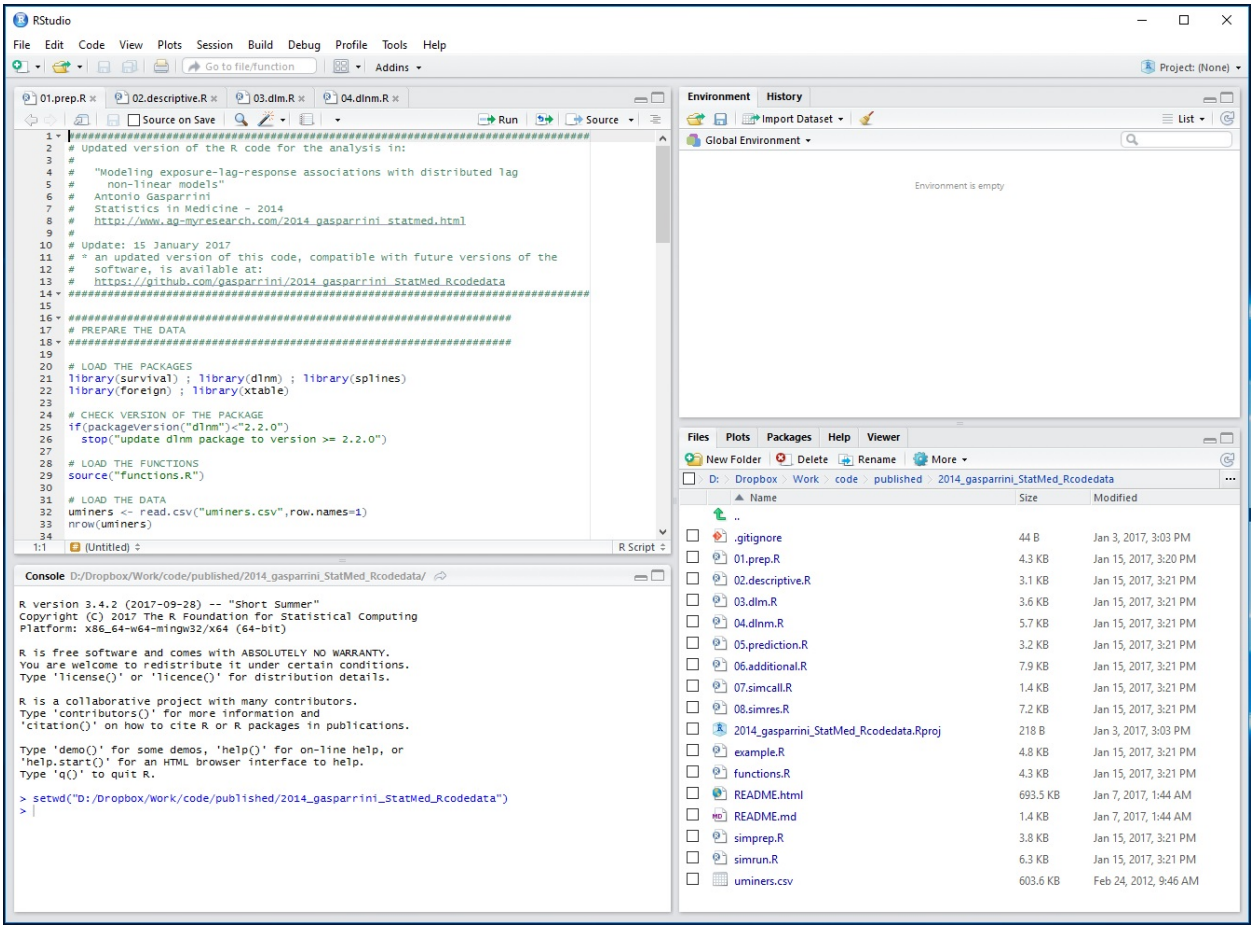

Figure 2: The graphical user interface (GUI) of RStudio

The RStudio GUI consists of four panels, plus a set of menus. Each of the panels includes one or more windows, with the main ones listed below:

- Top left panel. The EDITOR (or Source) window contains the scripts used in the session, and optionally other viewers. It allows multiple scripts, optionally referring to different directories, to be easily managed and searched.
- Bottom left panel. The CONSOLE window has the same features as the default R GUI seen above.
- Top right panel. The ENVIRONMENT window shows the R objects created in the session and allows sessions to be saved or loaded and datasets to be imported. The HISTORY window stores all the

commands inputted in the Console, which can be browsed and moved to the Console again or added to a script. Also, a command history can be saved or loaded.

- Bottom right panel. The FILES window provides a file manager, usually showing the content of the working directory. The PLOTS window helps managing plots, which are stored chronologically and can be opened in an external window or exported in different formats. The PACKAGES window lists all the installed packages, which can be loaded into the session, and allows the installation of new packages or update of the existing ones. The HELP window stores the help HTML pages opened during the session in chronological order.

The main functionalities of the RStudio and R GUI, summarized above, will be illustrated in the next sessions. You can refer to the R and RStudio documentation for further information.