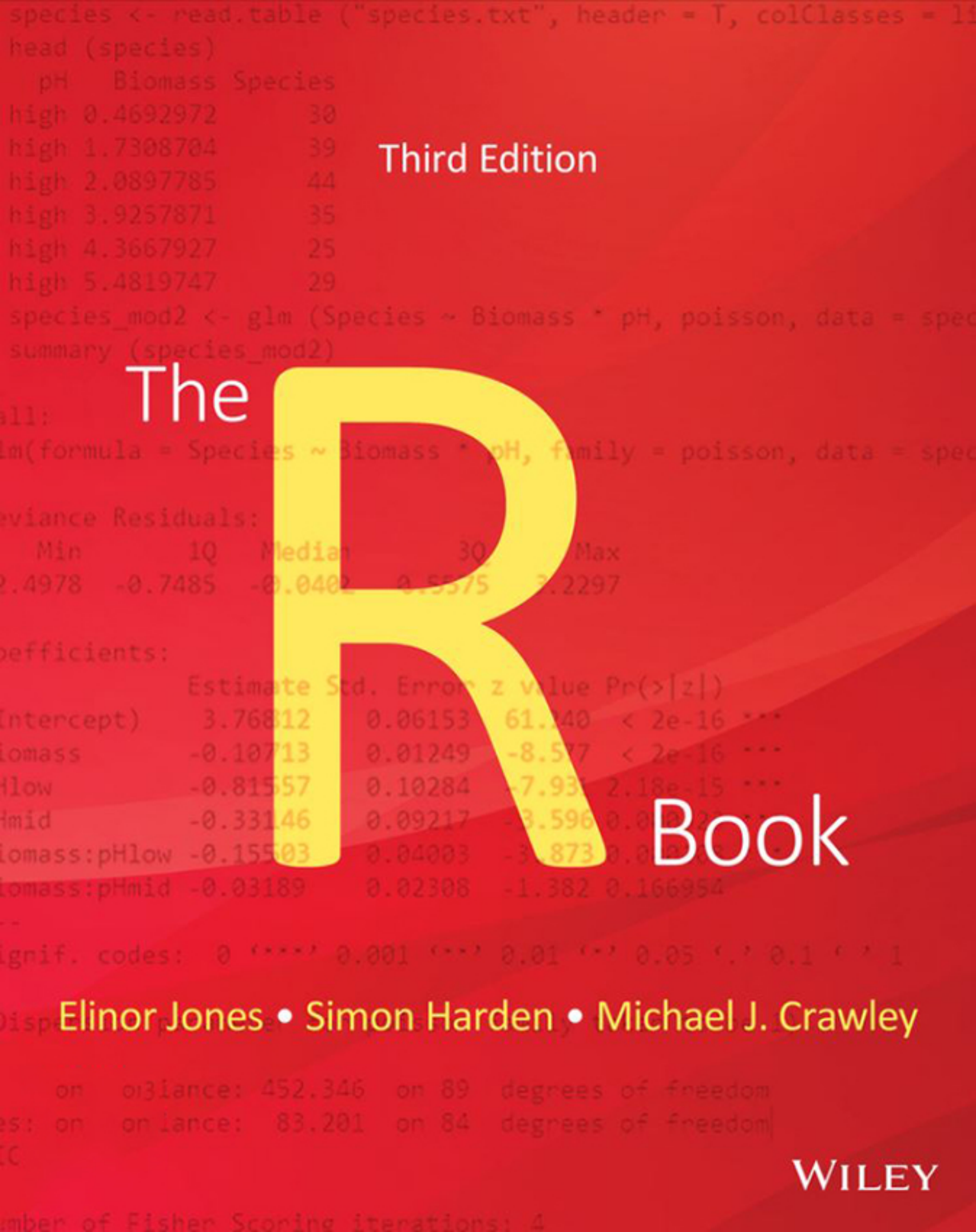
Third Edition
The R Book
Elinor Jones • Simon Harden • Michael J. Crawley
WILEY

The R Book

The R Book

Third Edition

Elinor Jones
University College London, UK

Simon Harden
University College London, UK

Michael J. Crawley
Imperial College London, UK

WILEY

This third edition first published 2023

Edition History: John Wiley & Sons Ltd (1e, 2007; 2e, 2013)

Registered Offices
John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

Editorial Office
9600 Garsington Road, Oxford, OX4 2DQ, UK

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Library of Congress Cataloging-in-Publication Data

Names: Jones, Elinor (Associate Professor), author. | Harden, Simon, author. | Crawley, Michael J., author.
Title: The R book / Elinor Jones, Simon Harden, and Michael J. Crawley.
Description: Third edition. | Hoboken, NJ : Wiley, 2022. | Includes bibliographical references and index.
Identifiers: LCCN 2022008352 (print) | LCCN 2022008353 (ebook) | ISBN 9781119634324 (cloth) | ISBN 9781119634409 (adobe pdf) | ISBN 9781119634430 (epub)
Subjects: LCSH: R (Computer program language) | Mathematical statistics–Data processing.
Classification: LCC QA276.45.R3 J662 2022 (print) | LCC QA276.45.R3 (ebook) | DDC 005.5/5–dc23/eng20220528
LC record available at https://lccn.loc.gov/2022008352
LC ebook record available at https://lccn.loc.gov/2022008353

Cover design: Wiley
Cover image: Courtesy of Simon Harden; © enjoynz/Getty Images

Set in 10/12pt HelveticaLTStd by Straive, Chennai, India

Contents

Detailed Contents

List of Tables

Preface

R is the most powerful tool in the known universe for carrying out statistical analysis, and it's free! This book is aimed at those who wish to carry out such work – exploring, plotting, and modelling data – but who do not have much experience in *R* and/or statistics. *R* is described from scratch with instructions for loading and getting going with the software in Chapter 1 and a description of its essential elements in Chapter 3. Later chapters discuss statistical methods and are written so that they can be used either as a beginner's guide or as a reference manual on particular procedures in *R*. The theory behind the analyses is covered in enough depth, we hope, to make it comprehensible but without overburdening the reader with too much mathematics. The datasets used to illustrate various analyses are available at https://www.wiley.com/go/jones/therbook3e.

Using *R* has become far simpler with the introduction of RStudio, which is also free (other editors are available). RStudio provides a friendly front end and easy access to tools, all of which seem a long way from *R*'s original rather forbidding command prompt. This book assumes the use of RStudio rather than using *R* directly, but the code presented will work using the latter setup too.

While there is still the usual hurdle of getting to know powerful software, the benefits, particularly in graphics and modelling, far outweigh the effort. Academic papers in many disciplines routinely use and report results using *R*. In addition, the open-source nature of the software means that users have added extra functionality by writing packages to broaden *R*'s capabilities. There are currently over 18,000 packages that, together with useful links and information, can be found at the official *R* distribution site, CRAN: https://cran.r-project.org/.

This book is contingent upon the existence of *R*. Those involved are too numerous to mention, but we are hugely grateful to all involved in its creation and continuing evolution. When you use *R*, *R* packages (e.g. `spatstat`), and RStudio, please cite them. Up-to-date citation details for each of these can be found by typing the following in *R*, respectively:

```
citation ()
citation ("spatstat")
RStudio.Version ()
```

Elinor Jones
Simon Harden
Michael J. Crawley
August 2022

Acknowledgments

This book would not exist without its previous editions so thanks, firstly, to the originating author, Michael J. Crawley.

It has been a pleasure to revise The R Book to create this third edition. We are very grateful to Professor Crawley for allowing us to use materials from previous versions, including his fantastic array of datasets that make a welcome return in this edition.

Finally, we would like to thank the Department of Statistical Science at University College London for giving us time and space to complete the book during a difficult period for everybody.

Elinor Jones
Simon Harden
August 2022

About the Companion Website

This book is accompanied by a companion website.

www.wiley.com/go/jones/therbook3e

This website include: Datasets

1

Getting Started

1.1 Navigating the book

The material covered in this book has been arranged by topic. The first few chapters cover the essentials, including basic technical knowledge (Chapter 2), the fundamentals of *R* (Chapter 3), and data handling in *R* (Chapter 4). Subsequent chapters deal with statistical procedures, including graphics (Chapters 5 and 6), statistical testing (Chapter 9), and common statistical models (from Chapter 10).

To make navigating the book easier, the following conventions will be used:

- New terms are highlighted in **bold** when first used;
- *R* functions and function arguments written in-line are highlighted in red, for example the `plot ()` function and the `pch` argument (note the use of the round brackets when referring to functions);
- Stand-alone *R* code is written in red, with output in blue, for example:

```
1+3

[1] 4
```

- Datasets, variable names, model names, and so on, are written in `typewriter font`;
- *R* packages (see Section 1.6) are highlighted in blue, for example `MASS`.

1.1.1 How to use this book

This book is intended to serve a wide audience from complete beginners through to those in need of an *R* reference manual. Below, we offer advice on how to use the book depending on level of experience in statistics and computing.

The R Book, Third Edition. Elinor Jones, Simon Harden and Michael J. Crawley.

Companion website: www.wiley.com/go/jones/therbook3e

Beginner in both computing and statistics

The book is structured principally with such a reader in mind. There are six key things to learn: how to arrange data, how to read data into *R*, how to check data once within *R*, how to select an appropriate analysis, how to interpret the output, and how to present the analysis for publication. A thorough understanding of the basics is essential before trying to do the more complicated things, so we recommend studying Chapters 3 to 4 carefully to begin with. Do all of the exercises that are illustrated in the text on your own computer.

Now comes the hard part, which is selecting the right statistics to use. Model choice is extremely important and is the thing that will develop most with experience. Don't be afraid to ask for expert help with this. Never do an analysis that is more complicated than it needs to be, so start by reading about graphical representations of data (Chapters 5 and 6). Sometimes this is all that's needed.

Student needing help with project work

A good understanding of variable types is key (broadly, variables are either numeric or categorical, see Section 1.8.1). An analysis of a dataset will depend – at least in part – on the type of variables in the dataset and the research question of interest. Does the research question point to a particular 'response' variable, and if so, what type of variable is this?

From here, the first port of call is to plot or tabulate the data, depending on the nature of the variables (see Chapters 5–7). That might be enough in itself, or further statistical analyses might be needed. For example, if the response variable (if any) is a count, consider using hypothesis tests (Chapter 9), tables (Chapter 7), or possibly a model (Chapter 11). If the response variable is a continuous measure (e.g. a weight), then consider using hypothesis tests (Chapter 9), or a regression model (Chapter 10).

Done some R and some statistics, but keen to learn more of both

The best plan is to skim quickly through the introductory material in case there is anything new to be learned. It is a good idea to read Chapter 3 on the fundamentals of the *R* language and Chapters 5 and 6 on graphics. Much of the rest of the book is organised by analysis type making it easy to jump to the relevant chapter.

Done regression, but want to learn more advanced statistical modelling

For readers who have experience of regression in another language, the best plan is to go directly to Chapters 10 and 11 to see how the output from linear models is handled by *R*. Familiarity with data input and dataframes is essential (Chapter 4), then the chapters on more advanced modelling should be accessible.

Experienced in statistics, but a beginner in R

The first thing is to get a thorough understanding of dataframes and data input to *R*, so start with Chapter 4. Then, chapters on statistical modelling should be accessible. It is a good idea to browse, for example Chapters 9 (Testing) and 10 (Regression) to understand the output from *R*. Working through Chapters 5 and 6 will provide the foundations of graphics in *R*.

Experienced in computing, but a beginner in R

Well-written *R* code is highly intuitive and very readable. The most unfamiliar parts of *R* are likely to be the way it handles functions and the way it deals with environments. It is impossible to anticipate the order in which more advanced users are likely to encounter material and hence want to learn about specific features of the language, but vectorised calculations, subscripts on dataframes, function-writing and suchlike are bound to crop up early (Chapter 3). When faced with an unfamiliar name in some code, just type the name immediately after a question mark; for example to find out more about the `rnbinom ()` function, type:

```
?rnbinom
```

Recognizing mathematical functions is quite straightforward because of their names and the fact that their arguments are enclosed in round brackets ( ). Subscripts on objects have square brackets [ ]. Multi-line blocks of *R* code are enclosed within curly brackets { }. The idea of lists might be new, or applying functions to lists; elements within lists have double square brackets [[ ]].

Look at the sections at the start of Chapter 3 as a starting point. The index is probably the most sensible entry point for queries about specifics.

Familiar with statistics and computing, but need a friendly reference manual

For information about a *topic*, use the chapter list and the Detailed Contents to find the most appropriate section. For aspects of the *R language*, look at the sections mentioned at the start of Chapter 3. Spending time browsing the contents of general material such as Chapters 5 and 6 on graphics is a good idea.

Get used to *R*'s help pages. Each *R* function has a help page which can be accessed by typing a question mark followed directly by the function name. To find out what all the graphics parameters mean, for instance, just type:

```
?par
```

1.2 *R* vs. RStudio

R is a powerful open-source software for statistical computing (R Core Team, 2021). It can be used directly, or for a more pleasing user experience, can be used via the RStudio interface which is freely available (RStudio Team, 2020). We strongly recommend using RStudio rather than *R* directly as it makes managing workspace easy and avoids some of *R*'s rather cumbersome features. The rest of the book will assume the use of RStudio, but all code presented will work – and provide identical results – if used in 'native' *R* instead.

We will generally write '*R*' instead of 'RStudio' throughout this book.

1.3 Installing *R* and RStudio

You will need to install both *R* and RStudio. Both will run under Windows, a number of flavours of Linux (more for *R* than for RStudio so check the links below) and even Apple's Mac OS X.

First download and install *R*. Note that this needs to be done first before attempting to download RStudio.

- Go to the CRAN project webpage https://cran.r-project.org/mirrors.html and choose the closest CRAN site to you (e.g. Imperial College London). It doesn't matter too much which of these is chosen if several are close by;
- Select the link for *Download R for …*, for your operating system;
- Follow the instructions, noting that the default set-up is perfectly adequate.

Now download RStudio.

- Go to the RStudio webpage https://rstudio.com/products/rstudio/download/#download and select the 'Download' for RStudio Desktop. The free version is generally adequate and is certainly so for this book.
- Follow the instructions. At some point you'll be asked to locate where *R* has been installed. Remember that RStudio is just an *R* interface.

Rather than downloading *R* and RStudio, there is the option of accessing the software online via RStudio Cloud (https://www.rstudio.com/products/cloud/). There are some advantages to using RStudio online, for example when working on a group project. However, for individual use, it is likely that downloading *R* and RStudio is the best way forward.

1.4 Using RStudio

Once installed, open RStudio. The screen is divided into three parts as in Figure 1.1.

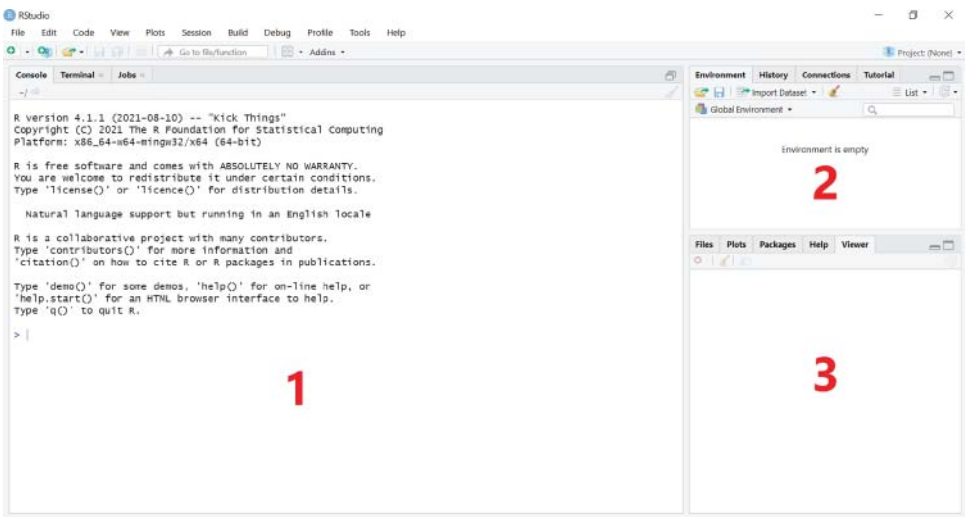

Figure 1.1 RStudio windows. RStudio, PBC

On the left is the **console**, numbered 1 in Figure 1.1. This is *R*. All code will be passed to the console where it will be executed, and numerical output will also be displayed here. The console displays the version number of *R*, its date, and version name (always comedic).

In the top right-hand corner is the **workspace**, numbered 2 in Figure 1.1. This is the control centre and gives an at-a-glance overview of what has been done so far in the session.

The bottom right corner, numbered 3 in Figure 1.1, hosts a number of things. Switch between them by clicking on the relevant tabs:

- **Files**: Shows the accessible file directories (more on this in Section 1.8.4);
- **Plots**: This is where plots and other graphics will be displayed;
- **Packages**: Lists packages that have been installed and provides functionality for installing others (more on this in Section 1.6);
- **Help**: As the name might suggest, help with various functions or procedures can be found here (more on this in Section 1.7);
- **Viewer**: Used for viewing local web content.

1.4.1 Using *R* directly via the console

Before exploring further, we'll return to the console. Below the header – which contains useful information about version number, citation, and a health warning – is a blank line with a `>` symbol in the left-hand margin. This is called the **prompt** and is *R*'s way of saying 'What now?'. Commands can be typed in directly here, though we suggest a more efficient way of working via **text editors** (see Section 1.4.2).

To begin with, we can use the console as a calculator, for example typing in the following command then pressing enter on the keyboard to execute the command (for neatness, we don't present the `>` at the start of each line of code in this book):

```
2 + 3

[1] 5
```

When working, a `+` is sometimes displayed at the left-hand side of the screen instead of `>`. This means that the last command typed is incomplete. The most common cause of this is forgetting one or more brackets. If what's missing is clear (e.g. a final right-hand bracket), then just type the missing character and press enter, at which point the command will execute. If a mistake has been made, then press the Esc key and the command line prompt `>` will reappear. Use the Up arrow key to scroll through previous commands, then use the Left and Right arrow keys to navigate to the mistake and correct it.

1.4.2 Using text editors

Writing commands in the console directly is rarely a good idea. It is good practice to keep a record of the code we use, which makes correcting mistakes, updating analyses, or just running the command(s) again very easy. RStudio has a built-in text editor to store and execute code.

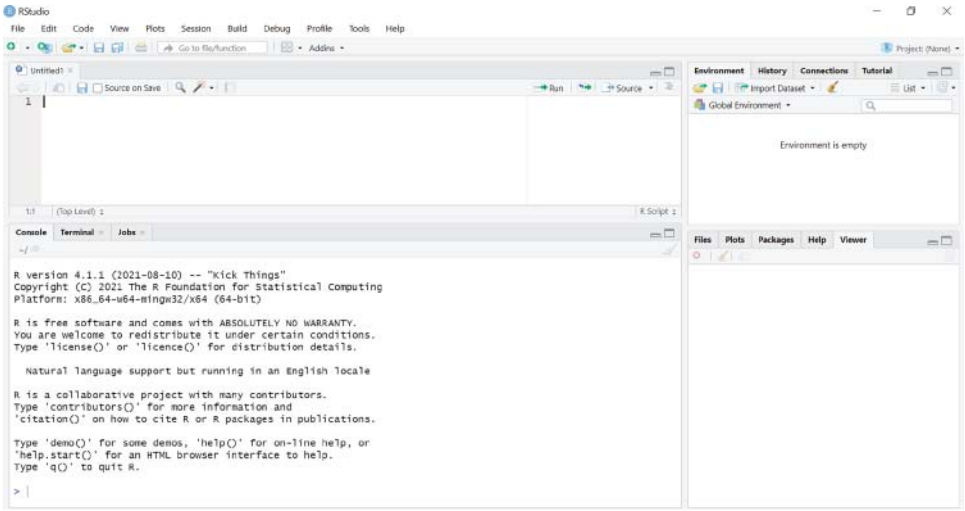

Figure 1.2 RStudio windows with text editor. RStudio, PBC

A failsafe way of opening a blank text editor, which doesn't depend on the operating system of your machine, is to go to *File*, then *New File*, then *R Script*. It will appear in the top left hand corner as in Figure 1.2.

The text editor is where we write commands, using a new line for each one. Click the text editor to activate it, before writing the following:

```
2 + 3
3 * 6
exp (2)
```

To run commands, highlight the relevant lines and click 'Run' (top right corner of the text editor) or press Ctrl and Enter, simultaneously. Output will be displayed in the console (bottom left). For readability, output will be shown directly beneath each of the relevant command throughout this book like this:

```
2 + 3

[1] 5

3 * 6

[1] 18

exp (2)

[1] 7.389056
```

The commands so far request calculations to be performed, but we can also define **objects** (see Chapter 3 for details), for example `a` is assigned (`<-`) the value 5 while we define `b` to be ln(10):

```
a <- 5
b <- log (10)
```

When we run this, we notice two things:

- there is no output in the console (because all we've done is define two objects);
- the **Environment** tab in the workspace has been populated with the definitions of `a` and `b`.

It is helpful to understand that *R* is an **object-orientated** programming (OOP) language: it is based on applying *actions* (commands) on *objects*. For example, a dataset which we load into *R* (see Chapter 4) will be considered by *R* as an object. Any action applied (e.g. finding the mean for each variable in the dataset) will be an action on a particular object (the dataset in this case). An object doesn't have to be a dataset, however, as we saw above.

The best way to learn *R* and RStudio is to play with them. The introduction here gives a very brief overview but is in no way complete. A good place to start is with RStudio cheatsheets https://www.rstudio.com/resources/cheatsheets/.

1.5 The Comprehensive *R* Archive Network

CRAN https://cran.r-project.org/ is the first port of call for everything to do with *R*. It is from here that you download and install *R* (see Section 1.3), find contributed packages to solve particular problems (see Section 1.6), find the answers to frequently asked questions, read about the latest developments, get programming tips, and much more besides.

It is well worth browsing through *The R Journal*, accessible via the CRAN webpage. This is the refereed journal of the *R* project for statistical computing. It features short- to medium-length articles covering topics that might be of interest to users or to developers of *R*, including

- **Add-on packages**: Short introductions to or reviews of *R* extension packages.
- **Changes in *R***: Details of recent changes to *R*.
- **Applications**: Demonstrating how a new or existing technique can be applied in an area of current interest using *R*, providing a fresh view of such analyses in *R* that is of benefit beyond the specific application.

1.5.1 Manuals

There are several manuals available on CRAN, for example (descriptions are taken from the webpage):

- *An Introduction to R* gives an introduction to the language and how to use *R* for doing statistical analysis and graphics.
- A draft of the *R Language Definition*, which documents the language *per se* – that is, the objects that it works on, and the details of the expression evaluation process, which are useful to know when programming *R* functions. This is perhaps the most important of all the manuals.

- *Writing R Extensions* covers how to create your own packages, write *R* help files, and use the foreign language (C, C + +, Fortran, …) interfaces.
- *Data Import/Export* describes the import and export facilities available either in *R* itself or via packages which are available from CRAN.
- *R Installation and Administration*, which is self-explanatory.

These manuals are also available in RStudio by going to the Help tab in the bottom right-hand corner and clicking the 'home' icon.

The most useful part of the site, however, is the Search facility. This is a good starting point for investigating the contents of most of the *R* documents, functions, and searchable mail archives.

1.5.2 Frequently asked questions

R has three collections of answers to FAQs:

- the *R* FAQ, which contains useful information for users on all platforms (Linux, Mac, Unix, Windows);
- the *R* Mac OS X FAQ for all users of Apple operating systems;
- the *R* Windows FAQ for all users of Microsoft operating systems.

Read the first of these, plus the appropriate one for your platform.

1.5.3 Contributed documentation

This contains a wide range of longer (more than 100 pages) and shorter manuals, tutorials, and exercises provided by users of *R*. You should browse these to find the ones most relevant to your needs.

1.6 Packages in *R*

A lot can be done with *R* or RStudio 'straight out of the box', also known as **base-*R***. Table 1.1 lists some of the packages that come supplied as part of the base-*R* installation.

Table 1.1 Libraries used in this book that come supplied as part of the base package of *R*.

Package name	Functionality
`lattice`	graphics for panel plots or trellis graphs
`MASS`	package associated with Venables and Ripley's book entitled *Modern Applied Statistics using S-PLUS*
`mgcv`	generalised additive models
`nlme`	mixed-effects models (both linear and nonlinear)
`nnet`	feed-forward neural networks and multinomial log-linear models
`spatial`	functions for kriging and point pattern analysis
`survival`	survival analysis, including penalised likelihood

However, there is a huge community of *R* users who contribute to its functionality via **packages**. A package contains additional functionality for *R* that can be loaded during a session. Navigating contributed packages can be tricky simply because there are so many of them, and the name of the package is not always as indicative of its function as one might hope.

Viewing existing packages can be done in RStudio by clicking on the Packages tab. Clicking the box next to a package loads it. A far better way of loading a package is to do so via the `library ()` function, which also means we have it as part of our code. For example, to load the `MASS` package (Venables and Ripley, 2002), which has a wide range of useful functions and datasets, type:

```
library (MASS)
```

See Section 1.6.3 for information on installing new packages.

1.6.1 Contents of packages

It is easy to use the `help` function to discover the contents of library packages. Supposing that we wanted to find out about the contents of the `spatial` package, we'd type:

```
library (help = spatial)
```

This brings up general information about the package in a new tab of the text editor in RStudio, including a list of all the functions and data sets.

To find out how to use, say, Ripley's *K* (`kfn ()`) from `spatial`, we load the package and then use `?` to query the function:

```
library (spatial)
?Kfn
```

1.6.2 Finding packages

There is no comprehensive cross-referenced index of packages, but there is a very helpful feature called 'Task Views' on the CRAN website, which explains the packages available under a limited number of usefully descriptive headings. Click on Task Views to see bundles of packages assembled by topic. Currently, there are 40 Task Views on CRAN as listed in Table 1.2.

Click on the Task View to get an annotated list of the packages available under any particular heading. If Base-*R* doesn't cover your needs, it is highly likely that a package exists that does.

1.6.3 Installing packages

The base package does not contain some of the libraries referred to in this book, but installing these is very simple.

It is best to install packages using the `install.packages ()` function, as shown below, rather than doing so via RStudio's Packages tab (therein, click on install, then search for the package needed). The packages used in this book are

```
install.packages ("akima")
install.packages ("boot")
```