Ben Marwick

---

# Archaeological Science with R

# Contents

# Part I

# Getting started

# One

# Welcome

## 1.1 Overview

The goal of "Archaeological Science with R" is to give you a solid foundation into using R to do archaeological science. By archaeological science, I mean, systematic, objective, and empirical research into past human behaviours using material remains and traces. The goal is not to be exhaustive, but to instead focus on what I think are the critical skills for doing archaeological science with R. Some archaeologists already use R in an *ad hoc* way, for a quick plot here, or a linear model there. But this book will show you how R can be at the center of your entire research workflow, from when you start working with raw data until your final report is complete.

The specific problem is that this book aims to solve is that the majority of archaeologists receive little or no training in scientific programming for data analysis and visualization. A command-line interface program such as R is unfamiliar and exotic to most archaeologists, yet R is well-equipped to perform almost every analytical and visualization task an archaeologist could need. Instead, most archaeologists use Microsoft Excel, SPSS, or similar commercial point-and-click software. The problem with this is that is it limits the development of new methods because the you are limited to the built-in suite. Commercial software also makes it difficult to demonstrate and ensure reproducibility due to the ephermeral nature of point-and-click interfaces. Commercial software limit transparency in research because the algorithms behind the functions are not available for inspection.

The major textbooks and handbooks of quantitative archaeology (Baxter

2003; 1994, Van Pool and Leonard 2011, Drennan 2009, Shennan 1997) are excellent for relevant statistical theory, discussion, and examples, but have three limitations that the proposed book will address.

First is the absence of 'plug-and-play' examples, leading the reader to invest substantial effort to implement a method described using formulae, creating many opportunities for error. Van Pool and Leonard 2011 instruct the reader to do their statistical analyses by hand, an impractical and outmoded recommendation. The amount of effort required by these texts is prohibitive to exploration and experimentation of new methods. This book includes reproducible examples using real data sets so you can easily step-through the analysis to explore different parameters, and easily interchange your own data.

Second, the existing books give little coverage to intuitive and methodologically more robust methods such as resampling and Bayesian analyses, favoring traditional parametric methods. This reflects a time when computational power was scarce, and computations often done by hand or with a calculator (cf Fletcher and Lock 1991). Modern computers can now easily handle as resampling and Bayesian methods, and R is unique in having a mature set of methods for computing common tests in these frameworks. These methods are also increasingly common in the published research literature. This book introduces these alternative approaches to give you more options in your analytical toolkit.

Third, the currently available books are silent on the practical mechanics of many common data analysis and visualization tasks in archaeology that are not traditionally considered statistics, such as displaying and analyzing stratigraphic information from excavation and spatial data from surveys. Learning a tool-chain for these tasks is traditionally done on your own, and often at great expense with proprietary software. This book demonstrates how to write simple programs that are especially useful for archaeologists. By doing this, and providing code for other common tasks, this book addresses the need for instruction in a comprehensive open source tool-chain for these common tasks in archaeological science. It does this by presenting a reproducible research environment to show how R and its contributed packages can be used for start-to-finish research into survey, excavation, and laboratory data. My hope is that practical and accessible introduction to reproducible research, such as this book aims to provide, will improve

openness and transparency in archaeology generally.

This is not a book of detailed discussions of statistical theory, and you will be referred to other authoritative texts on technical details of algorithms, etc. This is a book introducing R for practical and efficient implementations of common tasks in archaeological science, and serves as a springboard to more advanced R programming. The specific topics covered in this book are:

- Implementing reproducible research as a practice that is good for science generally, and good for your individual productivity
- Working with common archaeological data structures, ingesting them into R and manipulating them from messy formats to tidy formats ready for analysis
- Visualising spatial data by making maps, doing spatial analysis and site classification
- Visualising and analysing stratigraphic data from archaeological excavation
- Working with relative and absolute chronologies by doing seriation, and calibrating, analysing and visualising radiocarbon dates
- Key methods for working with stone artefacts, faunal remains, pottery, glass, and metal artefacts
- Compositional analysis using cluster and principle components analyses of multivariate data
- Simplifying collaborative research with version control
- Writing for publication more efficiently with literate programming

## 1.2 Why do archaeologists need to use code?

In recent years, biologists have seen great increases in volumes of genomic data, due to improvements in sequencing technology. This has lead them

to search for more efficient ways to analyse their data, and automate their analyses. They found that spreadsheet programs did not provide enough flexibility to conduct their analyses. As a result, many have turned to R, Python, and similar programming languages to gain access to a much wider variety of analytical options. The key detail here is a shift from the point-and-click interface of a spreadsheet program, to a command-line interface of a programming language. A command-line interface refers to interacting with software by sending instructions to the program as lines of plain text.

While archaeologists don't have the same urgency to manage a data deluge in the same way biologists have with genomic data, we can benefit from their efforts to use programming languages for scientific research. There are two big benefits to using a programming language instead of a point-and-click interface.

1.

## 1.3 Why R?

If you are new to R, you might wonder what makes learning such a quirky language worthwhile. To me, some of the best features are:

- It's free, open source, and available on every major platform. As a result, if you do your analysis in R, anyone can easily replicate it.
- A massive set of packages for statistical modelling, machine learning, visualisation, and importing and manipulating data. Whatever model or graphic you're trying to do, chances are that someone has already tried to do it. At a minimum, you can learn from their efforts.
- Cutting edge tools. Researchers in statistics and machine learning will often publish an R package to accompany their articles. This means immediate access to the very latest statistical techniques and implementations.
- Deep-seated language support for data analysis. This includes features likes missing values, data frames, and subsetting.

- A fantastic community. It is easy to get help from experts on the R-help mailing list, stackoverflow, or subject-specific mailing lists like R-SIG-mixed-models or ggplot2. You can also connect with other R learners via twitter, linkedin, and through many local user groups.
- Powerful tools for communicating your results. R packages make it easy to produce html or pdf reports, or create interactive websites.
- A strong foundation in functional programming. The ideas of functional programming are well suited to solving many of the challenges of data analysis. R provides a powerful and flexible toolkit which allows you to write concise yet descriptive code.
- An IDE tailored to the needs of interactive data analysis and statistical programming.

## 1.4 Who should read this book

This book is written for archaeologists who are keen to expand their analytical horizons and gain access to new methods and new, more efficient, ways of organising their research process. No prior knowledge about R or computer programming is necessary, but a curiosity about statistics would be an advantage, as well as a readiness to read beyond this book to make decisions about the suitability of statistical methods for your specific research questions.

## 1.5 Getting help

Developing fluency in a language like German or Chinese takes time and has ups and downs. Learning R is a similar process, and you should anticipate ups and downs as R becomes an increasingly central part of your workflow. Fortunately, there are a number of options for seeking help when you get stuck, or have a question. The simplest case is when you execute a line of code, only to get a cryptic error message in reply. Your first action might be to copy the text of the error message to your clipboard, and paste it into Google. In most cases, the top search results will be from Stack Overflow (`http://stackoverflow.com/`) or one of the R mailing lists (`https://www.r-project.org/mail.html`).

Stack Overflow is a free question-and-answer website that is focussed on computer programming. Participants receive points for asking clear questions, and for giving useful answers. This adds a competitive element to participating in the Q&A, and sometimes when you ask a well-formed question, several answers will appear very quickly from participants eager to earn points. The R mailing lists are more traditional email-based fora, with archives at several places online. Highly skilled R programmers and R-using scientists from a variety of disciplines are active on both the R mailing lists and Stack Overflow. I tend to prefer Stack Overflow because I find it easier to see when a question has been answered, compared to browsing the email list archives.

I have found that in the course of writing my question to submit to Stack Overflow, the process of simplifiying my code into a small, self-contained example leads me to discover the cause of my problem (often a misplaced comma or bracket) and I can answer my question before I need help from others. Similarly, I often find that many of my questions have already been asked and answered on Stack Overflow - the challenge is to recognise how similar an existing question is to your current problem, and thus how useful the answers are to your specific issue.

If you are sure that your problem is new and unique, and you want to submit a question, here is some general advice to help you get a useful answer quickly from the Stack Overflow and R mailing lists communities:

- Spend some time browsing previous posts to understand the community norms. Online communities have specific cultural values, and while some of these are spelled out in posting guides, many are also unwritten, and can only be learnt through mindful reading of previous correspondence. If you are familiar with the norms, you are more likely to have an efficient and satisfying interaction with the community.
- Make sure you have the latest version of R and of the package (or packages) you are having problems with. It may be that your problem is the result of a recently fixed bug.
- Spend some time creating a reproducible example. This mean a simplified version of your problem that another person can copy and

paste from their broswer into their R console to reproduce the error that you see. This can be quite an art-form, but the basic ingredients for a good reproducible exaple are: the packages you are using, a small data set, code, and a description of your R environment. R comes with many example data sets built-in, if you run `data()` you can see a list, and these are very convienent to use for reproducible examples. If you want to include your own data, you can use `dput()` to generate R code that another person can copy and paste to recreate your dataset. Your code should be easy to read, with spaces between operators (+, -, *, etc.) and after commas, and conformant to a style guide, such as Hadley Wickham's guide in Advanced R (`http://adv-r.had.co.nz/Style.html`). Your code should includes lines of commentary, which begin with #, to help others understand your problem. The usual way to communicate your R environment (ie. your operating system type and version), is to include the output from `sessionInfo()` into your question. Don't worry if some of these terms are unfamiliar at the moment, the main thing to know here is that help is available, and that the quality of help you receive is proportional to the effort you spend seeking it.

## 1.6 Prerequisites

To run the code in this book, you will need to have R installed on your computer, as well as the RStudio IDE, an application that makes it easier to use R. Both R and the RStudio IDE are free and easy to install.

### R

To install R, visit cran.r-project.org (`http://cran.r-project.org`). Then click the link that matches your operating system. What you do next will depend on your operating system.

- Mac users should click the most current release. This will be the `.pkg` file at the top of the page. Once the file is downloaded, double click it to open an R installer. Follow the directions in the installer to install R.

- Windows users should click "base" and then download the most current version of R, which will be linked at the top of the page.
- Linux users will need to select their distribution and then follow the distribution specific instructions to install R. cran.r-project.org (`https://cran.r-project.org/bin/linux/`) includes these instructions along side of the files to download.

## RStudio

Once you have R installed, it is time to download RStudio. To download RStudio, visit www.rstudio.com/download (`http://www.rstudio.com/download`).

Choose the installer for your system. Then click the link to download the application. Once you have the application, installation is easy. Once RStudio is installed, open it as you would open any other application.

## R Packages

Some of the most useful parts of R come in *packages*, collections of functions and code that you can download in addition to base R. We will use several packages in this book. These include the `dplyr`, `ggplot2`, `knitr`, `readr`, `rmarkdown`, `stringr`, and `tidyr` packages, among others.

There are two general ways to install packages for R. Both require you to have an internet connection, to start an R session (by opening the RStudio IDE), and to run a command at the command line.

The most common way to install R packages is to download them from the package repository at cran.r-project.org (`http://cran.r-project.org`). To do this run the command, `install.packages()`. Give `install.packages()` the name or names of the packages you wish to install as a character vector. R will download the packages from cran.r-project.org (`http://cran.r-project.org`) and install them in your system library.

You can use this method to download all but one of the packages listed above. To do so, open R and run the command

```
install.packages(c("devtools", "dplyr", "ggplot2",
                   "knitr", "readr", "rmarkdown",
                   "scales", "stringr", "tidyr"))
```

Some R packages are not stored on cran.r-project.org (`http://cran.r-project.org`), but are hosted in online repositories maintained by the package's developer. The most common place to host these packages is www.github.com (`http://www.github.com`).

For example, `aswr` is a collection of functions and data sets that I have assembled for this book and saved online as a github repository (github.com/benmarwick/aswr (`http://github.com/benmarwick/aswr`)).

You can install packages stored on github with the `install_github()` function in the `devtools` package. (You can install the `devtools` package itself from cran.r-project.org (`http://cran.r-project.org`) with `install.packages()`). To use the function, pass it a characterstring with the form "'/".

To install `aswr`, run the command

```
devtools::install_github("benmarwick/aswr")
```

```
library()
```

When R installs a package, it downloads the package to your system library. This does not automatically load the contents of the package into your current or future R sessions. To use the functions and data sets that come in an R package saved in your system library, you must load the package into your current R session with `library()`.

For example, to use the functions in the `tidyr` package, you would need to first run

```
library("tidyr")
```

You will need to rerun this command each time you open a new R session in which you wish to use the `tidyr` package.