Getting your data into R

How to load and export data

Dr Simon R. White

MRC Biostatistics Unit, Cambridge

2012

Part I

Case study: Reading data into R and getting output from R

Data formats

Not just applicable to R

Some pitfalls

- Many programs store information in special files which can only be opened by that program – this can cause problems
- Following a few simple guidelines can save a lot of time; proper data entry protocols make analysis easier
- Zero, non-response and missing are **different**

Un-learn these spreadsheet ideas

- Mixing raw data and functions applied to data
- Mixing data and how it is displayed (ever inserted an empty column to make a sheet look pretty?)
- Combining raw data and graphics (in a workbook)

Reading SAS, Minitab, SPSS, Stata

The ‘foreign’ package allows you to read data files from other statistical programmes. First you must load the package using:

```
> library(foreign)
```

`read.spss()`	Read an SPSS Data File
`read.dta()`	Read Stata Binary Files
`read.mtp()`	Read a Minitab Portable Worksheet
`read.xport()`	Read a SAS XPORT Format Library

What about Microsoft Excel?

You cannot (reliably) read Excel files into R directly
The simplest method is to save each sheet as a Comma Separated Value (CSV) file. Beware the defaults

- Excel saves the formatted version of values (which is very unhelpful): if you have set a column to 2 decimal places, only 2 decimal places will be saved in the csv file
- Functions saved as text, rather than their calculated value

Note

Excel, and spreadsheets in general, are not the place for statistical analysis or even data storage: they combine data, function, graphics, all together into workbooks, when the separation of these components is to be preferred
`www.burns-stat.com/pages/Tutor/spreadsheet_addiction.html`
(a good read on spreadsheets)

Reading csv files

`read.csv("filename.csv")`

Reads the contents of a file into a data frame

There are many options to this function, but of particular note

`header`	is the first line the column names (TRUE/FALSE)
`na.strings`	specifies strings to be interpreted as `NA`

`read.table("filename")`

General function for reading data from a file. Note, `read.csv()` calls the `read.table()` function with predefined parameters

Remember to assign the output of these functions to an object

Reading our first data into R (A)

Exporting from a spreadsheet

You should have the following directories on your machine:

```
C:/RCourse/
    data-original/
    data-exported/
    data/
```

In the `data-original` directory are several spreadsheet files. Open the `Centre_Information.xls` file and export it as a comma separated value file into the `data-exported` directory

Note: we will be using LibreOffice, an open source alternative to Microsoft Office

Note for Windows users: R uses '/' instead of '\' in file paths

Reading our first data into R (B)

```
> Data.Dir <- "C:/RCourse/data"

> Centre.Info <- read.csv(
    file.path(Data.Dir,"Centre_Information.csv") )

> str(Centre.Info)
```

What has R done? Are the assumed defaults sensible?

```
> Centre.Info.2 <- read.csv(
    file.path(Data.Dir,"Centre_Information.csv"),
    colClasses=c(NA,NA,NA,NA,"character") )
```

Sharing R objects

R can export data into a format for sharing with other programs

`write.table()`

General function for writing data to a file. See also `write.csv()`

If sharing with another R user, one can save R objects directly

`save( obj, file="obj.RData" )`

Saves `obj` in a file that can be loaded into another R session

`save.session( file="session.RData" )`

Save an entire R session – all objects that are returned by `ls()` Can `load()` file to resume working session

`load( file="obj.RData" )`

Load objects from file

Getting data out of R

```
> write.csv( Centre.Info )

> write.csv( Centre.Info,
    file= file.path( Data.Dir, "Out_1.csv" ) )
```

What has R done? Are the assumed defaults sensible?

```
> write.csv( Centre.Info,
    file= file.path( Data.Dir, "Out_1.csv" ),
    row.names=FALSE )
```

Getting output into a paper

```
> Tab.1 <- aggregate( Coverage ~ Region,
    Centre.Information, sum )

> write.csv( Tab.1 )
```

Importing a table into a paper

- Copy and paste content from R window into document
- Under ‘Table’ menu, select convert `Text to table`
- Adjust table within document

Could we make life easier?

```
> write.csv( Tab.1, quote=FALSE, row.names=FALSE )
```

Extra tricks

When a comma just will not work. . .

```
> write.table( Tab.1,
    quote=FALSE,
    row.names=FALSE, sep="\t" )
```

Change the order of rows and/or columns

```
> write.table( Tab.1[order(Tab.1$Coverage),],
    quote=FALSE,
    row.names=FALSE, sep="\t" )

> write.table( Tab.1[,c(2,1)],
    quote=FALSE,
    row.names=FALSE, sep="|" )
```