Reading data

Martina Sattlecker

Where's the spreadsheet?

- statistical packages normally have a spreadsheet
- R has a minimal-but-usable spreadsheet
- more emphasis on data generated/curated externally
- very powerful data import tools

review of data structures –vectors

- everything is based on vectors
- a vector is a series of values of the same type
- we've seen 'character', 'numeric' and 'logical' vectors
- a 'factor' is a special type of vector for categorical data

review of data structures –attributes

- attributes are information specific to an object
- can be anything including your own notes

```
attr(mydata,'comment')<-'deeply dubious data'
```

- attributes not called 'comment' are shown when you print

especially important attributes

dimensions, names :

```
a<-1:100
dim(a)<-c(10,10)
a[50]
a[10,5]
colnames(a)<-letters[1:10]
a[,’j’]
```

review of data structures - lists, data frames

- a table of data that's all one vector must be all one type
- your typical excel file isn't like that
- a list is a collection of any assorted objects
- a dataframe is a list of assorted vectors all the same length
  - the vectors (columns) must also have unique names

```
b<-data.frame(a)
b$j
b[,10]
```

- dataframes can be treated like matrices for many purposes
- there are a few stumbling blocks
  - compare a[1] and b[1]

the read.table() function

- the most common way to input tabular data, eg Excel
- many other possibilities exist
- save data as tab-or comma separated text
- one sheet per file
- use a version without too much extraneous junk
- simple row and column labels, one line header

Pathway and working directory

- Find your current working directory; set a new one

  getwd()
  setwd("")

- List files or directories (folders)

  list.files()
  list.dirs()

- For other functions to manipulate files

  ?files

use of read.table()

- Read in your data

```
my_df <- read.table("my_data.txt", header=TRUE)
```

- Note: read.table() returns a dataframe (cases correspond to rows; variables to columns)

- Check the data looks as you expect it to

```
head(my_df)
names(my_df)
str(my_df)
summary(my_df)
```

use of read.table(), continued

- the defaults don’t only work in the simplest cases
- safer to be explicit for general use
- It might take a series of steps to get the data in appropriately
- If possible, looking at the data in a text editor helps

use of read.table(), continued

```
cfdata<-
read.table("http://rcourse.iop.kcl.ac.uk/S3/regresssionof.txt")

cfdata<-
read.table("http://rcourse.iop.kcl.ac.uk/S3/regresssionof.txt”,
header=TRUE)

cfdata<-
read.table("http://rcourse.iop.kcl.ac.uk/S3/regresssionof.txt",
header=TRUE, sep="\t")

cfdata<-
read.table("http://rcourse.iop.kcl.ac.uk/S3/regresssionof.txt",
header=TRUE, sep="\t", as.is=TRUE)
```

Special forms of read.table()

- Read in the height and weight data in df_body.csv (a comma-separated values file)

```
body_data <- read.csv("df_body.csv")
```

- Give the columns the names "height" and "weight" on reading in the data:
  - One way is to re-read the data:

```
body_data <-     read.csv("df_body.csv”,col.names=c("id","height", "weight"))
```

  - Or, just rename the columns

```
names(body_data) <- c("id","height","weight")
```

keeping your data under control

- both workspace and command history can be saved
- you can restart where you left off
- this gives a false sense of security
- a much better strategy is to keep a log
- paste commands that have done what you want into an editor
- (optional) name it something.R
- put comments starting with #
- such a log can be used as a script to generate your analysis again

the foreign package

- add- on package, need to install and load

```
install.packages('foreign')
library(foreign)
```

- has methods for importing data files from most stats packages
- SAS,SPSS,STATA, not Statistica
- also some database like files
- .DBF (xbase, foxpro, clipper, etc)
- corresponding export methods as well

Read SPSS and STATA files

- Read in an SPSS .sav file

```
us_crime<-read.spss("crime.sav",to.data.frame=T)
```

- Read in a STATA .dta file

```
census<-read.dta("census.dta",convert.factors=FALSE)
```

data may need cleanup

- missing value codes often differ
- people often use -1 or 999
- statistics software varies as well

Microsoft Excel and Access

- RODBC R Open Database Connectivity
- easy to read in whole Excel or Access tables

```
install.packages("RODBC")
library(RODBC)
help(package=RODBC)
```

Reading Excel files

- Read in a Excel .xls file

```
myexcelchannel <-odbcConnectExcel("genotest.xls")
```

- To find out what tables are accessible from a connection

```
sqlTables(myexcelchannel)

sheet1 <- sqlFetch(myexcelchannel, "ABC")
sheet2 <- sqlFetch(myexcelchannel, "DEF")
sheet3 <- cbind(sheet1, sheet2)
```

Write and Save

- Save workspace

```
save.image(file="save_exercise.Rdata")
```

- Either open up a new R session and use load(file="save_exercise.Rdata")

- or, simply double click on the saved file

Write and Save

NB. with write(), transpose the matrix to print columns to file the same as they look in the R console

```
write(t(my_matrix), file="write_exercise_1.txt")
```

```
write.table(my_dataframe, file="write_exercise_2.txt", row.names=F)
```

```
write.csv(my_dataframe, file="write_exercise_3.txt", row.names=F, col.names=T)
```