Introduction to R

September 9, 2013

Jason Bryer (jason@bryer.org)
epsy530.bryer.org

What is R?

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues...

R provides a wide variety of statistical (linear and non linear modeling, classical statistical tests, time-series analysis, classifcation, clustering, ...) and graphical techniques, and is highly extensible. The S language is often the vehicle of choice for research in statistical methodology, and R provides an Open Source route to participation in that activity. (R-project.org)

Pros

- FREE! R is available as Free Software under the terms of the Free Software Foundation's GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS.
- Available for multiple platforms (i.e. Windows, Mac, Linux).
- Easily extensible with (currently) over 2,000 packages listed on CRAN. Scriptable.
- Publication grade graphics.
- Multiple ways of doing the same thing.
- Quickly becoming the de facto standard among statistician.

Cons

- Has a steeper learning curve.
- Multiple ways of doing the same thing.
- Can have difficulty with very large datasets.

R's Roots... S

- S is a language that was developed by John Chambers and others at Bell Labs.
- S was initiated in 1976 as an internal statistical analysis environment - originally implemented as Fortran libraries.
- Early versions of the language did not contain functions for statistical modeling.
- In 1988 the system was rewritten in C and began to resemble the system that we have today (this was Version 3 of the language). The book Statistical Models in S by Chambers and Hastie (the blue book) documents the statistical analysis functionality.
- Version 4 of the S language was released in 1998 and is the version we use today. The book Programming with Data by John Chambers (the green book) documents this version of the language.

History of S

- In 1993 Bell Labs gave StatSci (now Insightful Corp.) an exclusive license to develop and sell the S language.
- In 2004 Insightful purchased the S language from Lucent for $2 million and is the current owner.
- In 2006, Alcatel purchased Lucent Technologies and is now called Alcatel-Lucent.
- Insightful sells its implementation of the S language under the product name S-PLUS and has built a number of fancy features (GUIs, mostly) on top of it-hence the ”PLUS”.
- In 2008 Insightful is acquired by TIBCO for $25 million; future of S-PLUS is uncertain.
- The S language itself has not changed dramatically since 1998.
- In 1998, S won the Association for Computing Machinery’s Software System Award.

> “[W]e wanted users to be able to begin in an interactive environment, where they did not consciously think of themselves as programming. Then as their needs became clearer and their sophistication increased, they should be able to slide gradually into programming, when the language and system aspects would become more important.”

John Chambers, Stages in the Evolution of S

History of R

- 1991: Created in New Zealand by Ross Ihaka and Robert Gentleman. Their experience developing R is documented in a 1996 JCGS paper.
- 1993: First announcement of R to the public.
- 1995: Martin M?achler convinces Ross and Robert to use the GNU General Public License to make R free software.
- 1996: A public mailing list is created (R-help and R-devel)
- 1997: The R Core Group is formed (containing some people associated with S-PLUS). The core group controls the source code for R.
- 2000: R version 1.0.0 is released.
- 2012: R version 2.15.2 is released on October 31, 2012. There are now over 4,000 packages listed on CRAN.

Installing R

The latest version of R can be obtained from http://cran.r-project.org. For Windows the following should also be installed:

- RTools http://cran.r-project.org/bin/windows/Rtools/

For Mac the following should also be installed which are available from http://cran.r-project.org/bin/macosx/tools

- gfortran-4.2.3
- tcl/tk 8.5.5

Download RStudio http://www.rstudio.com/

R Version

The version of R I am running is:

```
R.version$version.string
```

```
[1] "R version 3.0.0 (2013-04-03)"
```

R is a big calculator

```
> 2 + 2
```

```
[1] 4
```

```
> 1 + sin(9)
```

```
[1] 1.412
```

```
> 23.76 * log(8)/(23 + atan(9))
```

```
[1] 2.02
```

R Packages

R Packages are what make R so popular. A package is a collection of R functions, data, and documentation to peform some task. There are currenlty over 4,000 packages!

You can install packages in RStudio from the `Packages` tab. Or you can use the `install.packages` function.

```
install.packages(c("ggplot2", "foreign", "psych"))
```

You can update installed packages using the `update.packages` function.

```
update.packages(ask = FALSE)
```

The `library` function will return a list of installed packages.

```
library()
```

Loading R Packages

Installed packages are not automatically loaded. To load a package, use the `require` function.

```
require(ggplot2)
require(foreign)
```

The `search` function will return a list of loaded packages.

```
search()
```

```
 [1] ".GlobalEnv"        "package:foreign"   "package:reshape"
 [4] "package:plyr"      "package:gdata"     "package:slidifyLibraries"
 [7] "package:slidify"   "tools:rstudio"     "package:graphics"
[10] "package:grDevices" "package:utils"     "package:datasets"
[13] "package:ggplot2"   "package:stats"     "package:methods"
[16] "Autoloads"         "package:base"
```

What's in a package?

Functions

The `ls(package='packagename')` function will return a list of functions that are available in a package. See the results of `ls()` for what the `package` parameter should be.

```
ls("package:foreign")
```

```
 [1] "data.restore"  "lookup.xport"  "read.arff"     "read.dbf"      "read.dta"
 [6] "read.epiinfo"  "read.mtp"      "read.octave"   "read.S"        "read.spss"
[11] "read.ssd"      "read.systat"   "read.xport"    "write.arff"    "write.dbf"
[16] "write.dta"     "write.foreign"
```

What's in a package?

Data

The `data()` function will return all data available in all installed packages. Or, specify the `package` parameter to limit the results to one package.

```
data(package = "ggplot2")
```

What's in a package?

Vignettes

The `vignette()` function will return all vignettes (package documentation, usually a PDF) for all installed packages. Or, specify the `package` parameter to limit teh results to one package.

```
vignette(package = "psych")
```

Getting Help

- R provides extensive documentation and help. The help.start() function will launch a webpage with links to:
  - The R manuals
  - The R FAQ
  - StatMethods
  - R-Blogger
  - and many other useful sites
- The help.search() function will search the help file for a particular word or phrase. For example:

```
help.search("cross tabs")
```

- To get documentation on a specific function, the `help()` function, or simply `?functionName` will open the documentation page in the web browser.
- Lastly, to search the R mailing lists, use the `RSiteSearch()` function.

Reading Excel Files

```
require(gdata)
mathAnxiety <- read.xls("../Data/MathAnxiety.xls", sheet = 1)
names(mathAnxiety)
```

```
 [1] "Gender" "q1"     "q2"     "q3"     "q4"     "q5"     "q6"     "q7"     "q8"     "q9"
[11] "q10"    "q11"    "q12"    "q13"    "q14"
```

```
nrow(mathAnxiety)
```

```
[1] 20
```

```
ncol(mathAnxiety)
```

```
[1] 15
```

Reading SPSS Files

The foreign package provides a function to read SPSS files.

```
shy = read.spss("../Data/Exercise2.sav", use.value.labels = FALSE, to.data.frame = TRUE)
```

```
re-encoding from latin1
```

```
names(shy)
```

```
[1] "rowtype_"  "varname_"  "age"       "technical" "social"    "frequency"
```

This data file contains six columns: social anxiety (soax), restricted emotionality (reemo), restricted affectionate behavior (reaff), intimate self-disclosure (isd), a single degree-of-freedom continuous measure of shyness (shy), and a three-group experimental structural variable (group).

Reading CSV Files

R can read virtually any type of plain text file with the read.table function. For convenience, the read.csv will provide a quick way of reading comma-separated values (CSV) files. For example:

```
titanic <- read.csv("../Data/titanic.csv")
```

But `read.csv` is really just a wrapper to the `read.table` function that can read virtually any plain text format.

```
titanic <- read.table("../Data/titanic.csv", header = TRUE, sep = ",", quote = "\"")
```

Data Frames

```
class(titanic)
```

```
[1] "data.frame"
```

```
head(titanic, n = 3)
```

```
  pclass survived                          name    sex   age sibsp parch ticket  fare   cabin
1      1        1  Allen, Miss. Elisabeth Walton female 29.00     0     0  24160 211.3      B5
2      1        1 Allison, Master. Hudson Trevor   male  0.92     1     2 113781 151.6 C22 C26
3      1        0   Allison, Miss. Helen Loraine female  2.00     1     2 113781 151.6 C22 C26
  embarked boat body                        home.dest
1        S    2   NA                    St Louis, MO
2        S   11   NA Montreal, PQ / Chesterville, ON
3        S        NA Montreal, PQ / Chesterville, ON
```

Structure

The `str` (short for structure) is arguably the most useful function in R. It will give you information about the structure of an object (variable) in R relevant to it's type.

```
str(titanic)
```

```
'data.frame':   1309 obs. of  14 variables:
 $ pclass   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ survived : int  1 1 0 0 0 1 1 0 1 0 ...
 $ name     : Factor w/ 1307 levels "Abbing, Mr. Anthony",..: 22 24 25 26 27 31 46 47 51 55 ...
 $ sex      : Factor w/ 2 levels "female","male": 1 2 1 2 1 2 1 2 1 2 ...
 $ age      : num  29 0.92 2 30 25 48 63 39 53 71 ...
 $ sibsp    : int  0 1 1 1 1 0 1 0 2 0 ...
 $ parch    : int  0 2 2 2 2 0 0 0 0 0 ...
 $ ticket   : Factor w/ 929 levels "110152","110413",..: 188 50 50 50 50 125 93 16 77 826 ...
 $ fare     : num  211 152 152 152 152 ...
 $ cabin    : Factor w/ 187 levels "","A10","A11",..: 45 81 81 81 81 151 147 17 63 1 ...
 $ embarked : Factor w/ 4 levels "","C","Q","S": 4 4 4 4 4 4 4 4 4 2 ...
 $ boat     : Factor w/ 28 levels "","1","10","11",..: 13 4 1 1 1 14 3 1 28 1 ...
 $ body     : int  NA NA NA 135 NA NA NA NA NA 22 ...
 $ home.dest: Factor w/ 370 levels "","?Havana, Cuba",..: 310 232 232 232 232 238 163 25 23 230 ...
```

NA versus NULL

R is just as much a programming language as it is a statistical software package. As such it represents null differently for programming (using NULL) than for data (using NA).

- NULL represents the null object in R: it is a reserved word. NULL is often returned by expressions and functions whose values are undefined.
- NA is a logical constant of length 1 which contains a missing value indicator. NA can be freely coerced to any other vector type except raw. There are also constants NA integer , NA real , NA complex and NA character of the other atomic vector types which support missing values: all of these are reserved words in the R language.

For more details, see http://opendatagroup.com/2010/04/25/r-na-v-null/