Introduction to R: Part I

Jeffrey C. Miecznikowski

March 26, 2015

R impact

The New York Times

Business Computing

WORLD | U.S. | N.Y. / REGION | BUSINESS | TECHNOLOGY | SCIENCE | HEALTH | SPORTS | OPINION

Data Analysts Captivated by R's Power

Left, Stuart Isett for The New York Times; right, Kieran Scott for The New York Times

R first appeared in 1996, when the statistics professors Robert Gentleman, left, and Ross Ihaka released the code as a free software package.

By ASHLEE VANCE
Published: January 6, 2009

To some people R is just the 18th letter of the alphabet. To others, it's the rating on racy movies, a measure of an attic's insulation or what pirates in movies say.

Related

Bits: R You Ready for R?

The R Project for Statistical Computing

R is also the name of a popular programming language used by a growing number of data analysts inside corporations and academia. It is becoming their lingua franca partly because data mining has entered a golden age, whether being used to set ad prices, find new drugs more quickly or fine-tune financial models. Companies as diverse as Google, Pfizer, Merck, Bank of America, the InterContinental Hotels Group and Shell use it.

- R is the 13th most popular language by IEEE Spectrum (2014)
- Google uses R for ROI calculations
- Ford uses R to improve vehicle design
- Twitter uses R to monitor user experience
- The US National Weather Service uses R to predict severe flooding
- R is used by New York Times to create infographics and interactive journalism applications
- R user estimates in the several millions

Source: New York Times, January 9th, 2009

R Background

- R was started in 1993 and is a GNU (free) implementation of S
- S is a statistical programming language developed at Bell Laboratories primarily by John Chambers
- R was created by Rob Gentleman and Ross Ihaka
- Versions are available for Windows, Mac, Linux, and Unix
- R uses a command line interface
- See www.r-project.org

R …

- Allows for effective data handling and storage
- Offers operators for calculations on arrays and matrices
- Offers a large collection of tools for data analysis
- Provides graphical facilities for data analysis
- Includes conditionals, loops, and user defined recursive functions.

The R environment

- R is often called the “R environment” since it a fully planned and coherent system rather than an incremental accumulations of very specific and inflexible tools as is the case with other data analysis software

R Features

- Libraries of functions offer cutting edge statistics
- A core set of packages are included with basic R installation with over 6000 more available at the Comprehensive R Archive Network (CRAN)
- See cran.r-project.org
- Extending R via libraries is simple and straightforward

R Flavors

- R can be installed from www.r-project.org
- R in UNIX-like machines may be slightly different than in Windows or OS X.
- R has newer user interfaces e.g. Rstudio

R Basics

- Starting and quitting an R session
- Basic commands
- Create objects, e.g. vectors and matrices
- Produce plots
- Read in data
- Write functions
- Load a library
- Seek help
- Save the results

Starting and Quitting R

When you use the R program it issues a prompt when it expects input commands

The default prompt is '>'

In using R under UNIX the suggested procedure for the first occasion is as follows: Start the R program with the command

```
> R
```

At this point R commands may be issued (see later).

In using R in Windows merely double click the R program or RStudio.

To quit R the command is

```
> q()
```

Basic Commands

To have R execute a command, simply type the command and press return, e.g. type $2 + 3$ and then press return

```
> 2+3
[1] 5
```

Arithmetic and order of operations work exactly in the ordinary way. Multiplication is `*`, division is `/` and exponentiation is `^`.

R is case sensitive, and the format for calling a function is `functionname`(arg1,arg2,...).

Basic Commands

For instance, the square root function and absolute value function are

```
> sqrt(9)
[1] 3
> abs(-5)
[1] 5
```

The "[1]" which keeps showing up means that the first item on that line is the first item in the vector of output.

Create a Vector

The simplest object in R is a scalar. Scalars are single numbers of type "numeric" or type "character". Characters are in "": `"x"` means the letter `x` while `x` means the object x.

Similarly, 1 means the number one, and `"1"` means the symbol used for the number one.

A vector is a set of scalars arranged in a one dimensional array, vectors can contain only one type of scalar, either numbers or characters but not both. If R sees both, it will change the numeric arguments to characters by enclosing them in quotes.

```
> blah = c(1,2,3,"d")
> blah
[1] "1" "2" "3" "d"
```

In the example above, R converted the numbers 1,2, and 3 to their respective symbols so that all 4 elements in `blah` would be characters.

Selecting from vectors

Subsets of vectors can be selected by using square brackets.

```
> my.vec=c(12,4,6,1,26)
> my.vec[1]
[1] 12
> my.vec[2:4]
[1] 4 6 1
> my.vec[c(1,3,5)]
[1] 12 6 26
```

To select everything except some particular set of elements, use a minus sign.

```
> my.vec[-2]
[1] 12 6 1 26
```

Vectors can also be searched using logic statements.

```
> my.vec[my.vec>6]
[1] 12 26
```

Logic statements

In the last example we examined,

```
> my.vec[my.vec>6]
[1] 12 26
```

Think of this as "we want to return my.vec but only where my.vec>6".

This is another way to subset data. Rather than give the observations we want, we put a 1 next to an observation we want and a 0 next to an observation we want to discard.

Note `my.vec>6` is a vector of 0's and 1's and has length equal to length of `my.vec`.

This vector (`my.vec>6`) of 0's and 1's determines what to keep in `my.vec`.

Create a matrix

A matrix is like a vector, but in two dimensions. To enter a matrix into R, enter a vector of data and the dimensions of the matrix. E.g.

```
> matrix(c(1:9),nrow=3,ncol=3,byrow=T)
     [,1] [,2] [,3]
[1,]    1    2    3
[2,]    4    5    6
[3,]    7    8    9
```

The "byrow" argument tells R to read the data in by row. The default is to read it in by column.

```
> matrix(c(1:9),nrow=3,ncol=3)
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
```

Vectors can be converted into matrices, by using the `as.matrix` function. This function changes a vector of length n into an n by 1 matrix. The `as.vector` function can be used to change a matrix into a vector.

Selecting from a matrix

To check the dimensions of a matrix, use the function `dim`.

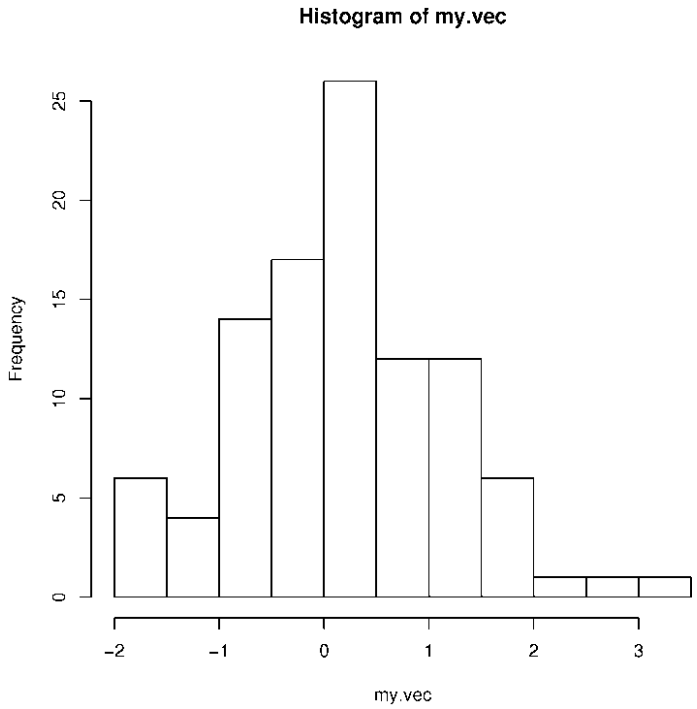
```
> mymat=matrix(c(1:9),nrow=3,ncol=3)
> dim(mymat)
[1] 3 3
> mymat[1:2,2:3]
     [,1] [,2]
[1,]    4    7
[2,]    5    8
```

Selecting from a matrix is similar to selecting from a vector, the first indice refers to the row, the second indice refers to the colums. So we've selected the first two rows of `mymat` and the 2nd and 3rd columns of `mymat`.

Produce a plot

Plot commands entered at the regular R prompt will cause pictures to appear in the R window.

```
> my.vec=rnorm(100,0,1)
> hist(my.vec)
```



Save the results

When quitting a session it is sometimes worth saving your output and possibly a history of the session

In Unix, when you quit your session, it will ask you if you want to save your workspace, by typing "y", R will automatically create a .Rdata subdirectory within your working directory that contains your variables. Also it will create a .Rhistory subdirectory containing a list of the commands executed.

In Windows, your data variables can be saved by going to tab "File>Save Workspace..." and the history can be saved by going to the tab "File>Save History ..."

Read Data into R

There are several R commands that should handle reading in data or troubleshooting the problems on reading data.

1. `read.table`
2. `scan`
3. `count.fields`
4. `readLines`
5. `read.xls` in `gdata` package

Note issuing `?command` will provide help on command.

Read Data into R

Things to consider with a dataset.

1. Weird Characters such as $, #, &, *, !, ~
2. Number of fields
3. Type of value in each field
4. Blanks in the data set
5. Delimiter between each field
6. Header
7. NA's
8. End of each line

With errors on reading in the data, often the problem is with one of the above items being inconsistent.

Loops in R

Most commonly you will use `for`, `while` loops. The syntax for a `for` loop is

```
for(variable in sequence){
statements
}
```

The syntax for a `while` loop is

```
while(condition){
statements
}
```

Note the `condition` should be a logical statement.

Example Loops

Here's a simple `while` loop:

```
z = 0
while(z < 5){
          z = z+2
          print(z)
} # end while loop
[1] 2
[1] 4
[1] 6
```

Here's a simple `for` loop:

```
  vector=c()
for(i in 1:10){
   vector[i] = i+1
   } # end for loop
```

Aside on vector declarations

Note in our last example we started with

```
vector=c()
```

This command informs R that `vector` is an empty vector, that will hopefully be populated later on.

Absence of this declaration you will get an error that `vector` does not exist.

Defining functions in R

Often you will want to write your own functions in R. Here's the syntax,

```
myfunction = function(arg1, arg2, ...){
statements
return(object)
} # end myfunction
```

Here's an example,

```
myfunction = function(x,y){
out = x+y
return(out)
}
myfunction(3,4)
[1] 7
```

Load a library

All R functions and datasets are stored in packages.

Only when a package is loaded are its contents available.

This is done for memory purposes and to aid package developers in protecting name clashes

To see what packages are installed at your site, issue the command

```
library()
```

To load a particular package, say `boot`, use a command like

```
library(boot)
```

Install a library

If the library you would like to load is not listed when issuing `library()`, you must install it using, say,

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

This command will ask you to choose a mirror (I don't think it matters what mirror you choose) and then once the package is installed it can be loaded as shown on the previous slide.

Help

Several ways of getting help, depending on the level of help required.

1. Download the R Reference guide from `http://cran.r-project.org/doc/manuals/`
2. For any specific command, type `?command`, e.g. `?plot`
3. Search the internet from sites such as google
4. Type `help(command)` e.g. `help(plot)`

On your own

Try the following tasks

1. Define

```
x = c(4,2,6)
y = c(1,0,-1)
```

2. Decide the following results and use R to check

   (a) `length(x)`

   (b) `sum(x)`

   (c) `sum(x^2)`

   (d) `x+y`

   (e) `x*y`

On your own

3. Write a function that squares the input and adds 20.
4. Write a function with a `for` loop with a vector input that returns a vector of -1's and 1's, -1 where the corresponding input was negative and 1 where the corresponding input was positive, 0 remains unchanged.
5. Produce a histogram of observations obtained from `rnorm(50)` using the `hist` command.
6. Load the library `MASS` and produce the histogram using the `truehist` function