Stats with Sparrows - 1

Julia Schroeder

September 2016

1

Revisiting what R can do for you

In this practical, you will practice using the R command line interface to do some basic stats. First, we revisit what you learned last week to make sure you can use these skills. In these practicals, we will be working from your home directories ("H:/") so, using Windows Explorer, open your home directory and create a new directory called 'SparrowStats'. You will be using this to store our files of code and data from each day.

A lot of this is revisiting last week. Start up R or RStudio, or whatever GUI you wish. Remember, you can download R from the Comprehensive R Archive Network (http://cran.r-project.org/). Make sure to create a new R script for each day, and do not simple copy-paste from this handout but rather try to think about what the code does. We wille xpect you to write such code by yourself every day.

Some good house-keeping: Checking and re-setting your working directory:

```
getwd()
setwd("H:/StatsWithSparrows")
getwd()
```

Remember to always comment your code:

```
getwd()                    # check which working directory we're in
setwd("H:/StatsWithSparrows")   # re-set workign directory to
H:/StatsWithSparrows
getwd()                    # check that what we did worked
```

Remember you can use basic calculator commands in R:

```
2*2+1
```

```
## [1] 5
```

```
2*(2+1)
```

```
## [1] 6
```

```
12/2^3
```

```
## [1] 1.5
```

```
(12/2)^3
```

```
## [1] 216
```

Remember you can assign values to variables:

```
x <- 5
x
```

```
## [1] 5
```

```
y <- 2
y
```

```
## [1] 2
```

```
x2 <- x^2
x2
```

```
## [1] 25
```

```
x
```

```
## [1] 5
```

```
a<-x2+x
a
```

```
## [1] 30
```

```
y2 <- y^2
z2 <- x2 + y2
z <- sqrt(z2)
print(z)
```

```
## [1] 5.385165
```

There are logical tests:

```
3>2
```

```
## [1] TRUE
```

```
3 >= 3
```

```
## [1] TRUE
```

```
4<2
```

```
## [1] FALSE
```

You can create vectors of different formats:

```
myNumericVector <- c(1.3,2.5,1.9,3.4,5.6,1.4,3.1,2.9)
myCharacterVector <- c("low","low","low","low","high","high","high","high")
myLogicalVector <- c(TRUE,TRUE,FALSE,FALSE,TRUE,TRUE,FALSE,FALSE)
```

We will use the str() function to get the structure of any variable.

```
str(myNumericVector)

##  num [1:8] 1.3 2.5 1.9 3.4 5.6 1.4 3.1 2.9

str(myCharacterVector)

##  chr [1:8] "low" "low" "low" "low" "high" "high" "high" ...

str(myLogicalVector)

##  logi [1:8] TRUE TRUE FALSE FALSE TRUE TRUE ...
```

It is important for statistical reasons, to always be on top of what which variable is - numeric, vector, ect. If you add values of different categories to one variable, you will create a mixed variable and R will coerce (force) it into one of the most basic mode that covers all:

```
myMixedVector <-c(1, TRUE, FALSE, 3, "help", 1.2, TRUE, "notwhatIplanned")
str(myMixedVector)

##  chr [1:8] "1" "TRUE" "FALSE" "3" "help" "1.2" "TRUE" ...
```

We install packages, and load them. Can you find out what the difference is between the library and the require command?

```
install.packages("lme4")
library(lme4)
require(lme4)
```

When we are lost, we ask R for help:

```
help(getwd)

help(log)
```

There is some special notation in R.

```
sqrt(4); 4^0.5; log(0); log(1); log(10); log(Inf)

## [1] 2

## [1] 2

## [1] -Inf

## [1] 0

## [1] 2.302585
```

```
## [1] Inf
```

Inf means infinity. You can specify if it's negative or positive. In R, *e* is typed in using an R function exp and we should type in exp(1). For π, just type in pi.

```
exp(1)

## [1] 2.718282

pi

## [1] 3.141593
```

R shows up to 6 decimal places, the default is to show 7 digits. To what precision should you report decimal places in your reports?

Clear your workspace at the beginning of each session, and between different parts of a project.

```
rm(list=ls())
```

Entering data

We can enter data in many ways, and you will have learned last week how to do that. Here, we will mostly simply read tables with headers:

```
d<-read.table("SparrowSize.txt", header=TRUE)
str(d)

## 'data.frame':    1770 obs. of  10 variables:
##  $ BirdID     : int  4401 4401 4405 4405 4405 4409 4409 4409 4409 4409 ...
##  $ Cohort     : int  1991 1991 1994 1994 1994 1994 1994 1994 1994 1994 ...
##  $ CaptureDate: Factor w/ 414 levels "01-Aug-06","01-Dec-07",..: 272 18
254 41 88 303 174 18 159 164 ...
##  $ CaptureTime: Factor w/ 293 levels "04:00","04:30",..: NA NA NA NA NA NA
NA NA NA NA ...
##  $ Tarsus     : num  18.9 18.8 19.1 19 19.1 ...
##  $ Bill       : num  NA NA NA NA NA NA NA NA NA NA ...
##  $ Wing       : num  82 79 77 78 77 76 76 73 79 77 ...
##  $ Mass       : num  29.4 31.6 29.9 31.6 31 ...
##  $ Sex        : int  1 1 0 0 0 1 1 1 1 1 ...
##  $ Sex.1      : Factor w/ 2 levels "female","male": 2 2 1 1 1 2 2 2 2 2
...
```