Lab 1: R Markdown and Data Manipulation

Hao Wang

Jan.23rd 2017

```
#setwd("D:/Dropbox/2017 Spring/ 603/lab/Lab 1")
setwd('~/Dropbox/2017 Spring/ 603/lab/Lab 1')
getwd()
```

```
## [1] "/Users/haowang/Dropbox/2017 Spring/ 603/lab/Lab 1"
```

Google's R Style Guide:

https://google.github.io/styleguide/Rguide.xml

Rmarkdown

http://rmarkdown.rstudio.com/lesson-1.html

R Markdown provides an authoring framework for data science. You can use a single R Markdown file to both

- save and execute code
- generate high quality reports that can be shared with an audience
- Comparing with LaTeX and Sweave, Markdown's grammar is simpler and more firendly for html display.

R Markdown documents are fully reproducible and support dozens of static and dynamic output formats.

I recommend using rmardown for coding and taking notes of your analyses.

Enter your R codes in the blocks. Rmarkdown is also compatible for other languages: Python, SQL, Stan, C++, Java.

For instance

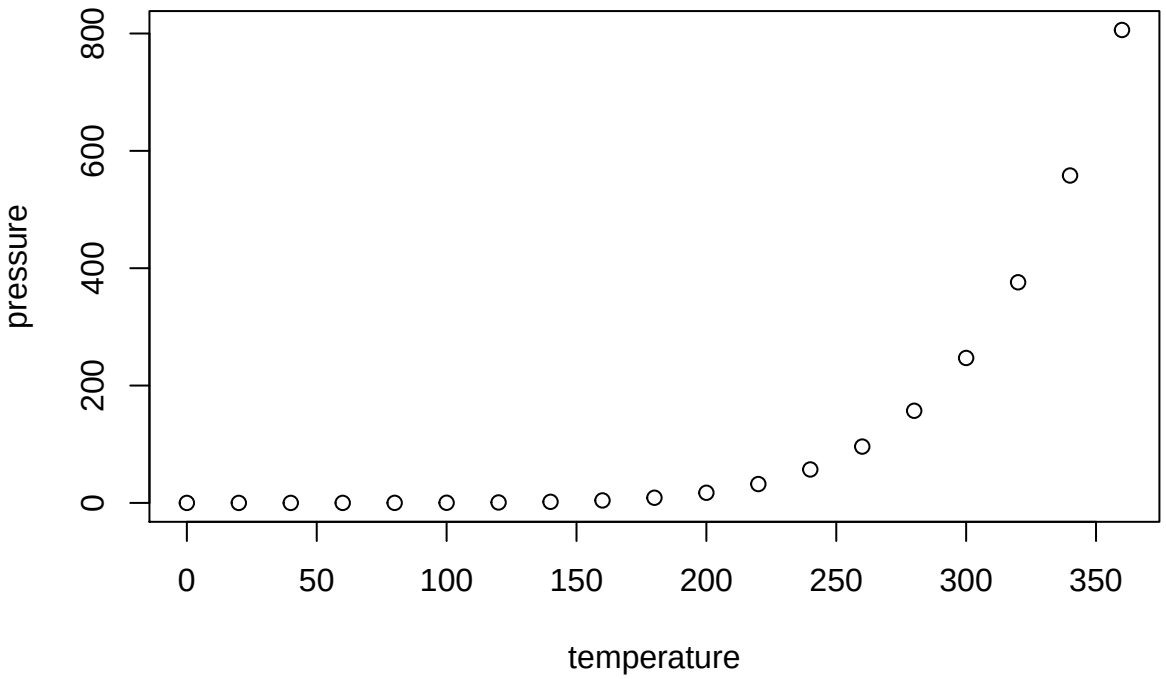
```
#Same R language used in blocks. Using # for marking.
#Echo means if you want to show the R codes in the final reports
#warning means if you want to turn the warning message on or off
#message means if you want to report the console result

library(car) #car is a basic package for applied regression
summary(cars) #cars is a default dataset included in the car package
```

```
##      speed           dist       
##  Min.   : 4.0   Min.   :  2.00  
##  1st Qu.:12.0   1st Qu.: 26.00  
##  Median :15.0   Median : 36.00  
##  Mean   :15.4   Mean   : 42.98  
##  3rd Qu.:19.0   3rd Qu.: 56.00  
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

- Lab Practice 1: create a .rmd file. Write down your name and set up your working directory.

A standard set-up for your codes.

- Date created
- Date lasted modified
- your working directory using setwd(")
- packages

For instance (You should replace with your own working directory)

```
#Hao Wang
#Date created: Jan. 12th / 2017
#Last modified: Jab. 15th
#setwd("D:/Dropbox/2017 Spring/ 603/lab/Lab 1")
#getwd()
options(digits=10)
library(knitr)
library(rmarkdown)
library(foreign)
#library(xlsx)
```

Read data in R.

R can handle differnt types of data (csv, txt, excel, stata, sas etc.)

1. Read csv (comma separated values) file: use read.csv function

```
#R can read files directly from the website
mydata.csv <- read.csv('http://www.cyclismo.org/tutorial/R/_static/simple.csv', header = TRUE, sep=",")
write.csv(mydata.csv, "mydata.csv")
#header option means taking the first row as variable names.
#sep option is determined by the way the data file is separated
```

2. Read txt

We create a txt data first and then import the txt file to R.

```
require(car)
write.table(cars,"cars.txt", sep="\t")
mydata.txt <-read.table("cars.txt")
```

3. Read dta(stata)

- For stata 12 and earlier (package 'foreign')
- For stata 13 and later (package'readstata13', this also works for earlier versions)

```
#install.packages('readstata13')
require(readstata13)
```

```
## Loading required package: readstata13
```

```
stata14 <-read.dta13('stata14.dta')
stata12 <-read.dta('stata12.dta')
```

4. Read excel

Similarly, we export an excel file and read it.

```
#install.packages("xlsx")
library(xlsx)
```

```
## Loading required package: rJava
```

```
## Loading required package: xlsxjars
```

```
library(car)
cars.xlsx <- cars
write.xlsx(cars, 'cars.xlsx', sheetName = "Sheet1", row.names =FALSE, col.names = TRUE)
```

- Lab Practice 2: find a data source of either format, load it into R.

Dealing with Missing Data

Missing data is very common in our research. Since the missing points are coded differently (for instance: 'NA', '99', '-999'), make sure to check the dataset before analysis.

Recoding Missing Data

```
y <- c(1,2,3,NA)
is.na(y) # returns a vector (FFFT)
```

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

```
#recode other values to missing, speed =10 to NA
mydata <- cars
mydata$speed[mydata$speed ==10] <- NA
mydata$speed
```

```
##  [1]  4  4  7  7  8  9 NA NA NA 11 11 12 12 12 12 13 13 13 13 14 14 14 14
## [24] 15 15 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24
## [47] 24 24 24 25
```

```
#Code na.omit() listwise delete all the missing observations. Be careful when using this command
#listwisde deletion is BAD!
#We can code NAs back to 10
mydata$speed[is.na(mydata$speed)] <- 10
mydata$speed
```

```
##  [1]  4  4  7  7  8  9 10 10 10 11 11 12 12 12 12 13 13 13 13 14 14 14 14
## [24] 15 15 15 16 16 17 17 17 18 18 18 18 19 19 19 20 20 20 20 20 22 23 24
## [47] 24 24 24 25
```

Recode data

Create new variables

```
mydata$speed2 <- (mydata$speed)^2

#This line create dummny variables based on speed
mydata$speed3 <- ifelse(mydata$speed > 10,
c("slow"), c("quick"))
mydata$speed3
```

```
##  [1] "quick" "quick" "quick" "quick" "quick" "quick" "quick" "quick"
##  [9] "quick" "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"
## [17] "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"
## [25] "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"
## [33] "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"
## [41] "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"  "slow"
## [49] "slow"  "slow"
```

```
# another example: create 3 categories
mydata$speed4[mydata$speed <10] <- "slow"
mydata$speed4[mydata$speed >= 10 & mydata$speed < 18] <- "middle"
mydata$speed4[mydata$speed >= 18] <- "quick"
mydata$speed4
```

```
##  [1] "slow"   "slow"   "slow"   "slow"   "slow"   "slow"   "middle"
##  [8] "middle" "middle" "middle" "middle" "middle" "middle"
## [15] "middle" "middle" "middle" "middle" "middle" "middle" "middle"
## [22] "middle" "middle" "middle" "middle" "middle" "middle" "middle"
## [29] "middle" "middle" "middle" "quick"  "quick"  "quick"  "quick"
## [36] "quick"  "quick"  "quick"  "quick"  "quick"  "quick"  "quick"
## [43] "quick"  "quick"  "quick"  "quick"  "quick"  "quick"  "quick"
## [50] "quick"
```

Subsetting data

```
require(car)
mydata <- cars
#selecting by observation values
mydata.sub1 <- subset(mydata, subset = speed >= 10)

#selecting by columns
mydata.sub2 <- subset(mydata, select = c(speed))

#reverse selecting
mydata.sub3 <- subset(mydata, select = -c(speed))
```

- Lab Practice 3: In the file you imported to R, code missing values to NA (if already coded as NA, code missing values to -99). Find another variable and code it into a dummy (categorical variable).