

# Introduction to R Programming

P. He

STATS 203, Winter 2010

# Outline

- 1 Get Started: Dealing with Data
- 2 Higher Level

# Read in / Save Data

- `mydata = read.table('location of the data')`
  - `mydata = read.table('Bacteria.txt')` /when the data file is in the same directory as R
  - `mydata = read.table('c:/data/Bacteria.txt')` /when the data file is in a different directory
  - `mydata = read.table('http://www-stat.stanford.edu/~nzhang/203_web/Data/Bacteria.txt')` /when data file is online
- Other read in functions: `read.csv()`, `scan()`, `read.fwf`
- Save data
  - `save.image()` /next time use "`load('.RData')`" to recover all data values from previous time
  - `save(x,y,'xy.Rdata')`
  - `write.table(x, file="", ...)`

# Read in / Save Data

- `mydata = read.table('location of the data')`
  - `mydata = read.table('Bacteria.txt')` /when the data file is in the same directory as R
  - `mydata = read.table('c:/data/Bacteria.txt')` /when the data file is in a different directory
  - `mydata = read.table('http://www-stat.stanford.edu/~nzhang/203_web/Data/Bacteria.txt')` /when data file is online
- Other read in functions: `read.csv()`, `scan()`, `read.fxf`
- Save data
  - `save.image()` /next time use "`load('.RData')`" to recover all data values from previous time
  - `save(x,y,'xy.Rdata')`
  - `write.table(x, file="", ...)`

# Read in / Save Data

- `mydata = read.table('location of the data')`
  - `mydata = read.table('Bacteria.txt')` /when the data file is in the same directory as R
  - `mydata = read.table('c:/data/Bacteria.txt')` /when the data file is in a different directory
  - `mydata = read.table('http://www-stat.stanford.edu/~nzhang/203_web/Data/Bacteria.txt')` /when data file is online
- Other read in functions: `read.csv()`, `scan()`, `read.fxf`
- Save data
  - `save.image()` /next time use “`load('.RData')`” to recover all data values from previous time
  - `save(x,y,'xy.Rdata')`
  - `write.table(x, file="", ...)`

# “Data Structure”

- Vectors: `x = c(1,2,3,4)` / `x = 1:4` / `x = seq(from=1, to = 4, by = 1)`
- Matrix: `matrix(1:6, nrow = 2, ncol = 3 /,byrow = TRUE)`
- Dataframe: Matrix with columns possibly of differing modes and attributes
  - `x = 1:4 ; y = c('a', 'b', 'c', 'd')` ; `xyDF = data.frame(x,y)`

# “Data Structure”

- Vectors: `x = c(1,2,3,4)` / `x = 1:4` / `x = seq(from=1, to = 4, by = 1)`
- Matrix: `matrix(1:6, nrow = 2, ncol = 3 /,byrow = TRUE)`
- Dataframe: Matrix with columns possibly of differing modes and attributes
  - `x = 1:4 ; y = c('a', 'b', 'c', 'd')` ; `xyDF = data.frame(x,y)`

# “Data Structure”

- Vectors: `x = c(1,2,3,4)` / `x = 1:4` / `x = seq(from=1, to = 4, by = 1)`
- Matrix: `matrix(1:6, nrow = 2, ncol = 3 /,byrow = TRUE)`
- Dataframe: Matrix with columns possibly of differing modes and attributes
  - `x = 1:4 ; y = c('a', 'b', 'c', 'd')` ; `xyDF = data.frame(x,y)`

# Operations on Data

- Numerical:

- vectors: +, -, \*, /, length(x)
- matrix: +, -, %\*%, solve(x, b) \*means solve for A in  $Ax=b$ , dim(), diag(), det(), t(), qr(), eigen(), svd()

- Statistical: mean, var(), sd(), max(), min(), quantile(), sum(), median(), rnorm/qnorm/pnorm,etc

- Index:

- vectors: x[1:3], x[x>=5]; x[c(1,3)]
- matrix: x[2,3], x[,4], x[-(1:2),]
- dataframe: data\$x[1:2]

# Operations on Data

- Numerical:
  - vectors: `+`, `-`, `*`, `/`, `length(x)`
  - matrix: `+`, `-`, `%*%`, `solve(x, b)` \*means solve for A in  $Ax=b$  , `dim()`, `diag()`, `det()`, `t()`, `qr()`, `eigen()`, `svd()`
- Statistical: `mean`, `var()`, `sd()`, `max()`, `min()`, `quantile()`, `sum()`, `median()`, `rnorm/qnorm/pnorm`, etc
- Index:
  - vectors: `x[1:3]`, `x[x>=5]`; `x[c(1,3)]`
  - matrix: `x[2,3]`, `x[,4]`, `x[-(1:2),]`
  - dataframe: `data$x[1:2]`

# Operations on Data

- Numerical:
  - vectors: +, -, \*, /, length(x)
  - matrix: +, -, %\*%, solve(x, b) \*means solve for A in  $Ax=b$ , dim(), diag(), det(), t(), qr(), eigen(), svd()
- Statistical: mean, var(), sd(), max(), min(), quantile(), sum(), median(), rnorm/qnorm/pnorm,etc
- Index:
  - vectors: x[1:3], x[x>=5]; x[c(1,3)]
  - matrix: x[2,3], x[,4], x[-(1:2),]
  - dataframe: data\$x[1:2]

# Control Structure

- If/Else: `if(condition){statement 1} else{statement 2}`
- Loop:
  - `for (i in 1:10){statement}`
  - `while(x<0){statement}`

# Control Structure

- If/Else: `if(condition){statement 1} else{statement 2}`
- Loop:
  - `for (i in 1:10){statement}`
  - `while(x<0){statement}`

# Function

- Write a function
  - `mymean -> function(x){m = sum(x)/length(x); m}`
- Execute the function
  - save as script file "mymean.R" ; `source('mymean.R')`
  - `x.mean=mymean(x)`

# Function

- Write a function
  - `mymean -> function(x){m = sum(x)/length(x); m}`
- Execute the function
  - save as script file "mymean.R" ; `source('mymean.R')`
  - `x.mean=mymean(x)`

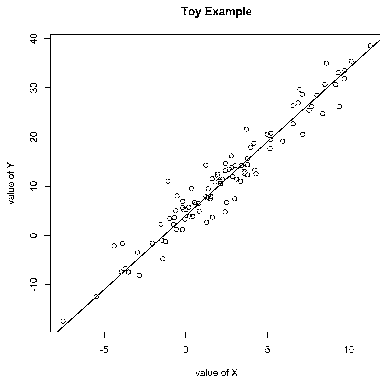
# Graphics

- Plotting functions
  - `plot(x)`, `plot(x,y)`, `boxplot(x)`, `hist(x)`, `qqplot(x,y)`, `image(x,y,z)`
  - `points(x,y)`, `lines(x,y)`, `abline(a,b)`, `title()`
  - parameters: `type`, `xlim/ylim`, `xlab/ylab`, `main`, etc

# Graphics Example

Ex:

```
x=rnorm(100, mean = 2, sd = 4); y= 3*x+4+rnorm(100, sd = 3);  
plot(x,y, xlab = 'value of X', ylab = 'value of Y', main = 'Toy Example');  
abline(4,3);
```



# Statistical Analysis

- Regression:

- `tmp = lm(y~x1+x2)` model:  $y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \varepsilon$
- `summary(tmp)`; `tmp$coefficients`, `tmp$residuals`, `tmp$fitted.values`

- Regression formulae

- remove the constant term: `tmp1 = lm(y~x-1)` model:  $y = ax + \varepsilon$
- interaction term: `tmp = lm(y~x1:x2)` model:  $y = \beta x_1x_2 + \varepsilon$
- full model: `tmp = lm(y~x1*x2)` model:  

$$y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_1x_2 + \varepsilon$$

# Statistical Analysis

- Regression:
  - `tmp = lm(y~x1+x2)` model:  $y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \varepsilon$
  - `summary(tmp)`; `tmp$coefficients`, `tmp$residuals`, `tmp$fitted.values`
- Regression formulae
  - remove the constant term: `tmp1 = lm(y~x-1)` model:  $y = ax + \varepsilon$
  - interaction term: `tmp = lm(y~x1:x2)` model:  $y = \beta x_1x_2 + \varepsilon$
  - full model: `tmp = lm(y~x1*x2)` model:
 
$$y = \beta_0 + \beta_1x_1 + \beta_2x_2 + \beta_3x_1x_2 + \varepsilon$$

# Reference

- R tutorial by Yuehwen Liao last year
- <http://cran.r-project.org/>
- Have a problem?
  - google R and Function Name
  - ?(function name)
  - ?.search(function name)