STAT 6210 - R Session 2

Contents

Working Directory and Data Loading In order to open a file on R, you should first know what is the *current working directory*.

Text

```
getwd()
```

```
## [1] "/Users/abarut/Dropbox/Teaching/STAT 6210 - Data Analysis/Lecture Notes"
```

To get a list of all the files in the current working directory, use `list.files`.

```
list.files()
```

```
##  [1] "Lec1.key"     "Lec1.pdf"     "Lec2.key"     "R1.html"
##  [5] "R1.pdf"       "R1.Rmd"       "R2.html"      "R2.pdf"
##  [9] "R2.Rmd"       "r2tests.html" "r2tests.Rmd"
```

Current working directory can be changed with `setwd`.

```
setwd("/Users/abarut/Downloads/")
getwd()
```

```
## [1] "/Users/abarut/Downloads"
```

Text files can be read with `read.table`.

```
setwd("/Users/abarut/Downloads/")
df <- read.table(file="example1.txt")
head(df)
```

```
##   V1 V2    V3
## 1  1 AL 41.59
## 2  1 AL 41.59
## 3  1 AK 27.67
## 4  1 AK 27.67
## 5  1 AZ 44.67
## 6  1 AZ 44.67
```

If you don't want to change the working directory, you can write down the complete filepath.

```
df <- read.table("/Users/abarut/Downloads/example1.txt")
head(df)
```

```
##   V1 V2    V3
## 1  1 AL 41.59
## 2  1 AL 41.59
## 3  1 AK 27.67
## 4  1 AK 27.67
## 5  1 AZ 44.67
## 6  1 AZ 44.67
```

csv files can be read with `read.csv`.

```r
setwd("/Users/abarut/Downloads/")
df <- read.csv("example1.csv")
head(df)
```

```
##   y state.abb             name  rep state.name gorevote
## 1 1        AL  SESSIONS (R AL) TRUE    Alabama    41.59
## 2 1        AL    SHELBY (R AL) TRUE    Alabama    41.59
## 3 1        AK MURKOWSKI (R AK) TRUE     Alaska    27.67
## 4 1        AK   STEVENS (R AK) TRUE     Alaska    27.67
## 5 1        AZ       KYL (R AZ) TRUE    Arizona    44.67
## 6 1        AZ    MCCAIN (R AZ) TRUE    Arizona    44.67
```

You can read data from the web by replacing the file directory with an http link.

```r
df <- read.csv("http://www.ats.ucla.edu/stat/data/hsb2.csv")
head(df)
```

```
##    id female race ses schtyp prog read write math science socst
## 1  70      0    4   1      1    1   57    52   41      47    57
## 2 121      1    4   2      1    3   68    59   53      63    61
## 3  86      0    4   3      1    1   44    33   54      58    31
## 4 141      0    4   3      1    3   63    44   47      53    56
## 5 172      0    4   2      1    2   47    52   57      53    61
## 6 113      0    4   2      1    2   44    52   51      63    61
```

To read a **SPSS** or **dta** (for SAS) file, you will need the `foreign` package.

```r
#install.packages("foreign")
library(foreign)
# read.spss(...)
# read.dta(...)
```

Similarly, to read Excel files, you can use the `xlsx` package.

`read.csv` and `read.table` have a lot of options. One is `header`, which tells R if the first line in the file should be read as column names or data.

```r
setwd("/Users/abarut/Downloads/")
df <- read.csv("example1.csv", header=FALSE)
head(df)
```

```
##   V1        V2                V3   V4          V5       V6
## 1  y state.abb              name  rep state.name gorevote
## 2  1        AL  SESSIONS (R AL) TRUE    Alabama    41.59
## 3  1        AL    SHELBY (R AL) TRUE    Alabama    41.59
## 4  1        AK MURKOWSKI (R AK) TRUE     Alaska    27.67
## 5  1        AK   STEVENS (R AK) TRUE     Alaska    27.67
## 6  1        AZ       KYL (R AZ) TRUE    Arizona    44.67
```

```r
df <- read.csv("example1.csv", header=TRUE)
head(df)
```

```
##   y state.abb             name  rep state.name gorevote
## 1 1        AL  SESSIONS (R AL) TRUE    Alabama    41.59
## 2 1        AL    SHELBY (R AL) TRUE    Alabama    41.59
## 3 1        AK MURKOWSKI (R AK) TRUE     Alaska    27.67
## 4 1        AK   STEVENS (R AK) TRUE     Alaska    27.67
## 5 1        AZ       KYL (R AZ) TRUE    Arizona    44.67
## 6 1        AZ    MCCAIN (R AZ) TRUE    Arizona    44.67
```

The other important option is `stringsAsFactors`, we will cover this later.

To export an R object as a text file or a csv file, use `write.table` or `read.csv`.

```r
setwd("/Users/abarut/Downloads/")
write.table(df,"my_df.txt")
write.csv(df,"my_df.csv")
```

Data Frames After loading a dataset, you might want to check what it looks like.

```r
head(df)
```

```
##   y state.abb             name  rep state.name gorevote
## 1 1        AL  SESSIONS (R AL) TRUE    Alabama    41.59
## 2 1        AL    SHELBY (R AL) TRUE    Alabama    41.59
## 3 1        AK MURKOWSKI (R AK) TRUE     Alaska    27.67
## 4 1        AK   STEVENS (R AK) TRUE     Alaska    27.67
## 5 1        AZ       KYL (R AZ) TRUE    Arizona    44.67
## 6 1        AZ    MCCAIN (R AZ) TRUE    Arizona    44.67
```

In order to check the data structure of the contents of data, use `str`.

```r
str(df)
```

```
## 'data.frame':    100 obs. of  6 variables:
##  $ y         : int  1 1 1 1 1 1 1 1 0 1 ...
##  $ state.abb : Factor w/ 50 levels "AK","AL","AR",..: 2 2 1 1 4 4 3 3 5 5 ...
##  $ name      : Factor w/ 100 levels "AKAKA (D HI)",..: 85 86 73 92 61 69 52 66 10 40 ...
##  $ rep       : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
##  $ state.name: Factor w/ 50 levels "Alabama","Alaska",..: 1 1 2 2 3 3 4 4 5 5 ...
##  $ gorevote  : num  41.6 41.6 27.7 27.7 44.7 ...
```

Integer (int), **numeric** (num), **character** (chr) and **factor**. *Factor*s come in two types, *ordered* and *unordered*. Factors are used for reducing characters to more managable data structures.

To transform *character*s into *factor*s, use `factor`. `levels` displays the possible values of a factor.

```
levels(df$state.abb)
```

```
##  [1] "AK" "AL" "AR" "AZ" "CA" "CO" "CT" "DE" "FL" "GA" "HI" "IA" "ID" "IL"
## [15] "IN" "KS" "KY" "LA" "MA" "MD" "ME" "MI" "MN" "MO" "MS" "MT" "NC" "ND"
## [29] "NE" "NH" "NJ" "NM" "NV" "NY" "OH" "OK" "OR" "PA" "RI" "SC" "SD" "TN"
## [43] "TX" "UT" "VA" "VT" "WA" "WI" "WV" "WY"
```

Other useful functions for diagnostics are `summary`, `dim` and `colnames`.

```
dim(df)
```

```
## [1] 100   6
```

```
summary(df)
```

```
##        y          state.abb             name          rep
##  Min.   :0.00   AK     : 2   AKAKA (D HI)  : 1   Mode :logical
##  1st Qu.:1.00   AL     : 2   ALLARD (R CO) : 1   FALSE:51
##  Median :1.00   AR     : 2   ALLEN (R VA)  : 1   TRUE :49
##  Mean   :0.77   AZ     : 2   BAUCUS (D MT) : 1   NA's :0
##  3rd Qu.:1.00   CA     : 2   BAYH (D IN)   : 1
##  Max.   :1.00   CO     : 2   BENNETT (R UT): 1
##                 (Other):88   (Other)       :94
##       state.name    gorevote
##  Alabama   : 2   Min.   :26.3
##  Alaska    : 2   1st Qu.:40.9
##  Arizona   : 2   Median :46.2
##  Arkansas  : 2   Mean   :45.2
##  California: 2   3rd Qu.:50.6
##  Colorado  : 2   Max.   :61.0
##  (Other)   :88
```

```
colnames(df)
```

```
## [1] "y"          "state.abb"  "name"       "rep"        "state.name"
## [6] "gorevote"
```

You can use `subset` to refer to certain columns.

```
subset(df,select=state.abb:gorevote)
```

```
##     state.abb               name   rep     state.name gorevote
## 1          AL     SESSIONS (R AL)  TRUE        Alabama    41.59
## 2          AL       SHELBY (R AL)  TRUE        Alabama    41.59
## 3          AK    MURKOWSKI (R AK)  TRUE         Alaska    27.67
## 4          AK      STEVENS (R AK)  TRUE         Alaska    27.67
```

```
## 5           AZ        KYL (R AZ)  TRUE        Arizona    44.67
## 6           AZ     MCCAIN (R AZ)  TRUE        Arizona    44.67
## 7           AR HUTCHINSON (R AR)  TRUE       Arkansas    45.86
## 8           AR    LINCOLN (D AR) FALSE       Arkansas    45.86
## 9           CA      BOXER (D CA) FALSE     California    53.45
## 10          CA  FEINSTEIN (D CA) FALSE     California    53.45
## 11          CO     ALLARD (R CO)  TRUE       Colorado    42.39
## 12          CO   CAMPBELL (R CO)  TRUE       Colorado    42.39
## 13          CT       DODD (D CT) FALSE    Connecticut    55.91
## 14          CT  LIEBERMAN (D CT) FALSE    Connecticut    55.91
## 15          DE      BIDEN (D DE) FALSE       Delaware    54.96
## 16          DE     CARPER (D DE) FALSE       Delaware    54.96
## 17          FL     GRAHAM (D FL) FALSE        Florida    48.84
## 18          FL     NELSON (D FL) FALSE        Florida    48.84
## 19          GA    CLELAND (D GA) FALSE        Georgia    42.98
## 20          GA     MILLER (D GA) FALSE        Georgia    42.98
## 21          HI      AKAKA (D HI) FALSE         Hawaii    55.79
## 22          HI     INOUYE (D HI) FALSE         Hawaii    55.79
## 23          ID      CRAIG (R ID)  TRUE          Idaho    27.64
## 24          ID      CRAPO (R ID)  TRUE          Idaho    27.64
## 25          IL     DURBIN (D IL) FALSE       Illinois    54.60
## 26          IL FITZGERALD (R IL)  TRUE       Illinois    54.60
## 27          IN       BAYH (D IN) FALSE        Indiana    41.01
## 28          IN      LUGAR (R IN)  TRUE        Indiana    41.01
## 29          IA   GRASSLEY (R IA)  TRUE           Iowa    48.54
## 30          IA     HARKIN (D IA) FALSE           Iowa    48.54
## 31          KS  BROWNBACK (R KS)  TRUE         Kansas    37.24
## 32          KS    ROBERTS (R KS)  TRUE         Kansas    37.24
## 33          KY    BUNNING (R KY)  TRUE       Kentucky    41.37
## 34          KY  MCCONNELL (R KY)  TRUE       Kentucky    41.37
## 35          LA     BREAUX (D LA) FALSE      Louisiana    44.88
## 36          LA   LANDRIEU (D LA) FALSE      Louisiana    44.88
## 37          ME    COLLINS (R ME)  TRUE          Maine    49.09
## 38          ME      SNOWE (R ME)  TRUE          Maine    49.09
## 39          MD   MIKULSKI (D MD) FALSE       Maryland    56.57
## 40          MD   SARBANES (D MD) FALSE       Maryland    56.57
## 41          MA    KENNEDY (D MA) FALSE  Massachusetts    59.80
## 42          MA      KERRY (D MA) FALSE  Massachusetts    59.80
## 43          MI   STABENOW (D MI) FALSE       Michigan    51.28
## 44          MI      LEVIN (D MI) FALSE       Michigan    51.28
## 45          MN     DAYTON (D MN) FALSE      Minnesota    47.91
## 46          MN  WELLSTONE (D MN) FALSE      Minnesota    47.91
## 47          MS    COCHRAN (R MS)  TRUE    Mississippi    40.70
## 48          MS       LOTT (R MS)  TRUE    Mississippi    40.70
## 49          MO   CARNAHAN (D MO) FALSE       Missouri    47.08
## 50          MO       BOND (R MO)  TRUE       Missouri    47.08
## 51          MT     BAUCUS (D MT) FALSE        Montana    33.36
## 52          MT      BURNS (R MT)  TRUE        Montana    33.36
## 53          NE      HAGEL (R NE)  TRUE       Nebraska    33.25
## 54          NE     NELSON (D NE) FALSE       Nebraska    33.25
## 55          NV     ENSIGN (R NV)  TRUE         Nevada    45.98
## 56          NV       REID (D NV) FALSE         Nevada    45.98
## 57          NH      GREGG (R NH)  TRUE  New Hampshire    46.80
## 58          NH      SMITH (R NH)  TRUE  New Hampshire    46.80
```

```
## 59         NJ      CORZINE (D NJ) FALSE     New Jersey    56.13
## 60         NJ   TORRICELLI (D NJ) FALSE     New Jersey    56.13
## 61         NM     BINGAMAN (D NM) FALSE     New Mexico    47.91
## 62         NM     DOMENICI (R NM)  TRUE     New Mexico    47.91
## 63         NY      CLINTON (D NY) FALSE       New York    60.21
## 64         NY      SCHUMER (D NY) FALSE       New York    60.21
## 65         NC      EDWARDS (D NC) FALSE North Carolina    43.20
## 66         NC        HELMS (R NC)  TRUE North Carolina    43.20
## 67         ND       CONRAD (D ND) FALSE   North Dakota    33.05
## 68         ND       DORGAN (D ND) FALSE   North Dakota    33.05
## 69         OH       DEWINE (R OH)  TRUE           Ohio    46.46
## 70         OH    VOINOVICH (R OH)  TRUE           Ohio    46.46
## 71         OK       INHOFE (R OK)  TRUE       Oklahoma    38.43
## 72         OK      NICKLES (R OK)  TRUE       Oklahoma    38.43
## 73         OR        SMITH (R OR)  TRUE         Oregon    46.96
## 74         OR        WYDEN (D OR) FALSE         Oregon    46.96
## 75         PA     SANTORUM (R PA)  TRUE   Pennsylvania    50.60
## 76         PA      SPECTER (R PA)  TRUE   Pennsylvania    50.60
## 77         RI       CHAFEE (R RI)  TRUE   Rhode Island    60.99
## 78         RI         REED (D RI) FALSE   Rhode Island    60.99
## 79         SC     HOLLINGS (D SC) FALSE South Carolina    40.91
## 80         SC     THURMOND (R SC)  TRUE South Carolina    40.91
## 81         SD      DASCHLE (D SD) FALSE   South Dakota    37.56
## 82         SD      JOHNSON (D SD) FALSE   South Dakota    37.56
## 83         TN        FRIST (R TN)  TRUE      Tennessee    47.28
## 84         TN     THOMPSON (R TN)  TRUE      Tennessee    47.28
## 85         TX        GRAMM (R TX)  TRUE          Texas    37.98
## 86         TX    HUTCHISON (R TX)  TRUE          Texas    37.98
## 87         UT      BENNETT (R UT)  TRUE           Utah    26.34
## 88         UT        HATCH (R UT)  TRUE           Utah    26.34
## 89         VT JEFFORDS (Indep VT) FALSE        Vermont    50.63
## 90         VT        LEAHY (D VT) FALSE        Vermont    50.63
## 91         VA        ALLEN (R VA)  TRUE       Virginia    44.44
## 92         VA       WARNER (R VA)  TRUE       Virginia    44.44
## 93         WA     CANTWELL (D WA) FALSE     Washington    50.13
## 94         WA       MURRAY (D WA) FALSE     Washington    50.13
## 95         WV         BYRD (D WV) FALSE  West Virginia    45.59
## 96         WV  ROCKEFELLER (D WV) FALSE  West Virginia    45.59
## 97         WI     FEINGOLD (D WI) FALSE      Wisconsin    47.83
## 98         WI         KOHL (D WI) FALSE      Wisconsin    47.83
## 99         WY         ENZI (R WY)  TRUE        Wyoming    27.70
## 100        WY       THOMAS (R WY)  TRUE        Wyoming    27.70
```

```
subset(df,y==0,select=state.abb:gorevote)
```

```
##    state.abb                name   rep    state.name gorevote
## 9         CA        BOXER (D CA) FALSE    California    53.45
## 17        FL       GRAHAM (D FL) FALSE       Florida    48.84
## 21        HI        AKAKA (D HI) FALSE        Hawaii    55.79
## 22        HI       INOUYE (D HI) FALSE        Hawaii    55.79
## 25        IL       DURBIN (D IL) FALSE      Illinois    54.60
## 39        MD     MIKULSKI (D MD) FALSE      Maryland    56.57
## 40        MD     SARBANES (D MD) FALSE      Maryland    56.57
## 41        MA      KENNEDY (D MA) FALSE Massachusetts    59.80
```

```
## 43         MI     STABENOW (D MI) FALSE      Michigan    51.28
## 44         MI        LEVIN (D MI) FALSE      Michigan    51.28
## 45         MN       DAYTON (D MN) FALSE     Minnesota    47.91
## 46         MN    WELLSTONE (D MN) FALSE     Minnesota    47.91
## 59         NJ      CORZINE (D NJ) FALSE    New Jersey    56.13
## 61         NM     BINGAMAN (D NM) FALSE    New Mexico    47.91
## 67         ND       CONRAD (D ND) FALSE  North Dakota    33.05
## 74         OR        WYDEN (D OR) FALSE        Oregon    46.96
## 77         RI      CHAFEE (R RI)  TRUE  Rhode Island    60.99
## 78         RI         REED (D RI) FALSE  Rhode Island    60.99
## 89         VT JEFFORDS (Indep VT) FALSE       Vermont    50.63
## 90         VT        LEAHY (D VT) FALSE       Vermont    50.63
## 94         WA       MURRAY (D WA) FALSE    Washington    50.13
## 95         WV         BYRD (D WV) FALSE West Virginia    45.59
## 97         WI     FEINGOLD (D WI) FALSE     Wisconsin    47.83
```

ggplot `ggplot` is the most frequently used R graphics package (with `lattice` being the close second). We first install it with

```
#install.packages("ggplot2")
#install.packages("reshape")
```

And load the package

```
library(ggplot2)
#or require(ggplot2)
library(reshape)
```

We will use an online dataset. This data contains standerdized scores from a national survey of high school seniors. Descriptions of variables are given below

id	scale	student id
female	nominal	(0/1)
race	nominal	ethnicity (1=hispanic 2=asian 3=african-amer 4=white)
ses	ordinal	(1=low 2=middle 3=high)
schtyp	nominal	type of school (1=public 2=private)
prog	nominal	type of program (1=general 2=academic 3=vocational)
read	scale	standardized reading score
write	scale	standardized writing score
math	scale	standardized math score
science	scale	standardized science score
socst	scale	standardized social studies score
hon	nominal	honors english (0/1)

```
df <- read.csv("http://www.ats.ucla.edu/stat/data/hsb2.csv")
```

```r
dim(df)
```

```
## [1] 200  11
```

```r
str(df)
```

```
## 'data.frame':    200 obs. of  11 variables:
##  $ id     : int  70 121 86 141 172 113 50 11 84 48 ...
##  $ female : int  0 1 0 0 0 0 0 0 0 0 ...
##  $ race   : int  4 4 4 4 4 4 3 1 4 3 ...
##  $ ses    : int  1 2 3 3 2 2 2 2 2 2 ...
##  $ schtyp : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ prog   : int  1 3 1 3 2 2 1 2 1 2 ...
##  $ read   : int  57 68 44 63 47 44 50 34 63 57 ...
##  $ write  : int  52 59 33 44 52 52 59 46 57 55 ...
##  $ math   : int  41 53 54 47 57 51 42 45 54 52 ...
##  $ science: int  47 63 58 53 53 63 53 39 58 50 ...
##  $ socst  : int  57 61 31 56 61 61 61 36 51 51 ...
```

```r
summary(df)
```

```
##        id            female            race             ses       
##  Min.   :  1.0   Min.   :0.000   Min.   :1.00   Min.   :1.00  
##  1st Qu.: 50.8   1st Qu.:0.000   1st Qu.:3.00   1st Qu.:2.00  
##  Median :100.5   Median :1.000   Median :4.00   Median :2.00  
##  Mean   :100.5   Mean   :0.545   Mean   :3.43   Mean   :2.06  
##  3rd Qu.:150.2   3rd Qu.:1.000   3rd Qu.:4.00   3rd Qu.:3.00  
##  Max.   :200.0   Max.   :1.000   Max.   :4.00   Max.   :3.00  
##      schtyp          prog           read          write     
##  Min.   :1.00   Min.   :1.00   Min.   :28.0   Min.   :31.0  
##  1st Qu.:1.00   1st Qu.:2.00   1st Qu.:44.0   1st Qu.:45.8  
##  Median :1.00   Median :2.00   Median :50.0   Median :54.0  
##  Mean   :1.16   Mean   :2.02   Mean   :52.2   Mean   :52.8  
##  3rd Qu.:1.00   3rd Qu.:2.25   3rd Qu.:60.0   3rd Qu.:60.0  
##  Max.   :2.00   Max.   :3.00   Max.   :76.0   Max.   :67.0  
##       math         science         socst     
##  Min.   :33.0   Min.   :26.0   Min.   :26.0  
##  1st Qu.:45.0   1st Qu.:44.0   1st Qu.:46.0  
##  Median :52.0   Median :53.0   Median :52.0  
##  Mean   :52.6   Mean   :51.9   Mean   :52.4  
##  3rd Qu.:59.0   3rd Qu.:58.0   3rd Qu.:61.0  
##  Max.   :75.0   Max.   :74.0   Max.   :71.0
```

```r
summary(subset(df, read >= 60))
```

```
##        id            female            race           ses       
##  Min.   :  3.0   Min.   :0.000   Min.   :1.0   Min.   :1.00  
##  1st Qu.: 76.5   1st Qu.:0.000   1st Qu.:4.0   1st Qu.:2.00  
##  Median :108.5   Median :0.000   Median :4.0   Median :3.00  
##  Mean   :109.8   Mean   :0.482   Mean   :3.7   Mean   :2.38  
##  3rd Qu.:143.2   3rd Qu.:1.000   3rd Qu.:4.0   3rd Qu.:3.00
```

```
##  Max.   :200.0   Max.   :1.000   Max.   :4.0   Max.   :3.00
##      schtyp          prog           read          write
##  Min.   :1.00   Min.   :1.00   Min.   :60.0   Min.   :43.0
##  1st Qu.:1.00   1st Qu.:2.00   1st Qu.:63.0   1st Qu.:57.0
##  Median :1.00   Median :2.00   Median :65.0   Median :60.0
##  Mean   :1.18   Mean   :1.95   Mean   :65.5   Mean   :59.5
##  3rd Qu.:1.00   3rd Qu.:2.00   3rd Qu.:68.0   3rd Qu.:65.0
##  Max.   :2.00   Max.   :3.00   Max.   :76.0   Max.   :67.0
##       math         science         socst
##  Min.   :35.0   Min.   :44.0   Min.   :41.0
##  1st Qu.:55.8   1st Qu.:55.0   1st Qu.:56.0
##  Median :60.5   Median :61.0   Median :61.0
##  Mean   :60.2   Mean   :59.7   Mean   :60.9
##  3rd Qu.:66.2   3rd Qu.:65.2   3rd Qu.:66.0
##  Max.   :75.0   Max.   :74.0   Max.   :71.0
```

We can obtain contingency tables with the `xtabs` function. .

```
xtabs( ~ female, data = df)
```

```
## female
##   0   1
##  91 109
```

```
xtabs( ~ race, data = df)
```

```
## race
##   1   2   3   4
##  24  11  20 145
```

```
xtabs( ~ prog, data = df)
```

```
## prog
##   1   2   3
##  45 105  50
```

```
xtabs( ~ ses + schtyp, data = df)
```

```
##    schtyp
## ses  1  2
##   1 45  2
##   2 76 19
##   3 47 11
```

```
(tab3 <- xtabs( ~ ses + prog + schtyp, data = df))
```

```
## , , schtyp = 1
##
##    prog
## ses  1  2  3
```

```
##   1 14 19 12
##   2 17 30 29
##   3  8 32  7
##
## , , schtyp = 2
##
##    prog
## ses  1  2  3
##   1  2  0  0
##   2  3 14  2
##   3  1 10  0
```

```
(tab2 <- xtabs( ~ ses + schtyp, data = df))
```

```
##    schtyp
## ses  1  2
##   1 45  2
##   2 76 19
##   3 47 11
```

To obtain the column means for all variables, we use `colMeans`

```
colMeans(df)
```

```
##      id  female    race     ses  schtyp    prog    read   write    math
## 100.500   0.545   3.430   2.055   1.160   2.025  52.230  52.775  52.645
## science   socst
##  51.850  52.405
```

If we want column means for different programs, we can do this with the `by` function,

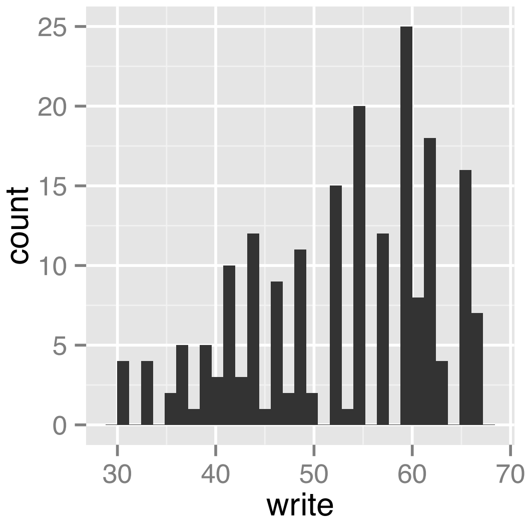
```
by(df[, 7:11], df$prog, colMeans)
```

```
## df$prog: 1
##    read   write    math science   socst
##   49.76   51.33   50.02   52.44   50.60
## --------------------------------------------------------
## df$prog: 2
##    read   write    math science   socst
##   56.16   56.26   56.73   53.80   56.70
## --------------------------------------------------------
## df$prog: 3
##    read   write    math science   socst
##   46.20   46.76   46.42   47.22   45.02
```

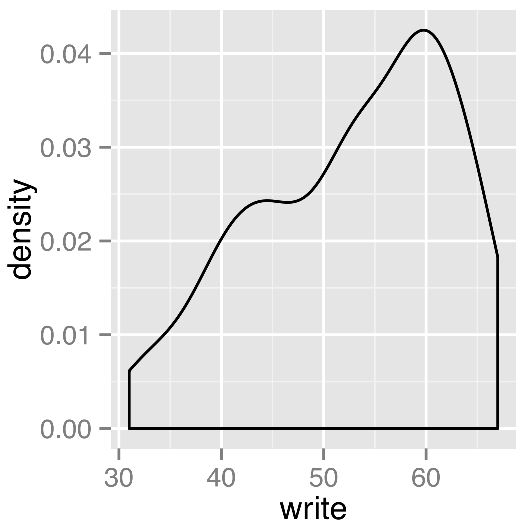
To plot with `ggplot`, we need to feed the data in, and then tell `ggplot` what are the covariates we are interested in. After feeding in the data, we can obtain plots by writing `+ plot type()`.

```
ggplot(df, aes(x = write)) + geom_histogram()
```

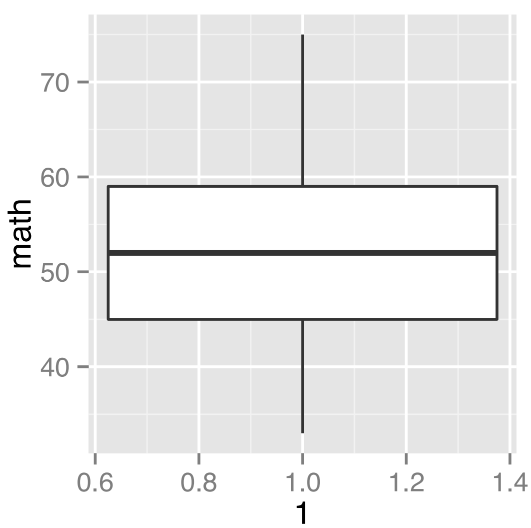
```
## stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```



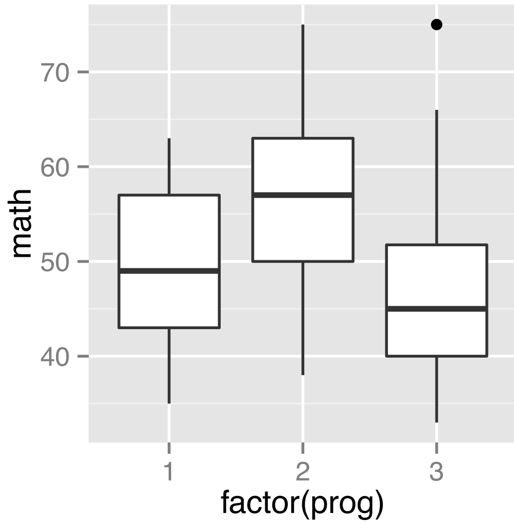

```
ggplot(df, aes(x = write)) + geom_density()
```




```
ggplot(df, aes(x = 1, y = math)) + geom_boxplot()
```

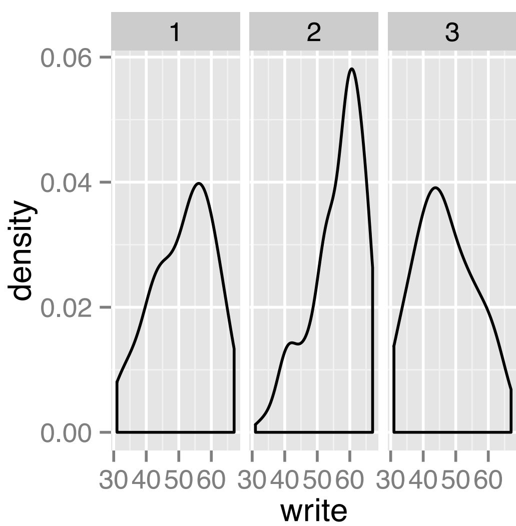


```r
ggplot(df, aes(x = factor(prog), y = math)) + geom_boxplot()
```



We can also add settings with the + function.

```r
ggplot(df, aes(x = write)) + geom_density() + facet_wrap(~ prog)
```



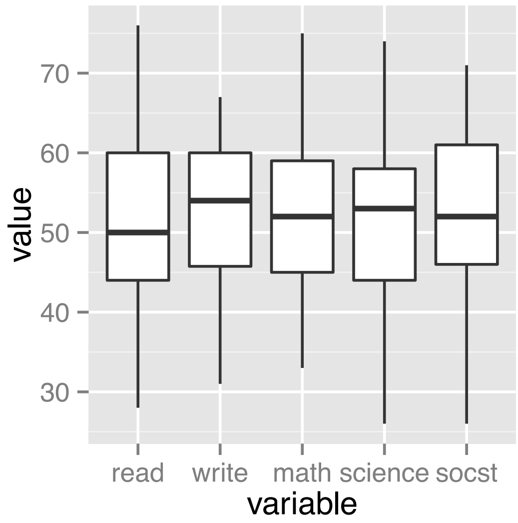
ggplot cannot extract more than one data point from each row. To display multiple columns as different data points, we first need to melt the data.

```r
head(melt(df[,7:11]))
```

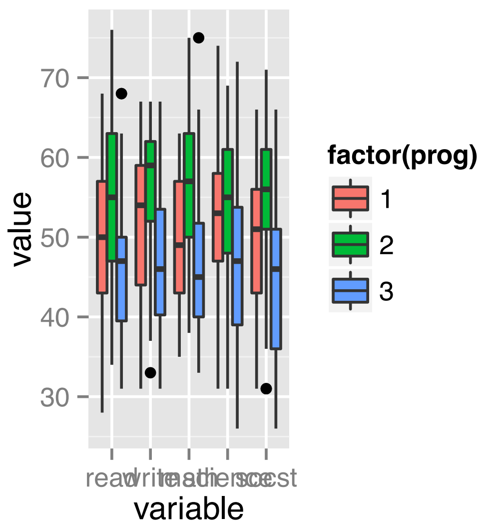
```
## Using  as id variables

##   variable value
## 1     read    57
## 2     read    68
## 3     read    44
## 4     read    63
## 5     read    47
## 6     read    44
```

```r
ggplot(melt(df[, 7:11]), aes(x = variable, y = value)) + geom_boxplot()
```

```
## Using  as id variables
```




```r
ggplot(melt(df[, 6:11], id.vars = "prog"),
       aes(x = variable, y = value, fill = factor(prog))) +
  geom_boxplot()
```



Parametric Tests

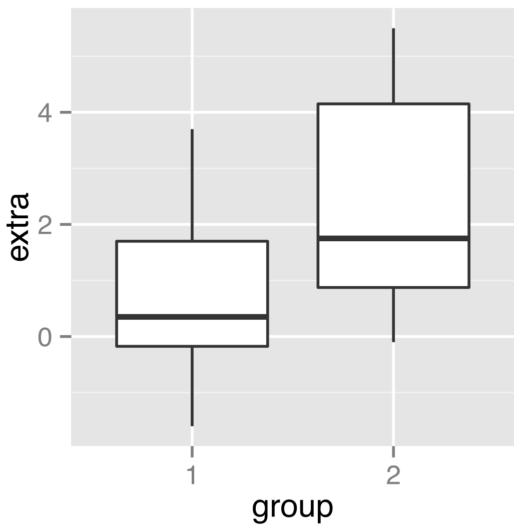
t-test For our first test we will use the same dataset that was used by Gosset in his paper that introduced the t distribution.

We first load the dataset.

```r
data(sleep)
head(sleep)
```

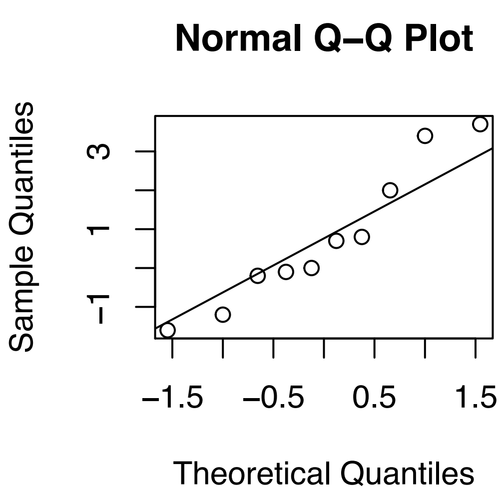
```
##   extra group ID
## 1   0.7     1  1
## 2  -1.6     1  2
## 3  -0.2     1  3
## 4  -1.2     1  4
## 5  -0.1     1  5
## 6   3.4     1  6
```

```r
qplot(group,extra,data=sleep,geom="boxplot")
```

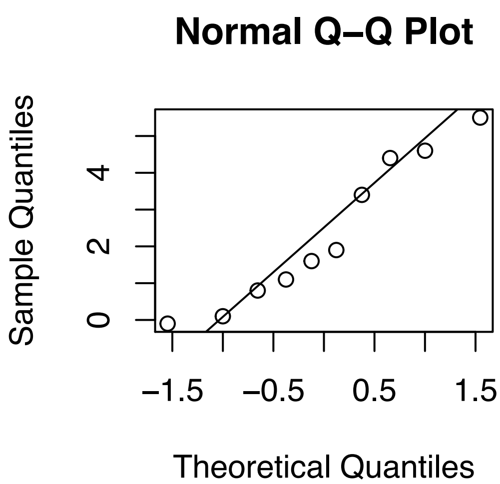


Before doing a t-test, we should check for normality.

```r
sleep1 <- subset(sleep, group==1, extra, drop=TRUE)
#or
#sleep1 <- sleep[sleep$group==1,"extra"]
qqnorm(sleep1)
qqline(sleep1)
```



```
sleep2 <- subset(sleep, group==2, extra, TRUE)
qqnorm(sleep2)
qqline(sleep2)
```



To test the null hypothesis that *the first drug (group=1) results in 0 hour of extra sleep* versus the alternative hypothesis that *it is larger than 0 hour*, we simply write

```
t.test(sleep1,mu=0,alternative=c("greater"))
```

```
##
##  One Sample t-test
##
## data:  sleep1
## t = 1.326, df = 9, p-value = 0.1088
## alternative hypothesis: true mean is greater than 0
## 95 percent confidence interval:
##  -0.2871     Inf
## sample estimates:
## mean of x
##      0.75
```

A two-sided test can be done via

```
t.test(sleep1,mu=0,alternative=c("two.sided"))
```

```
##
##  One Sample t-test
##
## data:  sleep1
## t = 1.326, df = 9, p-value = 0.2176
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.5298  2.0298
## sample estimates:
## mean of x
##      0.75
```

To perform a paired test, we write

```
t.test(sleep$extra~sleep$group,paired=TRUE)
```

```
##
##  Paired t-test
##
## data:  sleep$extra by sleep$group
## t = -4.062, df = 9, p-value = 0.002833
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.4599 -0.7001
## sample estimates:
## mean of the differences
##                   -1.58
```

```
#or
#t.test(extra~group,paired=TRUE,data=sleep)
```

You can change the parameter `mu` to change the null (and alternative) hypotheses. The `conf.level` parameter can be changed to see other confidence intervals.

For an unpaired test, there is an another important parameter: `var.equal`. If you *believe* that the samples have the same variance, you can increase the power of your test by changing this parameter to `TRUE`. The default is `FALSE`.

Binomial test of proportions Two tests are available for testing proportions. `prop.test` uses the normality approximation, and `binom.test` gives the exact p-value.

Let us first generate the dataset.

```
set.seed(1)
x1 <- rbinom(n=20,size = 1,prob = .4)
x1s <- sum(x1)
```

We first test if the probability is equal to 0.6.

```
prop.test(x1s,n=20,p = .6, alternative="two.sided")
```

```
##
##  1-sample proportions test with continuity correction
##
## data:  x1s out of 20, null probability 0.6
## X-squared = 0.4688, df = 1, p-value = 0.4936
## alternative hypothesis: true p is not equal to 0.6
## 95 percent confidence interval:
##  0.2785 0.7215
## sample estimates:
##   p
## 0.5
```

We repeat the same analysis with the exact test.

```r
binom.test(x1s,n=20,p = .6, alternative="two.sided")
```

```
##
##  Exact binomial test
##
## data:  x1s and 20
## number of successes = 10, number of trials = 20, p-value = 0.3703
## alternative hypothesis: true probability of success is not equal to 0.6
## 95 percent confidence interval:
##  0.272 0.728
## sample estimates:
## probability of success
##                    0.5
```

If we want to compare the proportion from two groups, we have to use `prop.test`.

```r
x2 <- rbinom(n=15,size = 1,prob = .65)
x2s <- sum(x2)
prop.test(c(x1s,x2s),c(20,15),alternative="two.sided")
```

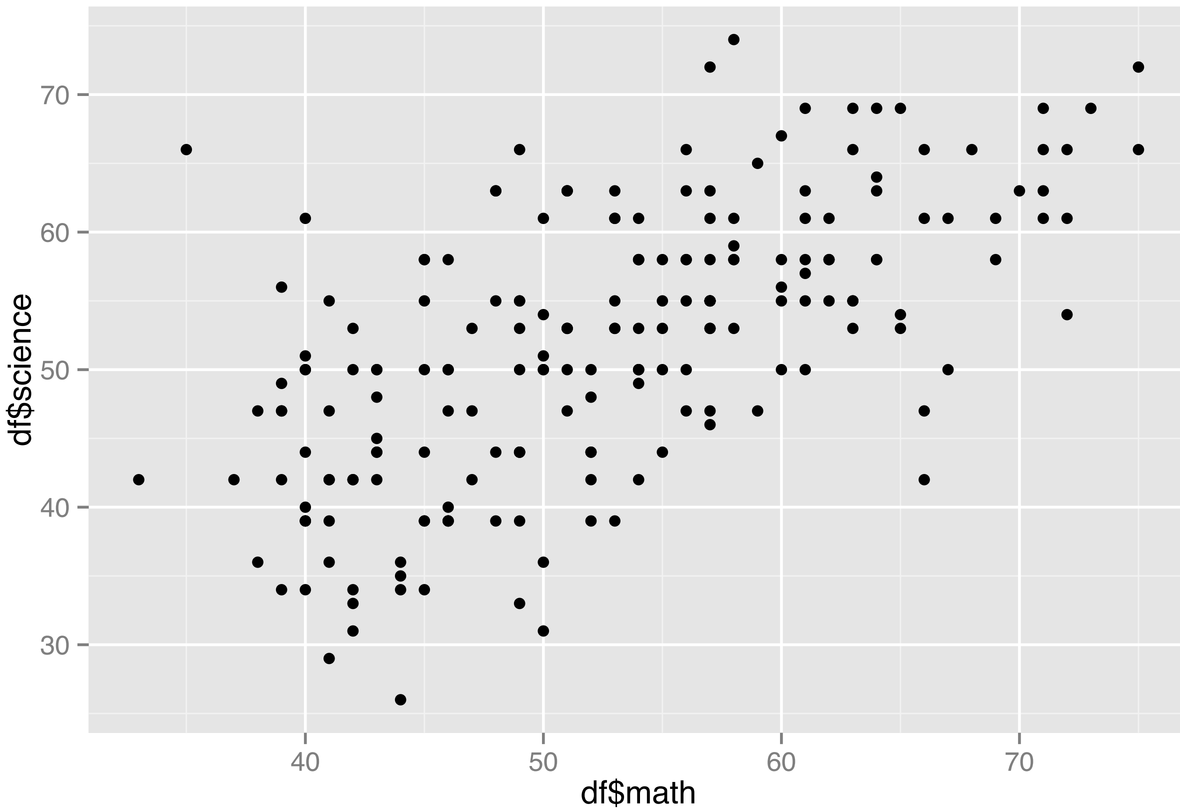
```
##
##  2-sample test for equality of proportions with continuity
##  correction
##
## data:  c(x1s, x2s) out of c(20, 15)
## X-squared = 1.094, df = 1, p-value = 0.2956
## alternative hypothesis: two.sided
## 95 percent confidence interval:
##  -0.6049  0.1382
## sample estimates:
## prop 1 prop 2
## 0.5000 0.7333
```

Testing Correlation We will use the high school senior dataset to demonstrate correlation tests.

```r
df <- read.csv("http://www.ats.ucla.edu/stat/data/hsb2.csv")
```

To test if math and science scores are correlated, we can use `cor.test`. We first plot the dataset.

```r
qplot(df$math,df$science)
```




```r
cor(df$math,df$science)
```

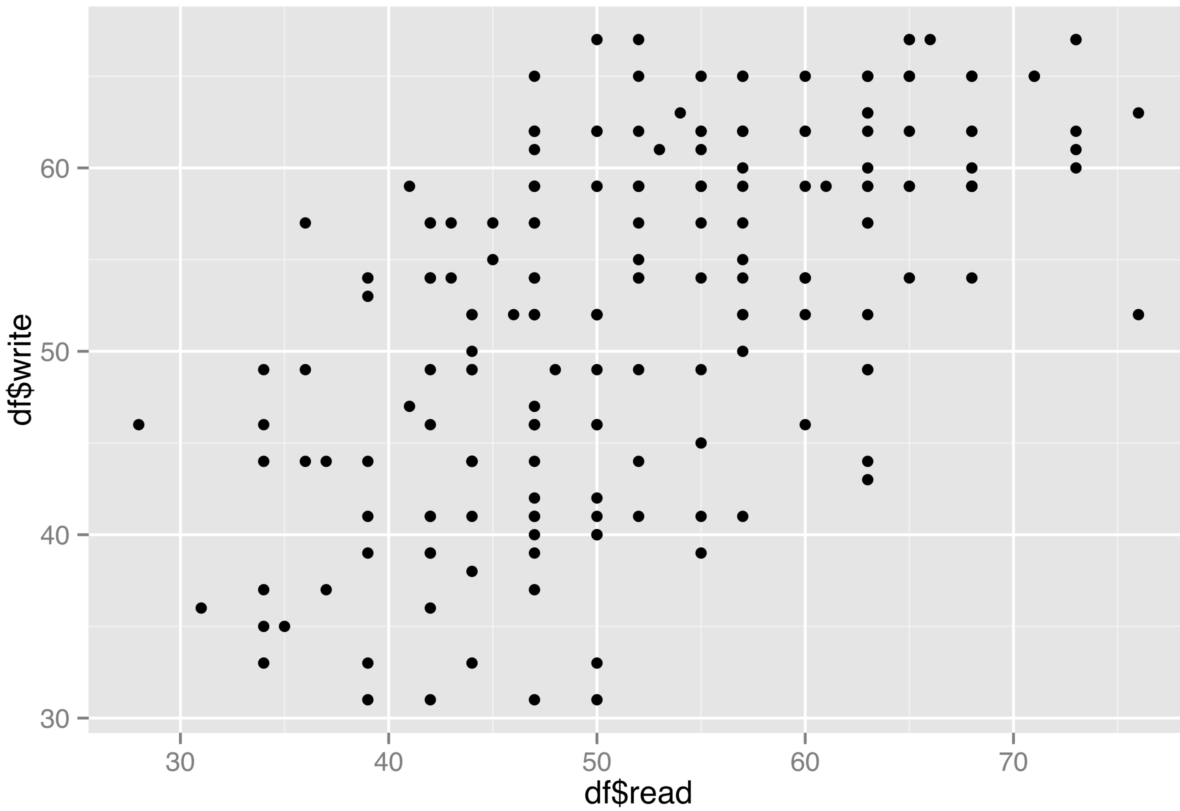
```
## [1] 0.6307
```

```r
cor.test(df$math,df$science,alternative = "two.sided")
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$math and df$science
## t = 11.44, df = 198, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.5392 0.7076
## sample estimates:
##    cor
## 0.6307
```

We repeat the analysis with read and write scores.

```r
qplot(df$read,df$write)
```




```
cor(df$read,df$write)
```

```
## [1] 0.5968
```

```
cor.test(df$read,df$write,alternative = "two.sided")
```

```
##
##  Pearson's product-moment correlation
##
## data:  df$read and df$write
## t = 10.47, df = 198, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.4994 0.6793
## sample estimates:
##    cor
## 0.5968
```

F-test F tests are used to test for the equality of variances. We will use Fisher's iris dataset.

```
data(iris)
head(iris)
```

```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
```

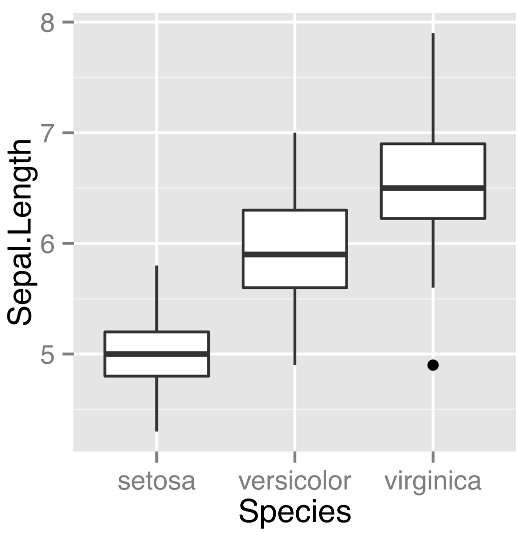
```
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```
summary(iris)
```

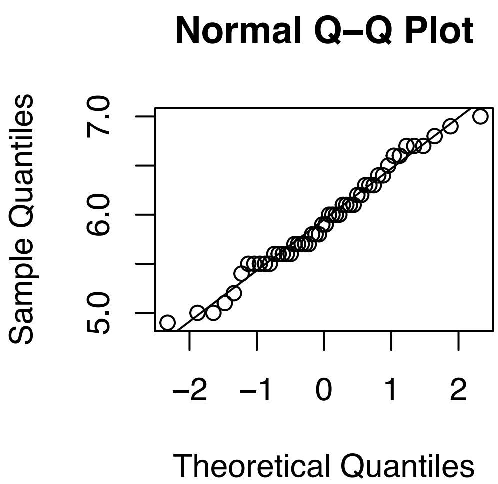
```
##   Sepal.Length   Sepal.Width    Petal.Length   Petal.Width
##  Min.   :4.30   Min.   :2.00   Min.   :1.00   Min.   :0.1
##  1st Qu.:5.10   1st Qu.:2.80   1st Qu.:1.60   1st Qu.:0.3
##  Median :5.80   Median :3.00   Median :4.35   Median :1.3
##  Mean   :5.84   Mean   :3.06   Mean   :3.76   Mean   :1.2
##  3rd Qu.:6.40   3rd Qu.:3.30   3rd Qu.:5.10   3rd Qu.:1.8
##  Max.   :7.90   Max.   :4.40   Max.   :6.90   Max.   :2.5
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

We will compare the variance of the sepal length of *versicolor* versus *virginica*. Let us start with a qualitative look at the normality of our observations.

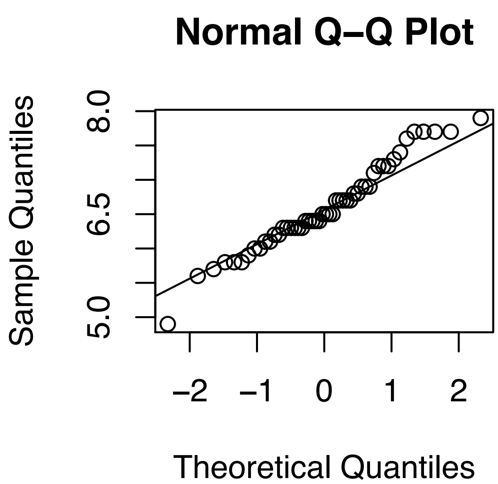
```
ggplot(iris,aes(x=Species,y=Sepal.Length))+geom_boxplot()
```




```
versicolor <- subset(iris, Species=="versicolor", Sepal.Length, drop=TRUE)
qqnorm(versicolor)
qqline(versicolor)
```




```
virginica <- subset(iris, Species=="virginica", Sepal.Length, drop=TRUE)
qqnorm(virginica)
qqline(virginica)
```



To test if the versicolors have the same variance as the virginicas, we need to conduct an F test.

```
var.test(versicolor,virginica)
```

```
##
##  F test to compare two variances
##
## data:  versicolor and virginica
## F = 0.6589, num df = 49, denom df = 49, p-value = 0.1478
## alternative hypothesis: true ratio of variances is not equal to 1
## 95 percent confidence interval:
##  0.3739 1.1612
## sample estimates:
## ratio of variances
##             0.6589
```

The F-test is inconclusive. Let's check for the mean difference.

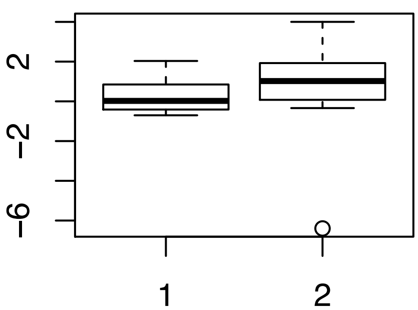
```r
t.test(versicolor,virginica,paired = FALSE, var.equal = TRUE)
```

```
##
##  Two Sample t-test
##
## data:  versicolor and virginica
## t = -5.629, df = 98, p-value = 1.725e-07
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.8819 -0.4221
## sample estimates:
## mean of x mean of y
##     5.936     6.588
```

Non-parametric Tests

Wilcoxon (or Mann-Whitney) for Independent Samples For this part, we will generate random samples from a t distribution with 3 degrees of freedom.

```r
set.seed(1)
x1 <- rt(20,df = 3)
x2 <- rt(15,df = 3) + 1
boxplot(x1,x2)
```



There is a mean shift of 1. Let's see if the Wilcox test can detect that.

```r
wilcox.test(x1,x2,alternative = "two.sided")
```

```
##
##  Wilcoxon rank sum test
##
## data:  x1 and x2
## W = 91, p-value = 0.05029
## alternative hypothesis: true location shift is not equal to 0
```

We also perform a t-test for completeness.

```r
t.test(x1,x2,alternative = "two.sided")
```

```
##
##  Welch Two Sample t-test
##
## data:  x1 and x2
## t = -0.8786, df = 16.27, p-value = 0.3924
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -1.9832  0.8199
## sample estimates:
## mean of x mean of y
##    0.2889    0.8706
```

Wilcoxon test can also be used to obtain confidence intervals for medians.

```r
wilcox.test(x1,x2,alternative = "two.sided",conf.int = TRUE)
```

```
##
##  Wilcoxon rank sum test
##
## data:  x1 and x2
## W = 91, p-value = 0.05029
## alternative hypothesis: true location shift is not equal to 0
## 95 percent confidence interval:
##  -1.619513  0.000493
## sample estimates:
## difference in location
##                -0.6858
```

Wilcoxon for Paired Samples We again use the `sleep` dataset to test if the drugs have similar effects.

```r
wilcox.test(sleep1,sleep2,alternative = "two.sided",paired=TRUE)
```

```
## Warning: cannot compute exact p-value with ties
## Warning: cannot compute exact p-value with zeroes
```

```
##
##  Wilcoxon signed rank test with continuity correction
##
## data:  sleep1 and sleep2
## V = 0, p-value = 0.009091
## alternative hypothesis: true location shift is not equal to 0
```