R review

To use R, we will usually read data from a file. For this class, these files will usually be either `.txt`, `.dat`, or `.csv`. The last version is a comma delimited file, where each field is separated by a comma instead of space or tabs. If you have data in an Excel spreadsheet, you can save it as a `.csv` file so that it can be read into R.

To read in data from a file, you can either type the website of the data (if you are online), or you can read the data from your computer. If you read data from your computer, you need to specify the path to the file, or have the file in the same directory as your R session.

R review

Here we'll give an example of reading in a file from my website

```
x <- read.table("http://www.math.unm.edu/~james/chile.txt",header=T)
```

If copying and pasting this code generates errors in R, then try typing the text directly inside R without copying and pasting (common problems are the quoation marks or the tilde sign copying and pasting differently). Also, try going to the website and see if you can download the data.
If the data exists in your computer in your current directory, you can type

```
x <- read.table("chile.txt",header=T)
```

R review

Here is an example of not reading in a file correctly. In this case, I didn't type `http://` at the start of the URL, and R was unable to find the file. This happens a lot when you either get the URL wrong or try to read in data that is mistakenly in a different directory from your R session.

```
> x <- read.table("www.math.unm.edu/~james/chile.txt")
Error in file(file, "rt") : cannot open the connection
In addition: Warning message:
In file(file, "rt") :
  cannot open file 'www.math.unm.edu/~james/chile.txt':
  No such file or directory
```

R review

Here we extract a variable and print a subset of the observations

```
> x$Length[x$group=="Chimayo"]
 [1] 14.0 15.5 12.5 16.0  8.5 12.5
  15.0 13.0 11.0 10.0 12.8
> x[x$Length>13,]
      group Length Width Thickness
14 Casados   13.5   3.5      1.55
15 Casados   14.0   3.0      1.77
16 Casados   15.0   3.0      1.59
18 Casados   13.5   3.0      1.58
21 Casados   14.0   4.0      1.99
22 Casados   13.3   3.3      1.71
23 Chimayo   14.0   3.5      1.80
24 Chimayo   15.5   3.5      1.81
26 Chimayo   16.0   3.5      1.82
29 Chimayo   15.0   3.5      1.95
```

R review

We'll continue the review using this data set. The data set consists of measurements of length, width, and thickness of green chiles cultivated at four locations in New Mexico in a particular year. There are a total of 44 observations (rows), and four variables (columns). Here are some things you can do with a data set that will be used to review some R functions

```
> head(x) # show the first 6 observations to check that
# data was read in correctly
> head(x)
    group Length Width Thickness
1 Alcalde   10.5   3.0      1.53
2 Alcalde    7.0   3.5      1.76
3 Alcalde   10.5   3.5      1.82
4 Alcalde   11.5   4.0      1.58
5 Alcalde   11.5   3.5      1.84
6 Alcalde    9.5   3.0      1.86
```

```
> table(x$group)
Alcalde Casados Chimayo Cochiti
     11       11       11       11
> mean(x$Length)
[1] 11.075
> sd(x$Width)
[1] 0.5032597
> by(x$Length,x$group,mean)
x$group: Alcalde
[1] 9.25
------------------------------------------------------------
x$group: Casados
[1] 13.3
------------------------------------------------------------
x$group: Chimayo
[1] 12.8
------------------------------------------------------------
x$group: Cochiti
[1] 8.95
```

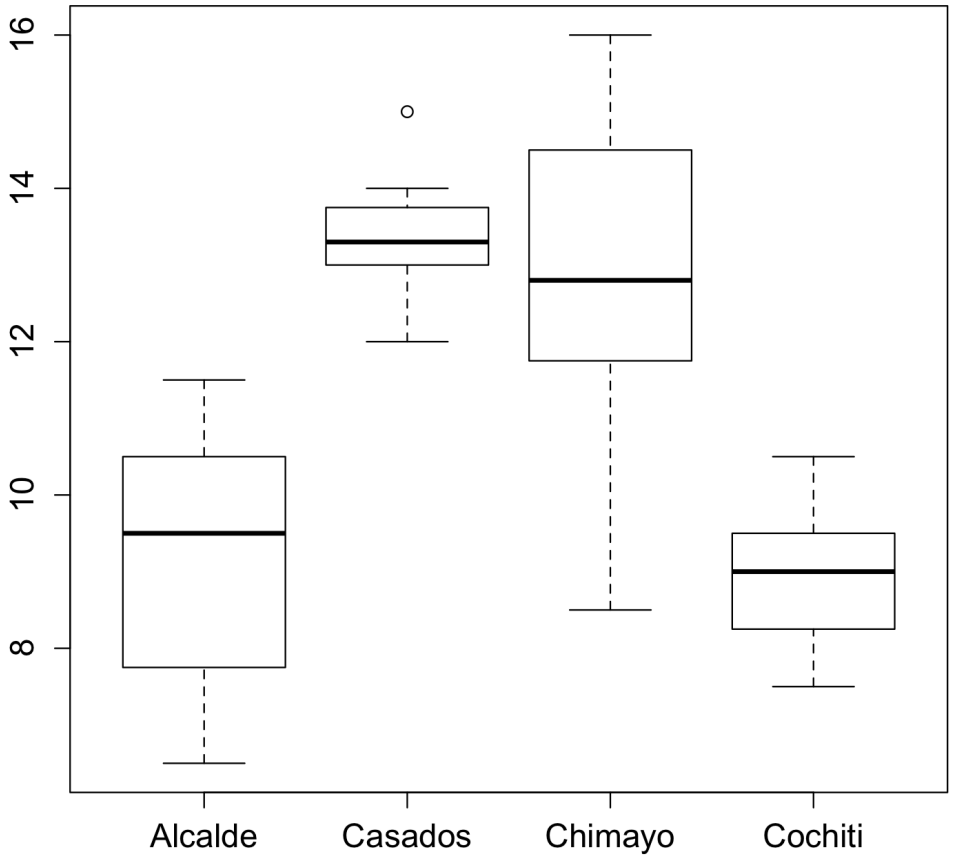
To use by() with combinations of variables, select a list of the columns used in the data set as follows: (here this is for columns 1 and 3):

```
> by(x$Length,x[,c(1,3)],mean)
group: Alcalde
Width: 2
[1] NA
------------------------------------------------------------
group: Casados
Width: 2
[1] NA
------------------------------------------------------------
group: Chimayo
Width: 2
[1] 12.5
------------------------------------------------------------
group: Cochiti
Width: 2
[1] 9
```

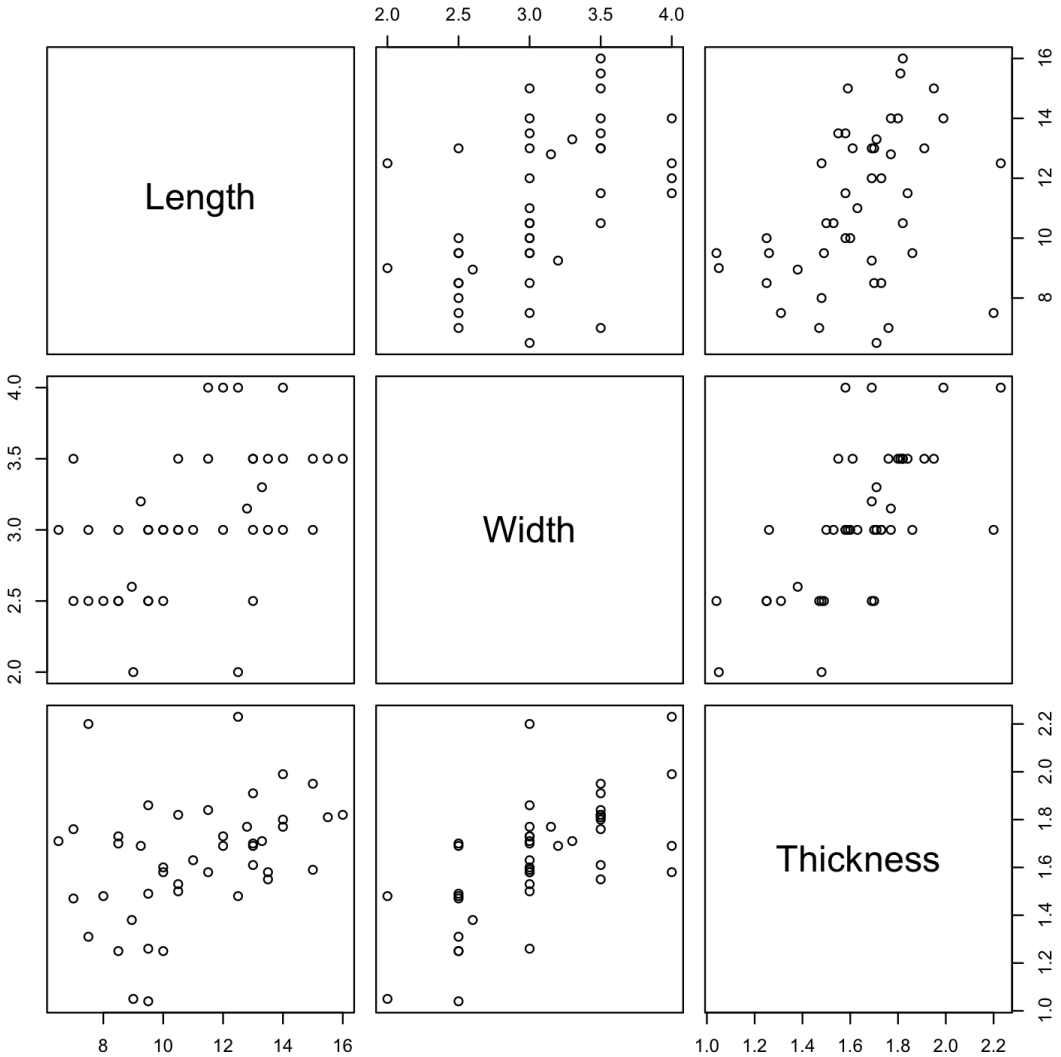
Here are some graphical ways to look at the data. Here `cex.axis=1.3` increases the label sizes by 30% to make them more readable. The code `pairs(x[,2:4])` means to make pairwise scatterplots for all rows, columns 2 through 4 of the data (so column 1 isn't included).

```
> boxplot(x$Length ~ x$group, cex.axis=1.3)
> pairs(x[,2:4])
> pairs(x[,-1]) # is equivalent
```

Boxplots



Boxplots


Length
Width
Thickness
2.0
2.5
3.0
3.5
4.0
8
10
12
14
16
1.0
1.2
1.4
1.6
1.8
2.0
2.2

R can also be used as a calculator, or to enter data directly by typing it in.

```
> a <- 2
> b <- c(4,5,7)
> a
[1] 2
> b
[1] 4 5 7
> a*b
[1] 8 10 14
> a+b
[1] 6 7 9
> b^2
[1] 16 25 49
> sqrt(b)
[1] 2.000000 2.236068 2.645751
> log(b)
[1] 1.386294 1.609438 1.945910
```

As a review of some of the statistical procedures we used last semester, here is a table

one sample t test	one group/population, quantitative data
matched pairs t test	one sample t-test on differences, quantitative data
two sample t-test	1 quantitative variable, 1 group variable (2 groups)
anova	quantitative response, 1 group variable with 2 or more
simple linear regression	1 predictor, 1 response, both quantiative
correlation test	2 quantitative variables that are paired, but not (e.g.,GPA and SAT score)
proportion test	binary variable (0 or 1), test whether frequency of 1s equal some proportion
2-sample proportion test	binary variable for two groups, test for equality of proportions
chi-square test (one way)	several categories, testing for equality of proportions or proportions matching theoretical values
chi-square test (two-way)	two table test of association, e.g., college major versus political affiliation

As a review of some of the statistical procedures we used last semester, here is a table

one sample t test	`t.test(x)`
matched pairs t test	`t.test(x-y)` or `diff <- x-y; t.test(diff)`
two sample t-test	`t.test(x,y)` or `t.test(x ~ group)`
anova	`a <- aov(x ~ group)` or `a <- lm(x ~ group)`
simple linear regression	`model <- lm(y ~ x)`
correlation test	`cor.test(x,y)`
proportion test	`prop.test(x,p=.5)`
2-sample proportion test	e.g., `prop.test(c(45,55),c(100,100))`
chi-square test (one way)	`chisq.test()`
chi-square test (two-way)	`chisq.test(M)` where `M` is a matrix

Many of the above procedures could be employed with the `chile.txt` data set. For example, we could

- use a t-test to see whether the length of the chile pods differs for Chimayo versus Cochiti
- use a t-test to see whether the width of the child pods differes for Chimayo versus Casados
- use ANOVA to test whether length is the same for all four types of chiles
- use regression to determine the relationship between length and width of child pods (either ignoring chile type or separately for each type)
- test the correlation of length and thickness
- Classify chiles as long verus short (say, greater than 11cm) and test whether the proportion of child pods that are long is greater than 50%. (proprtion test).
- test whehter the proportion of chile pods classified as long differs by location (this would be a one-way chi-squared)

In this exampe, a t-test is done for the Length of the Cochiti chiles. Note that the hypothesis test is testing whether the length is 0. The confidence interval is probably more useful.

```
> t.test(x$Length[x$group=="Cochiti"])

One Sample t-test

data:  x$Length[x$group == "Cochiti"]
t = 30.101, df = 10, p-value = 3.833e-11
alternative hypothesis: true mean is not equal to 0
95 percent confidence interval:
 8.287493 9.612507
sample estimates:
mean of x
     8.95
```

```
> t.test(x$Length[x$group=="Cochiti"],
x$Length[x$group=="Alcalde"])

Welch Two Sample t-test

data:  x$Length[x$group == "Cochiti"] and x$Length[x$group == "Al
t = -0.48637, df = 15.546, p-value = 0.6335
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -1.610688  1.010688
sample estimates:
mean of x mean of y
     8.95      9.25
```

```
> a <- aov(x$Length ~ x$group)
> summary(a)
             Df Sum Sq Mean Sq F value  Pr(>F)
x$group       3  173.5   57.83   22.45 1.1e-08 ***
Residuals    40  103.0    2.58
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

logic and for-loops in R

```
> sum(Long)
[1] 25
> length(Long)
[1] 44
> prop.test(25,44)

1-sample proportions test with continuity correction

data:  25 out of 44, null probability 0.5
X-squared = 0.56818, df = 1, p-value = 0.451
alternative hypothesis: true p is not equal to 0.5
95 percent confidence interval:
 0.4114174 0.7131820
sample estimates:
        p
0.5681818
```

logic and for-loops in R

Although you can do a lot of statistics in R without logic and for-loop programming in R, it would be a shame to take this course without learning a little about logic control and for-loops. These are extremely general concepts used in programming that is ubiquitous in our lives (Google, Facebook, cell phones, etc.).

R can evaluate statements as true or false. These true and false values can then be converted into 1s (TRUEs) and 0s (FALSEs).

```
> x$Length>10
 [1]  TRUE FALSE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE
[13]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
[25]  TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE FALSE
[37] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
> as.numeric(x$Length>10)
 [1] 1 0 1 1 1 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1
[39] 0 0 0 0 0 0
```

for loops

As another example, suppose you have Likert scale data, such as 1=strongly disagree, 2=disagree, 3=neutral, 4=agree, 5=stronngly agree. Now suppose you want to transform the data so that you collapse categories, and just want 1=disagree, 2=neutral, 3=agree.

One way of processing the data is with a for loop. For example

```
> data <- c(5,3,4,3,1,2,1,5,4,3)
> newdata <- 1:length(data)
> for(i in 1:length(newdata)) {
+ if(data[i] <= 2) newdata[i] <- 1
+ if(data[i] == 3) newdata[i] <- 2
+ if(data[i] >= 4) newdata[i] <- 3
+ }
> newdata
 [1] 3 2 3 2 1 1 1 3 3 2
```

for loops

Another way of achieving the same result is

```
> data <- c(5,3,4,3,1,2,1,5,4,3)
> newdata <- 1*(data <= 2) + 2*(data==3) + 3*(data >= 4)
> newdata
 [1] 3 2 3 2 1 1 1 3 3 2
```

logic and for-loops in R

For loops allow you to process a data set one row at a time or one column at a time, or to do a repetetive thing one at a time.
A simple example of a for loop is

```
> for(i in 1:10) {
+ print(i^2)
+ }
[1] 1
[1] 4
[1] 9
[1] 16
[1] 25
[1] 36
[1] 49
[1] 64
[1] 81
[1] 100
```

logic and for-loops in R

Often in R you can avoid for loops by processing a vector all at once, such as testing whether each element is above or below a certain threshold. But sometimes it is useful to do things one at a time. For loops can also be useful for simulation. Here we simulate the probability of getting four aces in a poker hand. Cards are numbered 1 through 52, and I assume cards 1 through 4 are aces.

```
> cards <- 1:52
> fourAces <- 1:100000
> for(i in 1:100000) {
+ hand <- sample(cards,5,replace=TRUE)
+ num_aces = sum(hand <= 4)
+ fourAces[i] <- (num_aces == 4)
+ }
> sum(fourAces)/100000
[1] 0.00016 # estimated probability is 0.016%
```