Psychology 350: An introduction to R for Psychological Research
Data entry and descriptives

William Revelle
Northwestern University
Evanston, Illinois USA

https://personality-project.org/courses/350

March, 2024

Basic R commands – remember don't enter the >

R is just a fancy calculator. Add, subtract, sum, products, group

```
> 2 + 2        #sum two numbers
[1] 4      #show the output
> 3^4     #3 raised to the 4th
[1] 81     #that was easy
> sum(1:10)  #find the sum of the first 10 numbers
[1] 55   #the answer
> prod(c(1, 2, 3, 5, 7)) #the product of the concatenated (c) numbers
[1] 210  #Note how we combined product with concatenate
```

It is also a statistics table (the normal distribution, the t, the F, the χ^2 distribution, the xyz distribution)

```
> pnorm(q = 1)  #the probability of a normal with value of 1 sd
[1] 0.8413447  #
> pt(q = 2, df = 20) #what about the probability of a t-test value of
[1] 0.9703672  #this is the upper tail
```

R is a set of distributions. Don't buy a stats book with tables!

Table: To obtain the density, prefix with *d*, probability with *p*, quantiles with *q* and to generate random values with *r*. (e.g., the normal distribution may be chosen by using dnorm, pnorm, qnorm, or rnorm.) Each function can be modified with various parameters.

Distribution	base name	P 1	P 2	P 3	example application
Normal	norm	mean	sigma		Most data
Multivariate normal	mvnorm	mean	r	sigma	Most data
Log Normal	lnorm	log mean	log sigma		income or reaction time
Uniform	unif	min	max		rectangular distributions
Binomial	binom	size	prob		Bernuilli trials (e.g. coin flips)
Student's t	t	df		nc	Finding significance of a t-test
Multivariate t	mvt	df	corr	nc	Multivariate applications
Fisher's F	f	df1	df2	nc	Testing for significance of F test
χ^2	chisq	df		nc	Testing for significance of χ^2
Exponential	exp	rate			Exponential decay
Gamma	gamma	shape	rate	scale	distribution theoryh
Hypergeometric	hyper	m	n	k	
Logistic	logis	location	scale		Item Response Theory
Poisson	pois	lambda			Count data
Weibull	weibull	shape	scale		Reaction time distributions

NORTHWESTERN UNIVERSITY

An example of using functions

R is vector oriented => what works for a single variable, works for vectors

R code

```
set.seed(42)     #to make for a reproducible example
x <- rnorm(1)   #find a random number
x       #show it
round(x,2)   #show it to 2 decimal places
#do it again
x <- rnorm(1)   #find a random number
round(x,2)
#and again
 x <- rnorm(1)   #find a random number
round(x,2)
set.seed(42)    #reset the seed to start the same sequence
x <- rnorm(10)   #create 10 random numbers
round(x,2)      #show them all
```

```
set.seed(42)     #to make for a reproducible example
>  x <- rnorm(1)   #find a random number
>  x       #show it
[1] 1.370958
>  round(x,2)   #show it to 2 decimal places
[1] 1.37
>  #do it again
>  x <- rnorm(1)   #find a random number
>  round(x,2)
```

Now, show the output

```
>  set.seed(42)    #to make for a reproducible example
>  x <- rnorm(1)  #find a random number
>  x      #show it
[1] 1.370958
>  round(x,2)  #show it to 2 decimal places
[1] 1.37
>  #do it again
>  x <- rnorm(1)  #find a random number
>  round(x,2)
[1] -0.56
>  #and again
>   x <- rnorm(1)  #find a random number
>  round(x,2)
[1] 0.36
>  set.seed(42)   #reset the seed to start the same sequence
>  x <- rnorm(10)  #create 10 random numbers
>  round(x,2)     #show them all
 [1]  1.37 -0.56  0.36  0.63  0.40 -0.11  1.51 -0.09  2.02 -0.06
```

An example of using r, p, and q for the normal distribution

R code

```
set.seed(42) #set the random seed to get the same sequence
x <- rnorm(5) #find 5 randomly distributed normals
round(x,2) #show them, rounded to 2 decimals
round(pnorm(x),2) #show their probabilities to 2 decimals
round(qnorm(pnorm(x)),2)  #find the quantiles of the normal
```

Produces this output

```
> set.seed(42) #set the random seed to get the same sequence
> x <- rnorm(5) #find 5 randomly distributed normals
> round(x,2) #show them, rounded to 2 decimals
[1]  1.37 -0.56  0.36  0.63  0.40
> round(pnorm(x),2) #show their probabilities to 2 decimals
[1] 0.91 0.29 0.64 0.74 0.66
> round(qnorm(pnorm(x)),2)  #find the quantiles of the normal
[1]  1.37 -0.56  0.36  0.63  0.40
#make up a probability table of the normal distribution from -3 to
 round(pnorm(seq(-3,3,.5)),2)
 [1] 0.00 0.01 0.02 0.07 0.16 0.31 0.50 0.69 0.84 0.93 0.98 0.99 1
```

Make up a table of probability values

We will create a `data.frame` which is just a table of rows and columns

R code

```
z <- seq(-3,3,.25)
p.df <- data.frame(normal = z, probability = round(pnorm(z),2))
p.df
```

```
    normal probability
1    -3.00        0.00
2    -2.75        0.00
3    -2.50        0.01
4    -2.25        0.01
5    -2.00        0.02
6    -1.75        0.04
7    -1.50        0.07
8    -1.25        0.11
9    -1.00        0.16
10   -0.75        0.23
11   -0.50        0.31
12   -0.25        0.40
13    0.00        0.50
14    0.25        0.60
15    0.50        0.69
16    0.75        0.77
17    1.00        0.84
18    1.25        0.89
19    1.50        0.93
20    1.75        0.96
```

Add the normal densities to this table

Note that we are rounding the numbers to be more readable.

R code

```
 p.df <- data.frame(p.df,density=round(dnorm(z),2))
p.df
```

```
    normal probability density
1    -3.00        0.00    0.00
2    -2.75        0.00    0.01
3    -2.50        0.01    0.02
4    -2.25        0.01    0.03
5    -2.00        0.02    0.05
6    -1.75        0.04    0.09
7    -1.50        0.07    0.13
8    -1.25        0.11    0.18
9    -1.00        0.16    0.24
10   -0.75        0.23    0.30
11   -0.50        0.31    0.35
12   -0.25        0.40    0.39
13    0.00        0.50    0.40
14    0.25        0.60    0.39
15    0.50        0.69    0.35
16    0.75        0.77    0.30
17    1.00        0.84    0.24
18    1.25        0.89    0.18
19    1.50        0.93    0.13
20    1.75        0.96    0.09
21    2.00        0.98    0.05
22    2.25        0.99    0.03
23    2.50        0.99    0.02
24    2.75        1.00    0.01
25    3.00        1.00    0.00
```

What if we don't round?

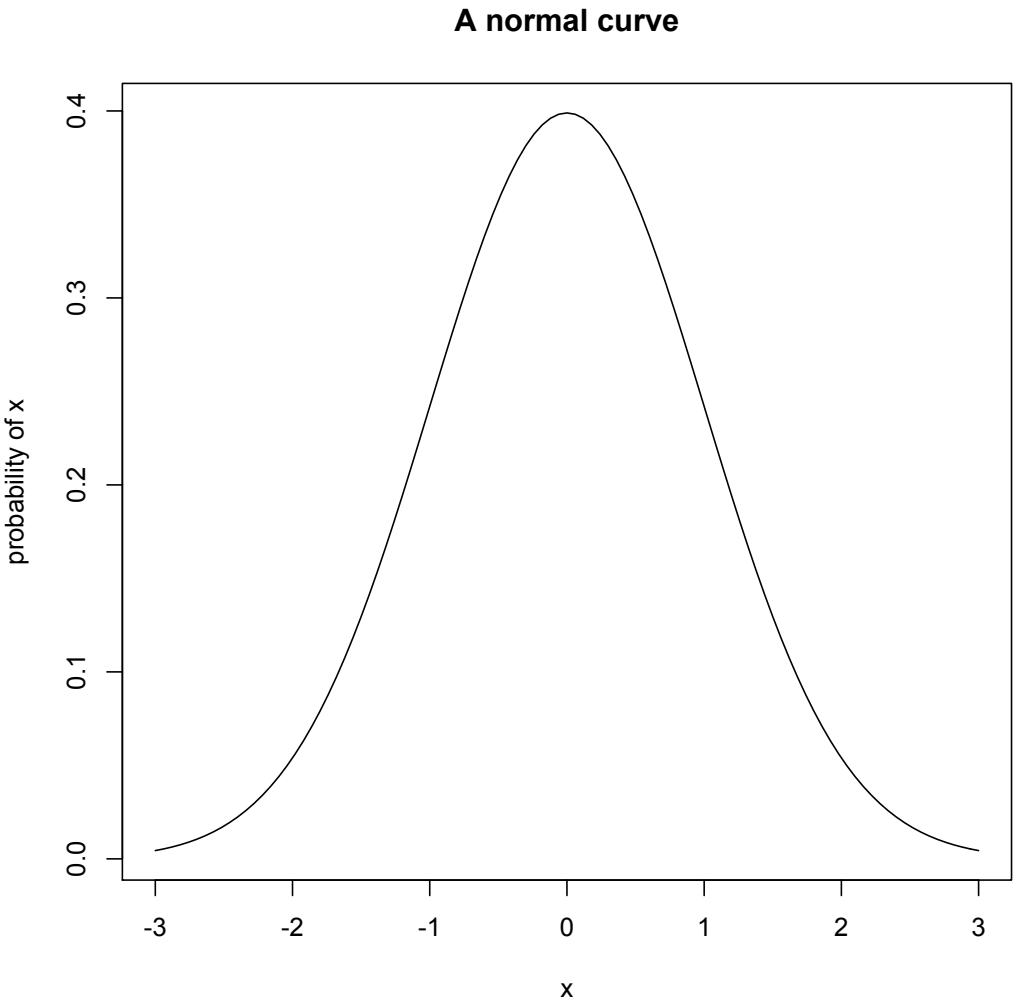
R code

```
p.df <- data.frame(normal=z, probability = pnorm(z),density=dnorm(z))
```

```
 p.df <- data.frame(normal=z, probability = pnorm(z),density=dnorm(z))
> p.df
   normal probability     density
1   -3.00 0.001349898 0.004431848
2   -2.75 0.002979763 0.009093563
3   -2.50 0.006209665 0.017528300
4   -2.25 0.012224473 0.031739652
5   -2.00 0.022750132 0.053990967
6   -1.75 0.040059157 0.086277319
7   -1.50 0.066807201 0.129517596
8   -1.25 0.105649774 0.182649085
9   -1.00 0.158655254 0.241970725
10  -0.75 0.226627352 0.301137432
11  -0.50 0.308537539 0.352065327
12  -0.25 0.401293674 0.386668117
13   0.00 0.500000000 0.398942280
14   0.25 0.598706326 0.386668117
15   0.50 0.691462461 0.352065327
16   0.75 0.773372648 0.301137432
17   1.00 0.841344746 0.241970725
18   1.25 0.894350226 0.182649085
19   1.50 0.933192799 0.129517596
20   1.75 0.959940843 0.086277319
21   2.00 0.977249868 0.053990967
22   2.25 0.987775527 0.031739652
23   2.50 0.993790335 0.017528300
24   2.75 0.997020237 0.009093563
25   3.00 0.998650102 0.004431848
```

```
round(p.df,digits=2)
   normal probability density
1   -3.00        0.00    0.00
2   -2.75        0.00    0.01
3   -2.50        0.01    0.02
4   -2.25        0.01    0.03
5   -2.00        0.02    0.05
6   -1.75        0.04    0.09
7   -1.50        0.07    0.13
8   -1.25        0.11    0.18
9   -1.00        0.16    0.24
10  -0.75        0.23    0.30
11  -0.50        0.31    0.35
12  -0.25        0.40    0.39
13   0.00        0.50    0.40
14   0.25        0.60    0.39
15   0.50        0.69    0.35
16   0.75        0.77    0.30
17   1.00        0.84    0.24
18   1.25        0.89    0.18
19   1.50        0.93    0.13
20   1.75        0.96    0.09
21   2.00        0.98    0.05
22   2.25        0.99    0.03
23   2.50        0.99    0.02
24   2.75        1.00    0.01
25   3.00        1.00    0.00
```

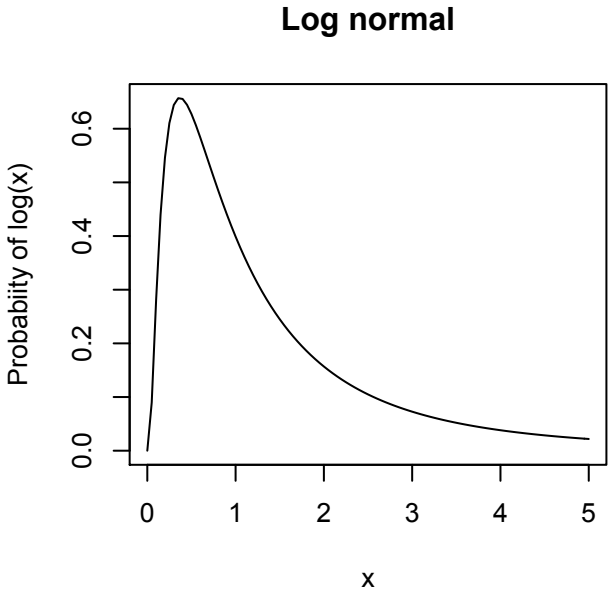
R can draw distributions

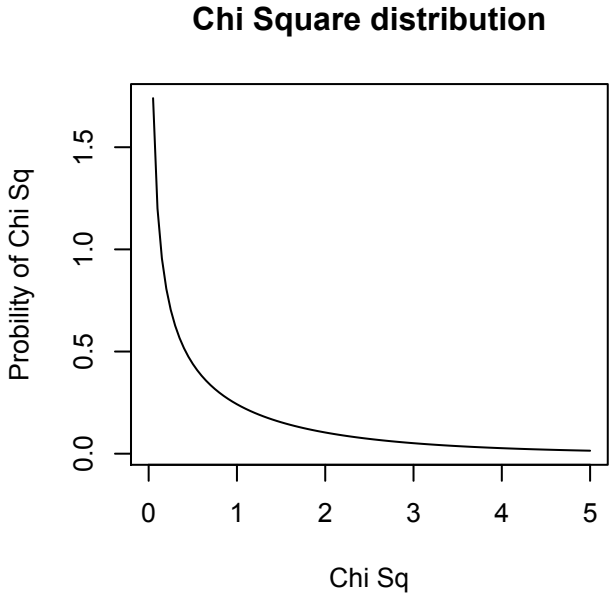


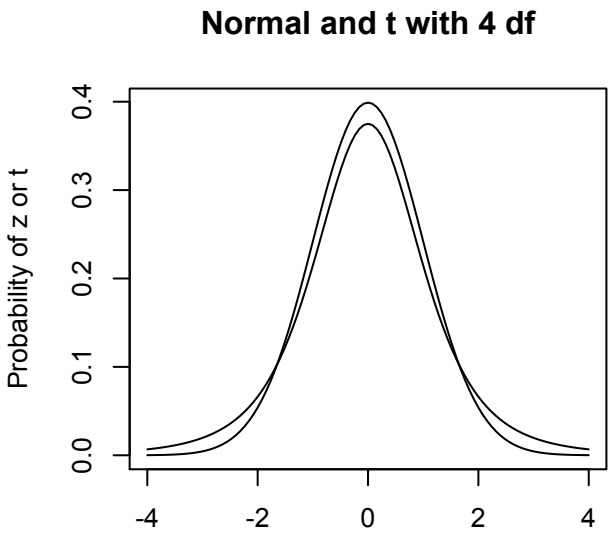
We do this by using the `curve` function to which we pass the values of the `dnorm` function.

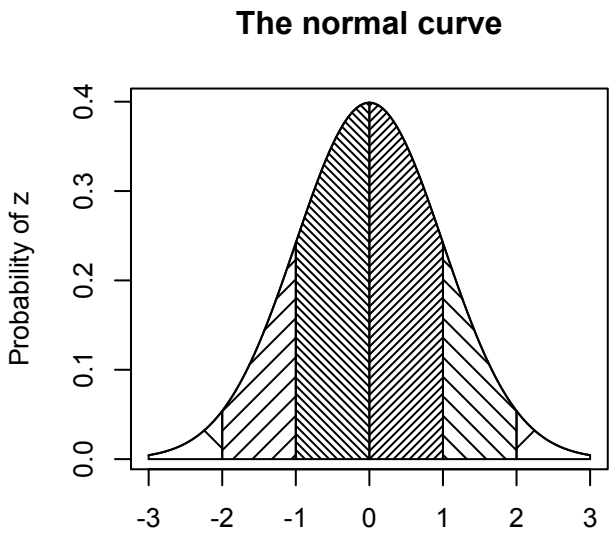
```
curve(dnorm(x),-3,3,
ylab="probability of
x",main="A normal
curve")
```

R can draw more interesting distributions









R is also a graphics calculator

R code

```
op <- par(mfrow=c(2,2))        #set up a 2 x 2 graph
curve(dlnorm(x),0,5,ylab='Probabiity of log(x)',main='Log normal')
curve(dchisq(x,1),0,5,ylab='Probility of Chi Sq',xlab='Chi Sq',main='Chi Square distribution
curve(dnorm(x),-4,4,ylab='Probability of z or t',xlab='z or t',main='Normal and t with 4 df')
curve(dt(x,4),add=TRUE)
#
#somewhat more complicated
#first draw the normal curve
curve(dnorm(x),-3,3,xlab="",ylab="Probability of z") #the range of x
title(main="The normal curve",outer=FALSE) #the title
#add the cross hatching by using polygons
xvals <-  seq(-3,-2,length=100)  #From -3 to 2 with 100 points
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=-45)
xvals <-  seq(-2,-1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=45)
xvals <-  seq(-1,-0,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=-45)
xvals <-  seq(2,3,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=2,angle=45)
xvals <-  seq(1,2,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=14,angle=-45)
xvals <-  seq(0,1,length=100)
dvals <- dnorm(xvals)
polygon(c(xvals,rev(xvals)),c(rep(0,100),rev(dvals)),density=34,angle=45)
op <- par(mfrow=c(1,1)) #back to a normal 1 x 1 graph
```

R can help teach with 100s of example data sets.

R code

```
> data()

> data(package="psych")
#see the names of the
       35 data sets
> data(package="psychTools")
#see the 42 here

> data(Titanic)
> ? Titanic


> data(cushny)
> ? cushny

> data(UCBAdmissions)
> ? UCBAdmissions
```

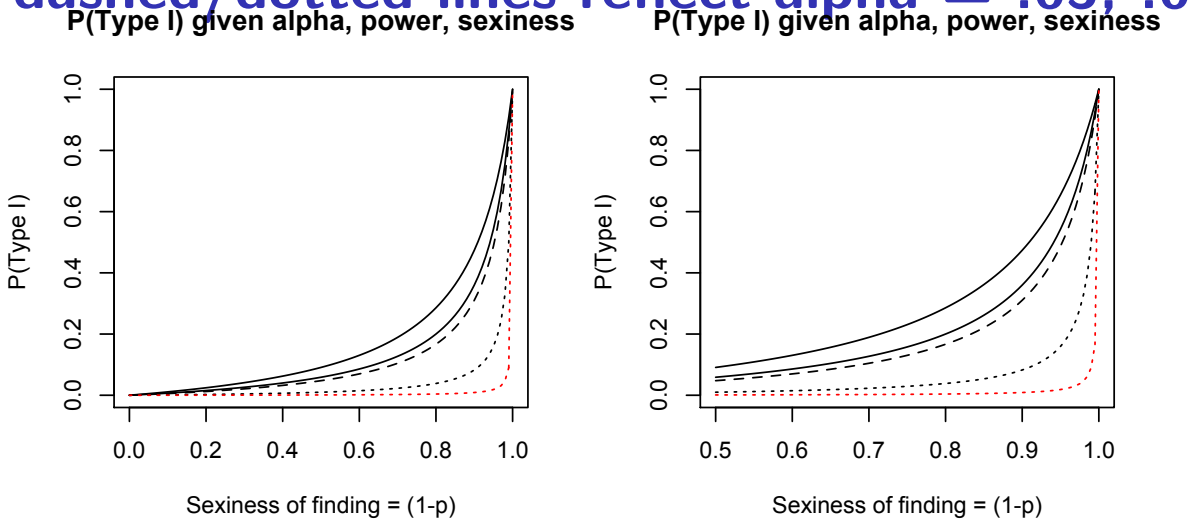
1. This opens up a separate text window and lists all of the data sets in the currently loaded packages.
2. Show the data sets available in a particular package (e.g., *psych*).
3. And even more data sets in *psychTools*).
4. Gets the particular data set with its help file (e.g., the survival rates on the Titanic cross classified by age, gender and class).
5. Another original data set used by "student" (Gossett) for the t-test.
6. The UC Berkeley example of "sex discrimination" as a Simpson

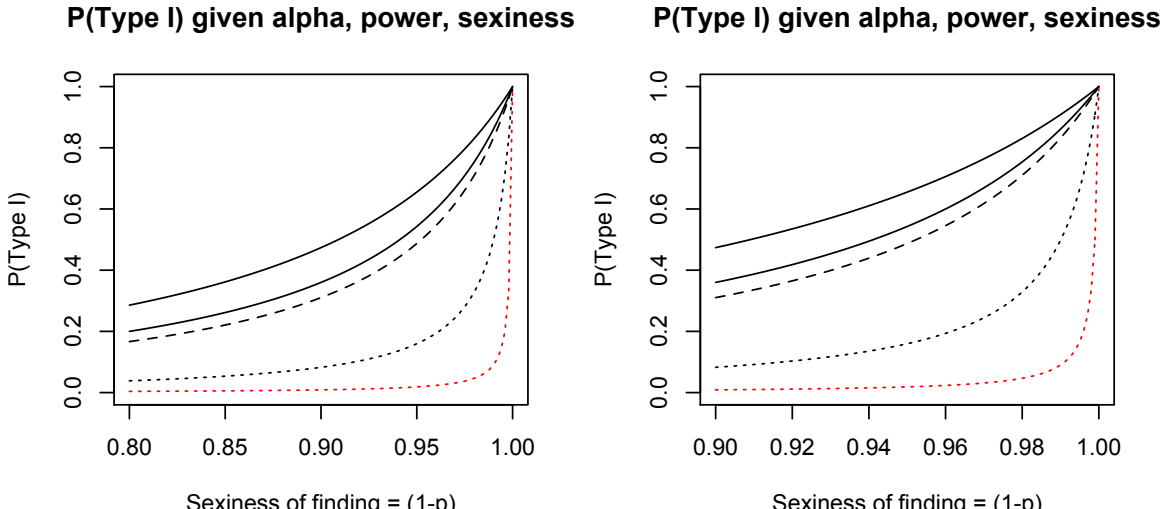
some of the psychTools data sets: data(package="psychTools")

epi	Eysenck Personality Inventory (EPI) data for 3570 partici
epi.bfi	13 personality scales from the Eysenck Personality Invent inventory
epi.dictionary	Eysenck Personality Inventory (EPI) data for 3570 partici
epi.keys (epiR)	Eysenck Personality Inventory (EPI) data for 3570 partici
epiR	Eysenck Personality Inventory (EPI) data for 3570 partici
galton	Galton's Mid parent child height data
heights	A data.frame of the Galton (1888) height and cubit data s
holzinger.dictionary	The raw and transformed data from Holzinger and Swineford
holzinger.raw	The raw and transformed data from Holzinger and Swineford
holzinger.swineford	The raw and transformed data from Holzinger and Swineford
income	US family income from US census 2008
iqitems	16 multiple choice IQ items
msq	75 mood items from the Motivational State Questionnaire f participants
msqR	75 mood items from the Motivational State Questionnaire f participants
neo	NEO correlation matrix from the NEO_PI_R manual
peas	Galton's Peas
sai	State Anxiety data from the PMC lab over multiple occasio
sai.dictionary	State Anxiety data from the PMC lab over multiple occasio
spi	A sample from the SAPA Personality Inventory including an dictionary and scoring keys.
spi.dictionary	A sample from the SAPA Personality Inventory including an dictionary and scoring keys.
spi.keys	A sample from the SAPA Personality Inventory including an dictionary and scoring keys.
tai	State Anxiety data from the PMC lab over multiple occasio
veg (vegetables)	Paired comparison of preferences for 9 vegetables

NORTHWESTERN UNIVERSITY

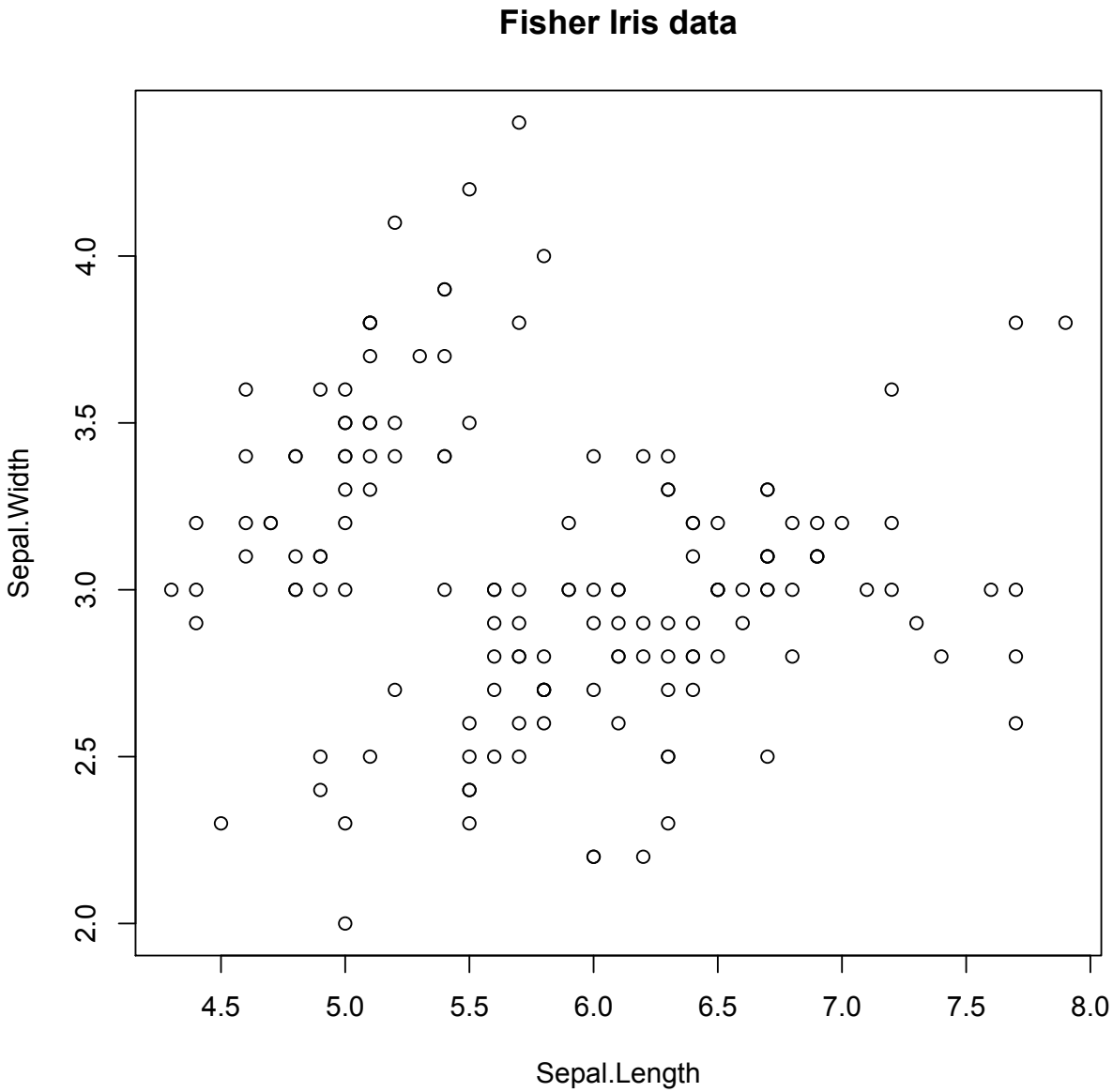
R can show current statistical concepts:
Type I Errors: It is not the power, it is the prior likelihood
dashed/dotted lines reflect alpha = .05, .01, .001 with power = 1





1. Extreme claims require extreme probabilities
2. Given that a finding is "significant", what is the likelihood that it is a Type I error?
3. Depends upon the prior likelihood (the 'sexiness') of the claim.

A simple scatter plot using `plot` with Fisher's Iris data set.



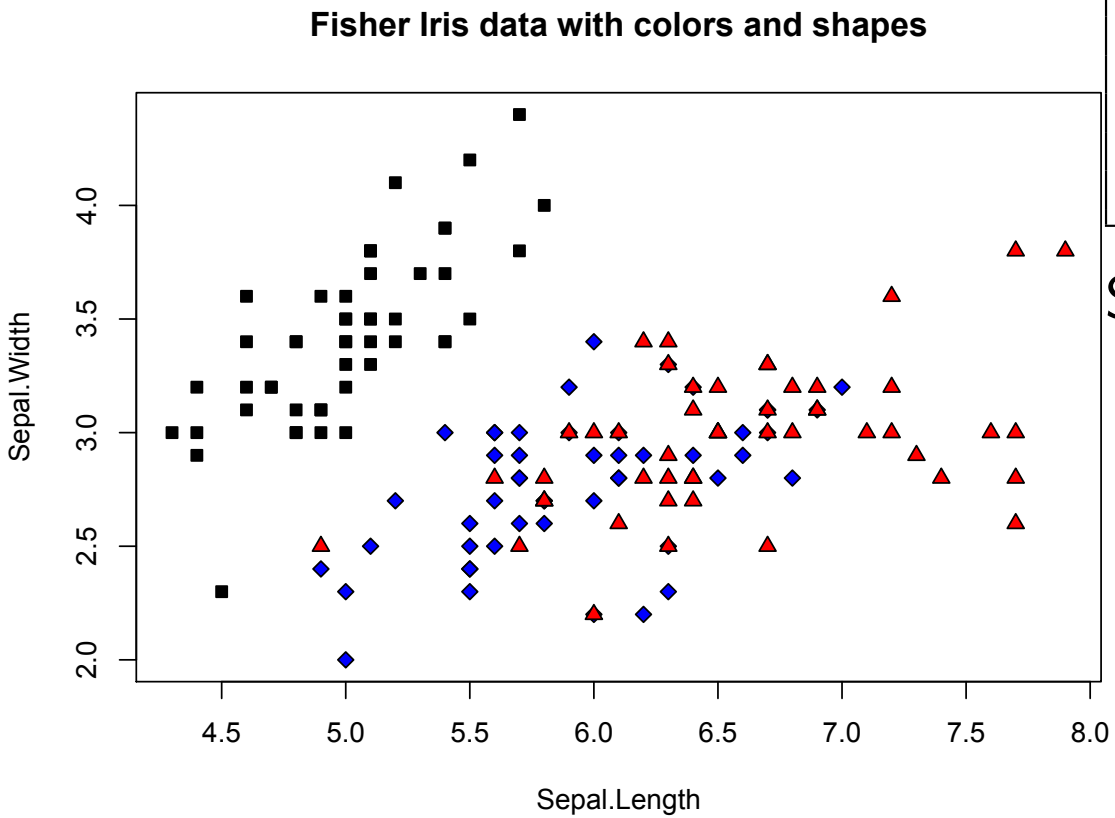
R code

```
plot(iris[1:2],xlab="Sepal.Length",
  ylab="Sepal.Width"
,main="Fisher Iris data")}
```

Set parameters

1. xlab for x axis label
2. ylab for y axis label
3. main for title

A simple scatter plot using `plot` with some colors and shapes



R code

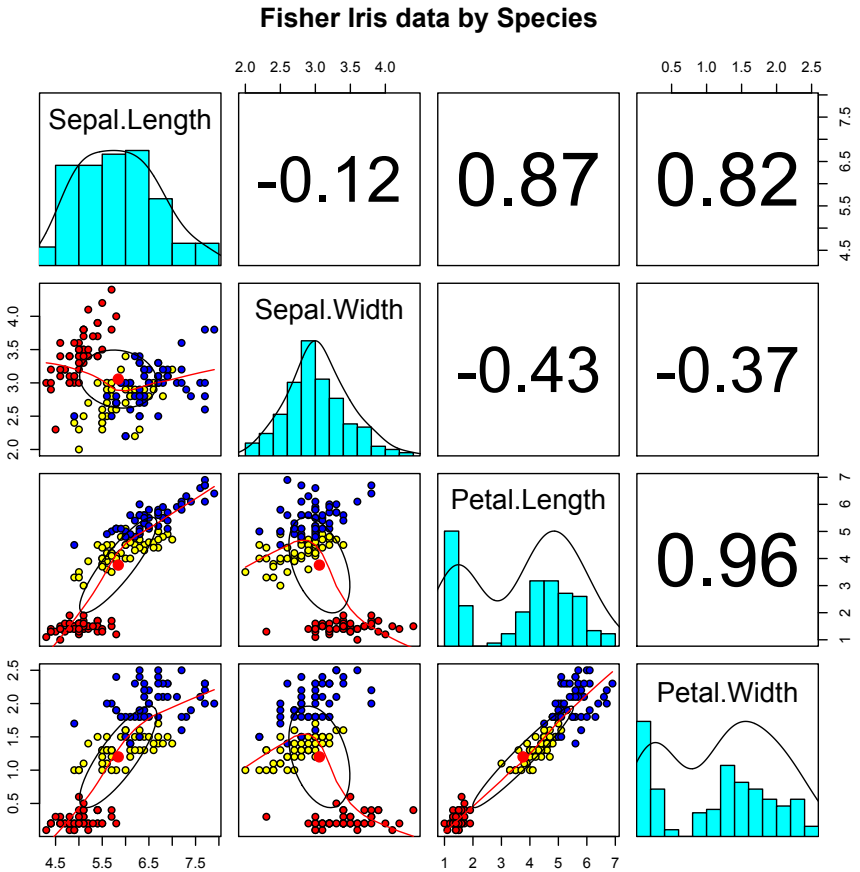
```
plot(iris[1:2],xlab="Sepal.Length",
ylab="Sepal.Width" ,
main="Fisher Iris data with
colors and shapes",
 bg=c("black","blue",
"red")[iris[,"Species"]],
 pch=21+ as.numeric(iris[,5]))
```

Set parameters

1. bg for background colors of symbols
2. pch chooses the plot character
3. Note how these depend upon iris[,5] which is the species

Show the various graphic options for plot character (pch)

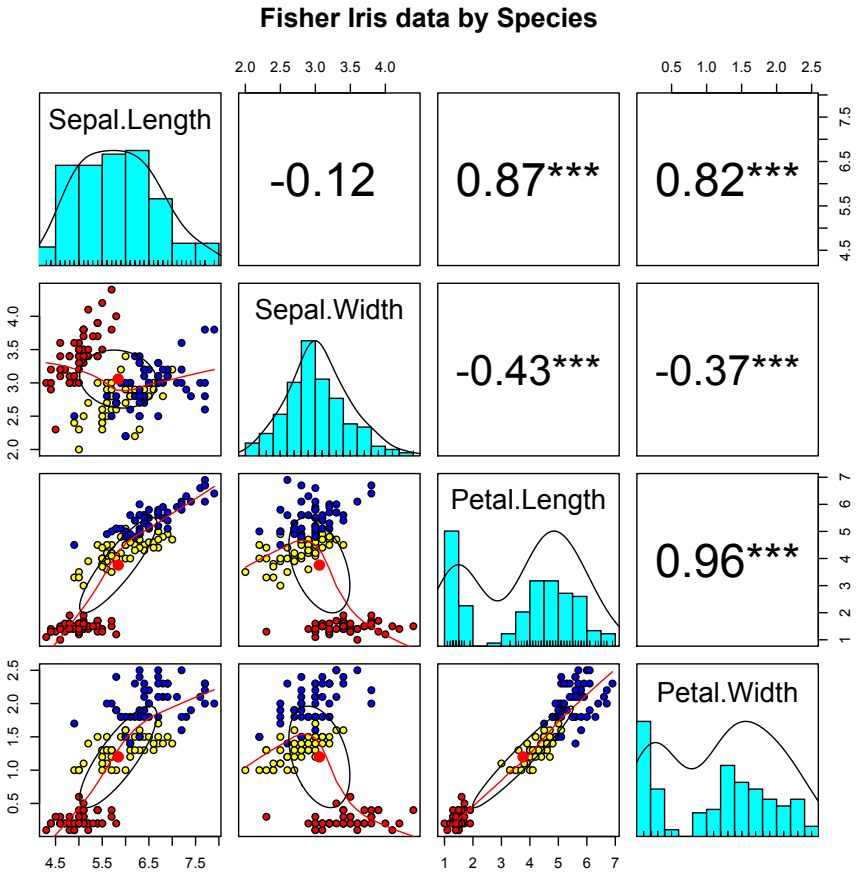
A scatter plot matrix plot with loess regressions using `pairs.panels`



1. Correlations above the diagonal
2. Diagonal shows histograms and densities
3. scatter plots below the diagonal with correlation ellipse
4. locally smoothed (loess) regressions for each pair
5. optional color coding of grouping variables.

```
pairs.panels(iris[1:4],bg=c("red","yellow","blue")
[iris$Species],pch=21,main="Fisher Iris data by
Species")
```

A scatter plot matrix plot with loess regressions using `pairs.panels`



Show "significance" using magic asterisks

```
pairs.panels(iris[1:4],bg=c("red","yellow","blue")
[iris$Species],pch=21,main="Fisher Iris data by
Species",stars=TRUE)
```

Many ways to get data

1. Simulate them (but be sure to call it simulation)
2. Copy in from the clipboard after copying from another program: `read.clipboard`
3. This requires the *psychTools*
4. Reading a local file `read.file` which just calls two functions
   - 4.1 Finding the file using `file.choose`
   - 4.2 Reading the file using `read.table` or `read.spss` or ...
5. Reading a remote file `read.file`

Reading from the clipboard

Get the file using your browser: eg.
http://personality-project.org/r/datasets/simulation.txt
copy to the clipboard

R code

```
my.data <- read.clipboard()  #this takes what is in the clipboard and
          makes into the my.data object
dim(my.data)  #to see how many rows and columns
headTail(my.data) # show the first and last n lines
describe(my.data) #basic descriptives
```

```
 my.data <- read.clipboard()
> dim(my.data)
[1] 72  7
> headTail(my.data)
    Time Anxiety Impulsivity sex Arousal Tension Performance
1      9       4           9   1      50      55          40
2     19       8           8   1      70      64          90
3      9       5          10   2      50      69          48
4      9       4           1   2      57      55          68
...  ...     ...         ... ...     ...     ...         ...
69    19       6           1   1      66      53          88
70     9       5          10   2      48      63          40
71    19       6           8   2      69      60          95
72    19      10           1   2      66      48          93
> describe(my.data)
            vars  n  mean    sd median trimmed   mad min max range  skew kurtosis   se
Time           1 72 14.28  5.03   19.0   14.34  0.00   9  19    10 -0.11    -2.02 0.59
Anxiety        2 72  5.24  2.18    5.0    5.24  2.97   0  10    10 -0.04    -0.65 0.26
```

Reading from a local file

R code

```
my.data <- read.file() #a window opens and you find the file
dim(my.data)  #to see how many rows and columns
#headTail(my.data) # show the first and last n lines
describe(my.data) #basic descriptives
```

```
 my.data <- read.file()
Data from the .txt file /Users/WR/Box Sync/pmc_folder/314/datasets/simulation.txt
               has been loaded.
> dim(my.data)
[1] 72  7
> describe(my.data)
            vars  n  mean    sd median trimmed   mad min max range  skew kurtosis   se
Time           1 72 14.28  5.03   19.0   14.34  0.00   9  19    10 -0.11    -2.02 0.59
Anxiety        2 72  5.24  2.18    5.0    5.24  2.97   0  10    10 -0.04    -0.65 0.26
Impulsivity    3 72  4.90  3.98    4.5    4.88  5.19   0  10    10  0.02    -1.83 0.47
sex            4 72  1.50  0.50    1.5    1.50  0.74   1   2     1  0.00    -2.03 0.06
Arousal        5 72 60.90  8.10   66.0   61.29  5.93  48  70    22 -0.27    -1.67 0.96
Tension        6 72 56.83  6.29   57.0   57.14  5.93  38  69    31 -0.53     0.42 0.74
Performace     7 72 72.21 17.41   78.0   73.19 18.53  38  98    60 -0.43    -1.10 2.05
```

Read from a remote file

R code

```
file.name <- "http://personality-project.org/r/datasets/simulation.txt"
my.data <- read.file(file.name)  #goes to the remote location and reads it
```

The `read.file` looks at the suffix of the file to figure out what to do.

```
> file.name <- "http://personality-project.org/r/datasets/simulation.txt"
> my.data <- read.file(file.name)  #goes to the remote location and reads it
Data from the .txt file http://personality-project.org/r/datasets/simulation.txt has been
> describe(my.data)
             vars  n  mean    sd median trimmed   mad min max range  skew kurtosis   se
Time            1 72 14.28  5.03   19.0   14.34  0.00   9  19    10 -0.11    -2.02 0.59
Anxiety         2 72  5.24  2.18    5.0    5.24  2.97   0  10    10 -0.04    -0.65 0.26
Impulsivity     3 72  4.90  3.98    4.5    4.88  5.19   0  10    10  0.02    -1.83 0.47
sex             4 72  1.50  0.50    1.5    1.50  0.74   1   2     1  0.00    -2.03 0.06
Arousal         5 72 60.90  8.10   66.0   61.29  5.93  48  70    22 -0.27    -1.67 0.96
Tension         6 72 56.83  6.29   57.0   57.14  5.93  38  69    31 -0.53     0.42 0.74
Performace      7 72 72.21 17.41   78.0   73.19 18.53  38  98    60 -0.43    -1.10 2.05
>
```

Two ways of reading SPSS remote files

SPSS.sav files actually store the raw coding information. Generally we want to convert these to numeric values. But if we want to see what is there, we want to not convert.

R code

```
fn <- "http://personality-project.org/r/datasets/Cars.sav"
data1 <- read.file(fn)  #go and get it and convert to a normal data.frame
data2 <- read.file(fn,use.value.labels=TRUE) #don't convert the value labels
```

```
fn <- "http://personality-project.org/r/datasets/Cars.sav"
> data1 <- read.file(fn)  #go and get it and convert to a normal data.frame
Data from the SPSS sav file http://personality-project.org/r/datasets/Cars.sav has been lo
> data2 <- read.file(fn,use.value.labels=TRUE) #don't convert the value labels
Data from the SPSS sav file http://personality-project.org/r/datasets/Cars.sav has been lo
 head(data1)
  MPG ENGINE HORSE WEIGHT ACCEL YEAR ORIGIN CYLINDER FILTER_.
1  18    307   130   3504  12.0   70      1        8        0
2  15    350   165   3693  11.5   70      1        8        0
3  18    318   150   3436  11.0   70      1        8    \    0
4  16    304   150   3433  12.0   70      1        8        0
5  17    302   140   3449  10.5   70      1        8        0
6  15    429   198   4341  10.0   70      1        8        0
> head(data2)
  MPG ENGINE HORSE WEIGHT ACCEL YEAR   ORIGIN    CYLINDER     FILTER_.
1  18    307   130   3504  12.0   70 American 8 Cylinders Not Selected
2  15    350   165   3693  11.5   70 American 8 Cylinders Not Selected
3  18    318   150   3436  11.0   70 American 8 Cylinders Not Selected
4  16    304   150   3433  12.0   70 American 8 Cylinders Not Selected
5  17    302   140   3449  10.5   70 American 8 Cylinders Not Selected
6  15    429   198   4341  10.0   70 American 8 Cylinders Not Selected
```

R: a collection of functions

1. A function takes an input and returns an output
   - Can be thought of as y = f(x)
   - Functions operate on something (x) by doing something (typically associated with the name of the function) and then return a value
   - x can be a scaler, a vector, a matrix, a list, ...
   - y can be a scaler, a vector, a matrix, a list ...
   - Almost all functions include their parameters with ()
   - Every function has a help page associated with it (? the function name)
2. A short list of functions (by Tom Short)

Thinking about functions

The short list of functions (by Tom Short) includes most important Core R functions

1. Getting help
2. Input/Output
3. Data creation
4. Slicing and Dicing
5. Variable conversion
6. Variable information
7. ...

Some very simple functions (from psych)

Two ways of describing effect sizes, Cohen's d (standardized differences of means) and r (Pearson correlation coefficient). How do we convert from on to the other?

$$d = \frac{2r}{\sqrt{1 - r^2}} \qquad r = \frac{d}{d^2 + 4}$$

Show these

R code

```
r2d
d2r
```

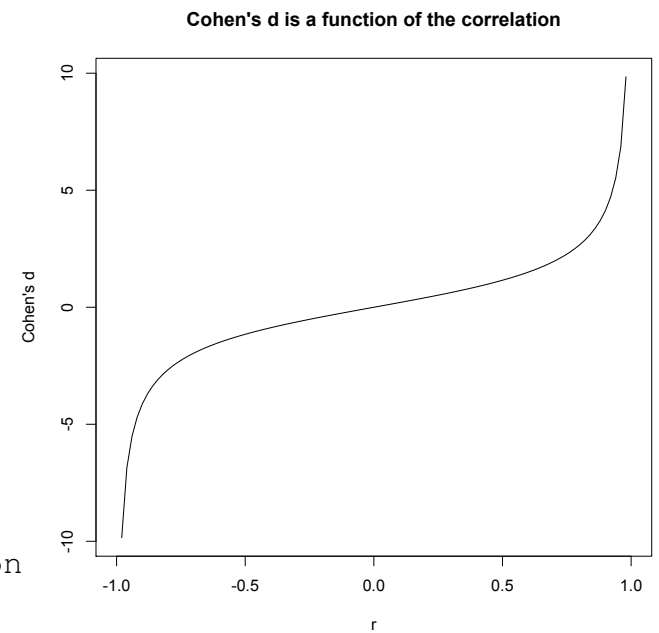
```
r2d
function (rho)
{
    2 * rho/sqrt(1 - rho^2)
}

d2r
function (d)
{
    d/sqrt(d^2 + 4)
}
```

Lets use these two functions

R code

```
r <- seq(from =0, to= 1, by =.1)
df.r2d <- data.frame(r,d= r2d(r))
curve(r2d(x),-1,1,
    main="Cohen's d is a function of the correlation",
      xlab = "r", ylab="Cohen's d")
```

```
df.r2d <- data.frame(r,d= r2d(r))
> round(df.r2d,2)
     r r2d.r.
1  0.0   0.00
2  0.1   0.20
3  0.2   0.41
4  0.3   0.63
5  0.4   0.87
6  0.5   1.15
7  0.6   1.50
8  0.7   1.96
9  0.8   2.67
10 0.9   4.13
11 1.0    Inf
> curve(r2d(x),-1,1, main="Cohen's d is a function
 of the correlation",
 xlab = "r", ylab="Cohen's d")
```



To explore a function, type its name `read.file`

1. The first line specifies the parameters that are passed to the function
2. We take advantages of several other functions
3. in particular, the following functions

if	A logical function tests truth of statement (if(A) { then do something} else {do something else}
missing	returns TRUE if the object is missing
file_ext	examines the suffix of an object (e.g., a file name)
%in%	returns a logical vector (TRUE or FALSE) if elements in A are found in object B
switch	Go to the part of code matching the object in switch

Annotated function: readFile

```
 read.file
function (file = NULL, header = TRUE,
     use.value.labels = FALSE,
    to.data.frame = TRUE, sep = ",",
    quote = "\"", widths = NULL,
    f = NULL, filetype = NULL, ...)
{
    if (missing(f) && missing(file))
        f <- file.choose()
    if (missing(f) && !missing(file))
        f <- file
    #calls a function to find the file extension
    suffix <- file_ext(f)
   #but we can override this if we want
    if (!missing(filetype))
        suffix <- filetype
    if (suffix %in% c("txt", "TXT", "text",
         "data", "DAT"))
        suffix <- "txt"
          if (suffix %in% c("R", "r"))
        suffix <- "R"
    if (suffix %in% c("rds", "RDS", "Rds"))
        suffix <- "rds"
```

1. The first line specifies the parameters that are passed to the function.
2. Notice the use of { to start and } end the function
3. We take advantages of several other functions
4. in particular, the fuctions
   - if,
   - missing,
   - file_ext,
   - %in%
   - switch

To see how a function works, just type its name

R code

```
 read.file
function (file = NULL, header = TRUE, use.value.labels = FALSE,
    to.data.frame = TRUE, sep = ",", quote = "\"", widths = NULL,
    f = NULL, filetype = NULL, ...)
{
    if (missing(f) && missing(file))
        f <- file.choose()
    if (missing(f) && !missing(file))
        f <- file
    suffix <- file_ext(f)     #calls a function to find the file extension
    if (!missing(filetype))    #but we can override this if we want
        suffix <- filetype
    if (suffix %in% c("txt", "TXT", "text", "data", "DAT"))
        suffix <- "txt"
    if (suffix %in% c("R", "r"))
        suffix <- "R"
    if (suffix %in% c("rds", "RDS", "Rds"))
        suffix <- "rds"
    if (suffix %in% c("Rda", "rda", "Rdata", "rdata"))
        suffix <- "Rda"
    if (suffix %in% c("SYD", "syd", "sys", "SYS"))
        suffix <- "SYD"
    if (!missing(widths)) {
        result <- read.fwf(f, widths, ...)
        message("The fixed width file ", f, "has been loaded.")
    }
    else {
        switch(suffix, sav = {
            result <- read.spss(f, use.value.labels = use.value.labels,
                to.data.frame = to.data.frame)
            message("Data from the SPSS sav file ", f, " has been loaded.")
```

read.file (continued)

R code

```
}, txt = {
          result <- read.table(f, header = header, ...)
          message("Data from the .txt file ", f, " has been loaded.")
      }, rds = {
          result <- try(readRDS(f, ...), silent = TRUE)
          if (class(result) == "try-error") {
              result <- f
              message("I had problems reading this file, \ntry  load('",
                f, "') instead. \nCaution, this might replace an object currently in your e
          } else {
              message("File ", f, " has been loaded.")
          }
      }, R = {
          result <- dget(f, ...)
          message("File ", f, " has been loaded.")
      }, XPT = {
          result <- read.xport(f, ...)
          message("File ", f, " has been loaded.")
      }, xpt = {
          result <- read.xport(f, ...)
          message("File ", f, " has been loaded.")
```

read.file (continued some more) showing some error messages

R code

```
  }, Rda = {
           result <- f
           message("To load this ", suffix, " file (or these files) you need to load('",
               f, "')   \nCaution, this might replace an object currently in
                    your environment.")
       }, SYD = {
           result <- read.systat(f, to.data.frame = to.data.frame)
           message("Data from the systat SYD file ", f, " has been loaded.")
       }, jmp = {
           result <- f
           message("I am sorrry.  To read this .jmp file, it must first be saved
             as either a \"txt\" or \"csv\" file.
                If you insist on using SAS formats, try .xpt or .XPT")
       }, sas7bdat = {
           result <- f
           message("I am sorry.  To read this .sas7bdat file, it must first be saved as
             either a xpt, or    XPT file in SAS, or as a \"txt\" or \"csv\" file. \
                 ?read.ssd in foreign for help.")
       }, {
           message("I  am sorry. \nI can not tell from the suffix what file type is this.
            Rather than try to read it, I will let you specify a better format.")
       })
   }
   return(result)
}
```

A brief example with real data

1. Get the data
2. Descriptive statistics
   - Graphic
   - Numerical
3. Inferential statistics using the linear model
   - regressions
4. More graphic displays

Get the data and describe it

1. First read the data, either from a built in data set, a local file, a remote file, or from the clipboard.
2. Describe the data using the describe function from *psych*

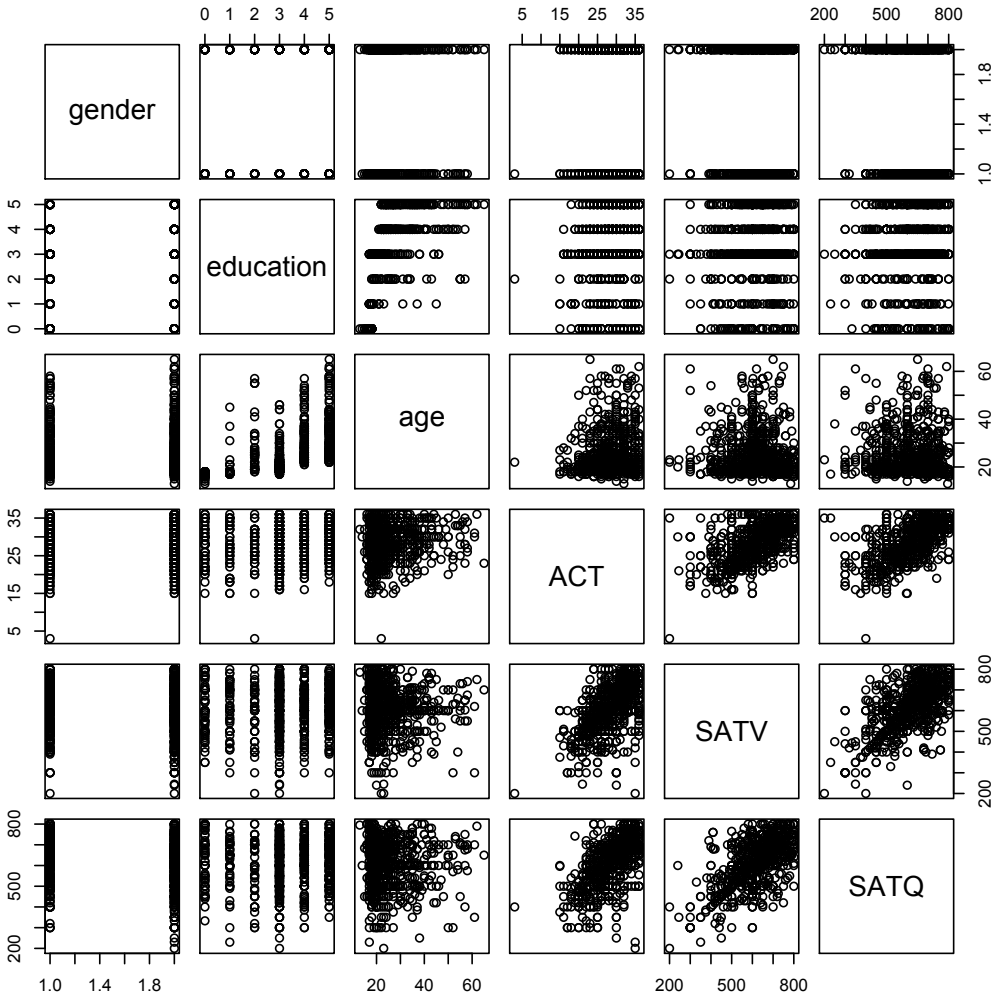
R code

```
my.data <- sat.act  #an example data file that is part of psych
#or
#my.data <-read.file()   #look for it on your hard drive
#or
file.name <-"http://personality-project.org/r/aps/sat.act.txt"
#now read it either locally or remotely
 my.data <- read.file(file.name)
#or if you have copied the data to the clipboard
# my.data <- read.clipboard()  #you can read it from there
describe(my.data) #report basic descriptive statistics
```

```
          var   n   mean     sd median trimmed    mad min max range  skew kurtosis   se
gender      1 700   1.65   0.48      2    1.68   0.00   1   2     1 -0.61    -1.62 0.02
education   2 700   3.16   1.43      3    3.31   1.48   0   5     5 -0.68    -0.06 0.05
age         3 700  25.59   9.50     22   23.86   5.93  13  65    52  1.64     2.47 0.36
ACT         4 700  28.55   4.82     29   28.84   4.45   3  36    33 -0.66     0.56 0.18
SATV        5 700 612.23 112.90    620  619.45 118.61 200 800   600 -0.64     0.35 4.27
SATQ        6 687 610.22 115.64    620  617.25 118.61 200 800   600 -0.59     0.00 4.41
```

Graphic display of data using `pairs`

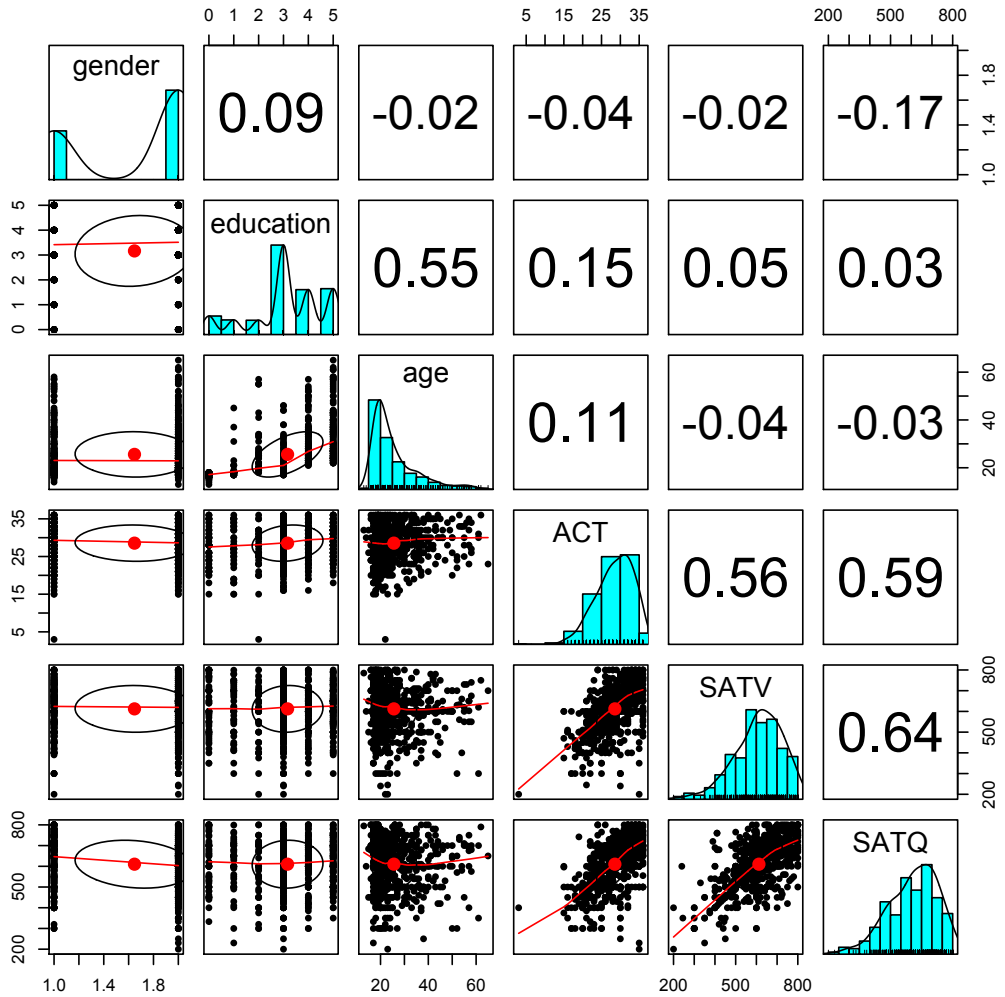
pairs(my.data) #Note the outlier for ACT



1. A Scatter Plot Matrix (SPLOM) plot
2. The diagonal shows the vairables
3. The pairwise scatter plots show the column as x axis and row as y axis
4. A lot of redundancy in the grapch
5. `pairs.panels` extended from `pairs`

Graphic display of data using `pairs.panels`

pairs.panels(my.data) #Note the outlier for ACT



1. A Scatter Plot Matrix (SPLOM) plot
2. The diagonals show the histogram and densities
3. The pairwise scatter plots show the column as x axis and row as y axis
4. The correlations reflect the row and column
5. `pairs.panels` extended from `pairs`

Clean up the data using `scrub`. Use ?scrub for help on the parameters.

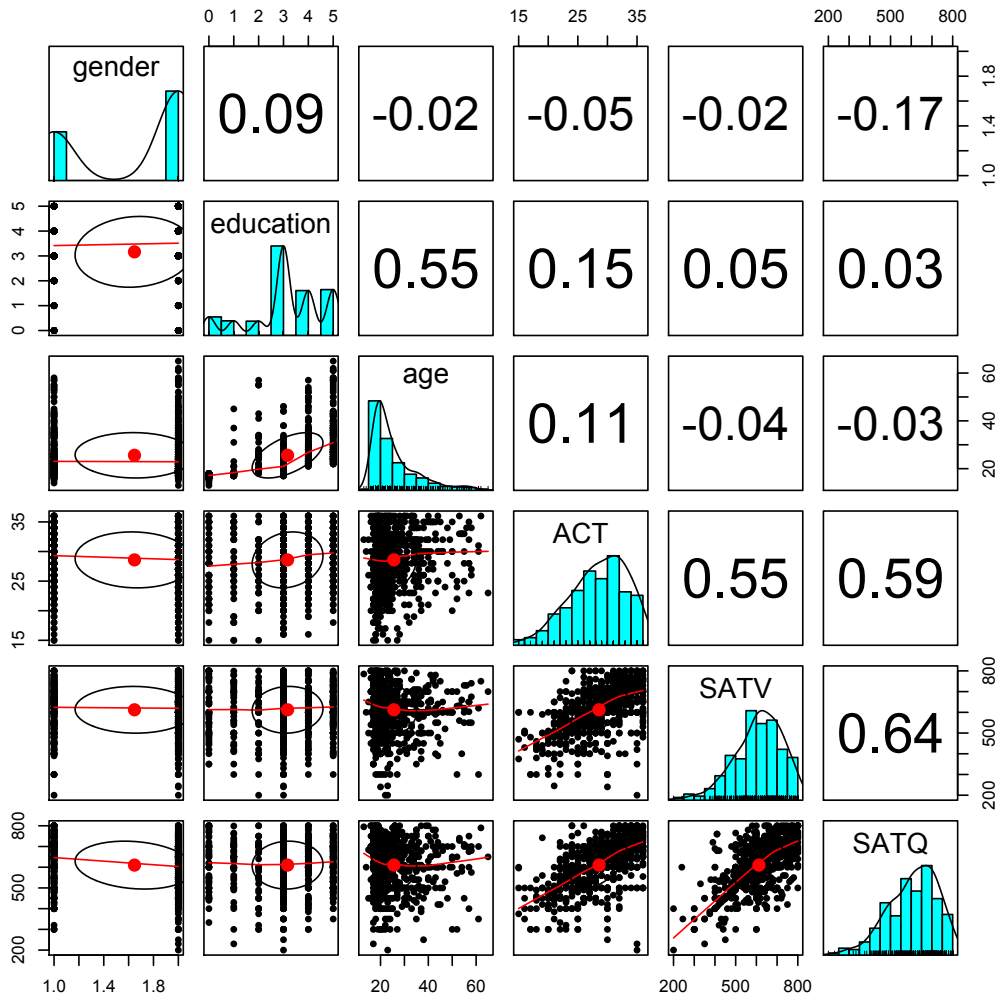
We noticed an outlier in the ACT data in the previous graph (you always graph your data, don't you).
We also noticed that the minimum value for ACT was unlikely (of course, you always describe your data).
So we change any case below 4 on the ACT to be missing (NA).

R code

```
cleaned <- scrub(my.data,"ACT",min=4)  #what data set,
               #which variable, what value to fix
describe(cleaned)   #look at the data again
pairs.panels(cleaned)
```

	var	n	mean	sd	median	trimmed	mad	min	max	range	skew	kurtosis	se
gender	1	700	1.65	0.48	2	1.68	0.00	1	2	1	-0.61	-1.62	0.02
education	2	700	3.16	1.43	3	3.31	1.48	0	5	5	-0.68	-0.06	0.05
age	3	700	25.59	9.50	22	23.86	5.93	13	65	52	1.64	2.47	0.36
ACT	4	699	28.58	4.73	29	28.85	4.45	15	36	21	-0.50	-0.36	0.18
SATV	5	700	612.23	112.90	620	619.45	118.61	200	800	600	-0.64	0.35	4.27
SATQ	6	687	610.22	115.64	620	617.25	118.61	200	800	600	-0.59	0.00	4.41

Graphic display of cleaned data using `pairs.panels`



1. A Scatter Plot Matrix (SPLOM) plot
2. The diagonals show the histogram and densities
3. The pairwise scatter plots show the column as x axis and row as y axis
4. The correlations reflect the row and column

Find the pairwise correlations, round to 2 decimals

This also shows how two functions can be nested. We are rounding the output of the cor function.

R code

```
#specify all the parameters being passed
round(cor(x=sat.act,use="pairwise"),digits=2)
#the short way to specify the rounding parameter
round(cor(cleaned,use="pairwise"),2)
```

```
          gender education   age   ACT  SATV  SATQ
gender      1.00      0.09 -0.02 -0.05 -0.02 -0.17
education   0.09      1.00  0.55  0.15  0.05  0.03
age        -0.02      0.55  1.00  0.11 -0.04 -0.03
ACT        -0.05      0.15  0.11  1.00  0.55  0.59
SATV       -0.02      0.05 -0.04  0.55  1.00  0.64
SATQ       -0.17      0.03 -0.03  0.59  0.64  1.00
```

Display it differently using the lowerCor function

Operations that are done a lot may be made into your own functions. Thus, `lowerCor` finds the pairwise correlations, rounds to 2 decimals, displays the lower half of the correlation matrix, and then abbreviates the column labels to make them line up nicely

R code

```
lowerCor(cleaned)
```

```
           gendr edctn age   ACT   SATV  SATQ
gender      1.00
education   0.09  1.00
age        -0.02  0.55  1.00
ACT        -0.05  0.15  0.11  1.00
SATV       -0.02  0.05 -0.04  0.55  1.00
SATQ       -0.17  0.03 -0.03  0.59  0.64  1.00
lowerCor
function (x, digits = 2, use = "pairwise", method = "pearson") #optional default values
{
    R <- cor(x, use = use, method = method) #specify options to cor
    lowerMat(R, digits) #this passes the digits parameter from the function to this function
    invisible(R)  #return (but don't show) the square matrix without rounding
}
```

Sampling error

1. Our data are thought to be samples from a population
2. Samples are probabilistic and will tend towards the population as the sample is bigger
3. Errors of sampling error will tend towards normal theory as sample sizes get bigger
4. For small samples, the t-distribution better represents the probable error
5. But what if the sample is not normal?
6. Enter the bootstrap

Bootstrap resampling (Efron, 1979; Efron and Gong, 1983)

1. Consider the observed sample as the population
2. Sample from this (with replacement) many times
3. Find the relevant statistic in these many resamplings
4. This will give an idea of the variability of our estimates

Bootstrap resampling

1. Given a distribution of n numbers (say 100), sample them with replacement N times
2. sample is of size n but some of those are the same people
3. How many of them are repeated?
4. We can find out by doing it!

The sampling of samples

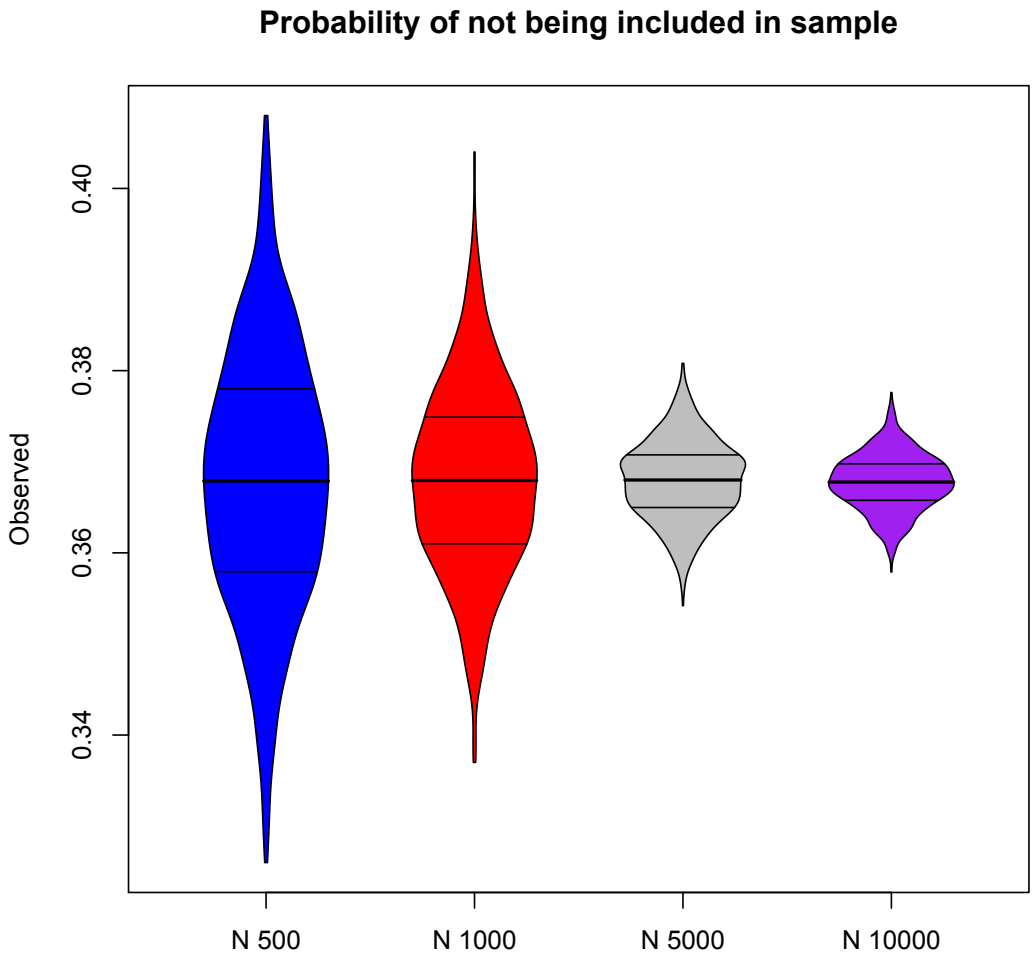
R code

```
N <- c(500,1000,5000,10000)  #number of observations
N.trials <- 1000 #how many replications
set.seed(42)
results <- matrix(rep(NA,N.trials * length(N)),ncol=length(N))
for(j in  1: length(N)){
original <- seq(1:N[j])
for (i in 1:1000) {  #number of replications to show the effect
samp <- sample(original,N[j],replace = TRUE)
results[i,j] <- 1 - sum(original %in% samp)/N[j]
}  }
colnames(results) <- paste("N",N)
print(describe(results),3)
violin(results,main ="Probability of not being included in sample")
```

```
print(describe(results),3)
        vars    n  mean    sd median trimmed   mad   min   max range   skew kurtosis se
N 500      1 1000 0.368 0.014  0.368   0.368 0.015 0.326 0.408 0.082 -0.043   -0.055  0
N 1000     2 1000 0.368 0.010  0.368   0.368 0.010 0.337 0.404 0.067  0.052    0.085  0
N 5000     3 1000 0.368 0.004  0.368   0.368 0.004 0.354 0.381 0.027 -0.039   -0.169  0
N 10000    4 1000 0.368 0.003  0.368   0.368 0.003 0.358 0.378 0.020 -0.063   -0.054  0
```

NORTHWESTERN UNIVERSITY

Density distribution of 1000 replications of bootstrap



Testing the significance of one correlation using `cor.test`.

The Core-R function uses normal theory

R code

```
cor.test(my.data$ACT,my.data$SATQ)
```

```
        Pearson's product-moment correlation

data:  my.data$ACT and my.data$SATQ
t = 18.9822, df = 685, p-value < 2.2e-16
alternative hypothesis: true correlation
                  is not equal to 0
95 percent confidence interval:
 0.5358435 0.6340672
sample estimates:
      cor
0.5871122
```

1. Specify the variables to correlate
2. Various statistics associated with the correlation.
3. But what if you want to do many tests? Use `corr.test`

Test many correlations for significance using `corr.test`

R code

```
corr.test(cleaned)
```

```
all:corr.test(x = cleaned)
Correlation matrix
          gender education   age   ACT  SATV  SATQ
gender      1.00      0.09 -0.02 -0.05 -0.02 -0.17
education   0.09      1.00  0.55  0.15  0.05  0.03
age        -0.02      0.55  1.00  0.11 -0.04 -0.03
ACT        -0.05      0.15  0.11  1.00  0.55  0.59
SATV       -0.02      0.05 -0.04  0.55  1.00  0.64
SATQ       -0.17      0.03 -0.03  0.59  0.64  1.00
Sample Size
          gender education age ACT SATV SATQ
gender       700       700 700 699  700  687
...
SATQ         687       687 687 686  687  687
Probability values (Entries above the diagonal are
                          adjusted for multiple tests.)
          gender education  age  ACT SATV SATQ
gender      0.00      0.17 1.00 1.00    1    0
education   0.02      0.00 0.00 0.00    1    1
age         0.58      0.00 0.00 0.03    1    1
```

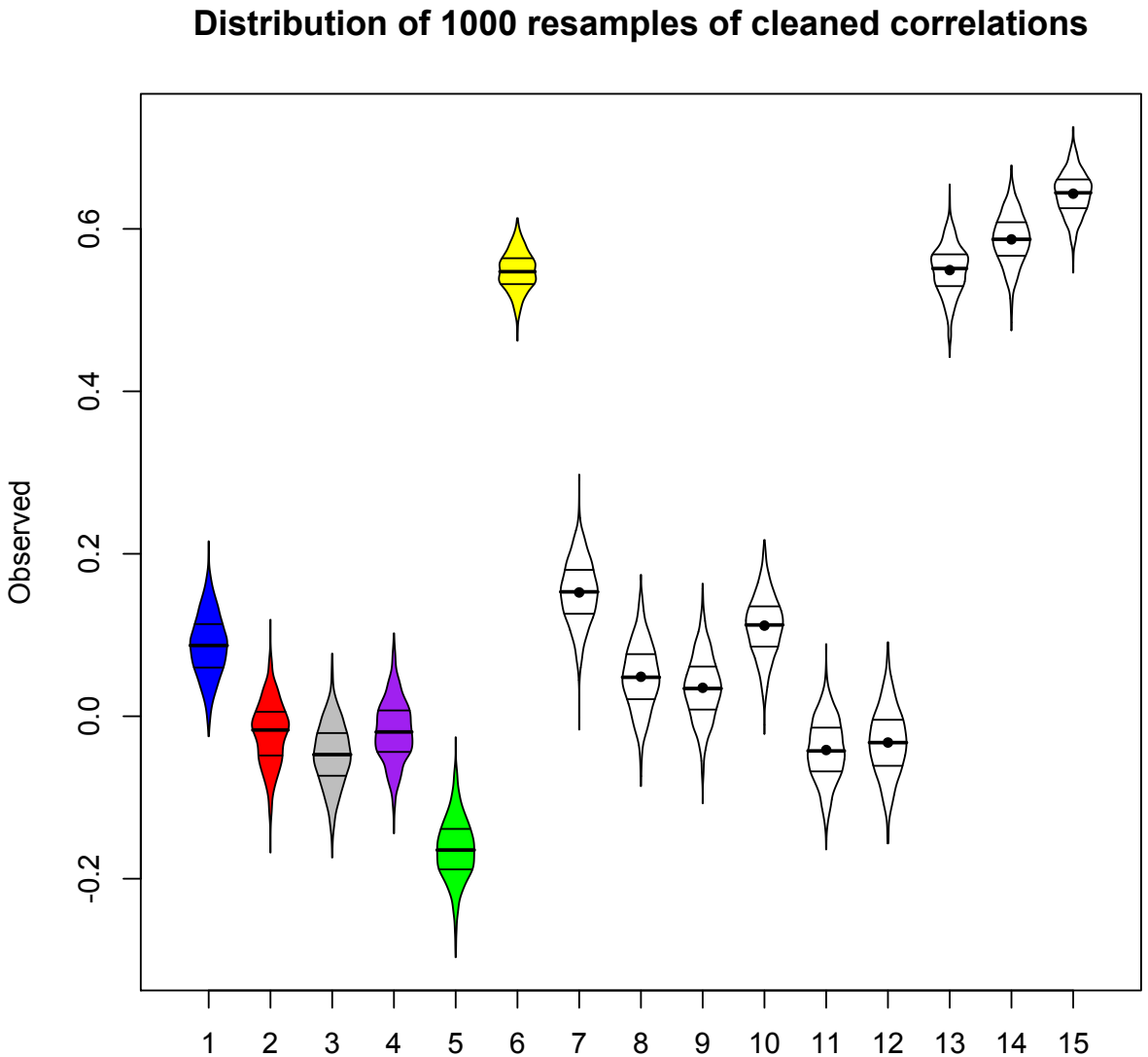
The `cor.ci` function does bootstrap resamplings

1. For each of N times, take samples of n subjects (with replacement)
2. Find the correlations for each sample
3. Save these as an object (replicated)
4. organize the data to find the empirical confidence limits

R code

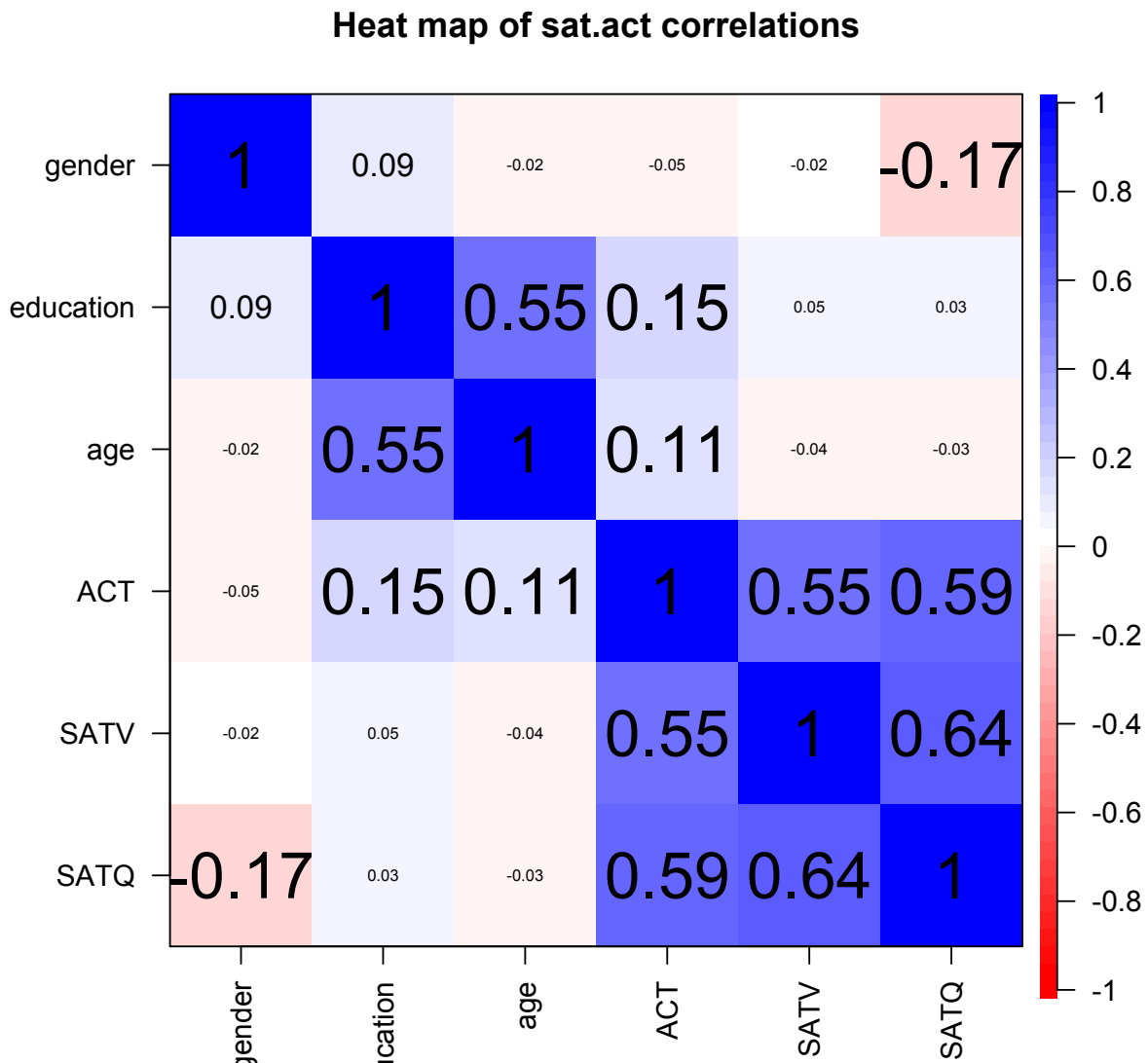
```
ci <- cor.ci(cleaned, n.iter=1000) #take 1000 replications
names(ci)
violin(ci$replicates,main="Distribution of 1000 resamples of cleaned correlations")
```

A violin plot of the resamples



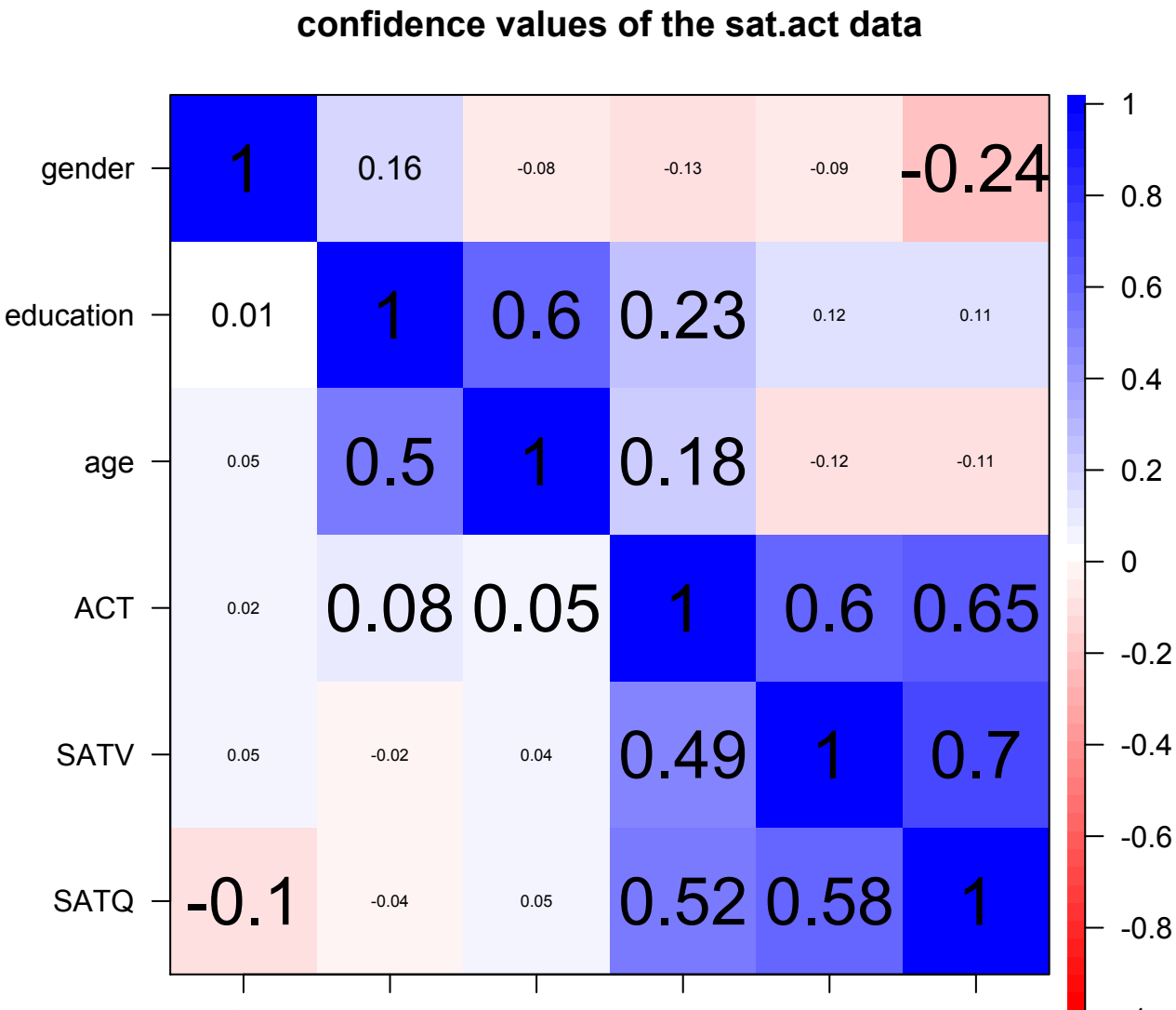
The SAT.ACT correlations. Confidence values from resampling

```
ci <- cor.ci(cleaned,main='Heat map of sat.act')
```



The SAT.ACT bootstrapped confidence intervals of correlation

cor.plot.upperLowerCi(ci,main="Heat map of sat.act")



Are education and gender independent? χ^2 Test of association

```
T <- with(my.data,table(gender,education))
```

```
> T
      education
gender   0   1   2   3   4   5
     1  27  20  23  80  51  46
     2  30  25  21 195  87  95
```

1. First create a table of associations
   - Do this on our data (my.data)
   - Use the "with" command to specify the data set
2. Show the table
3. Apply χ^2 test

```
> chisq.test(T)

        Pearson's Chi-squared test

data:  T
X-squared = 16.0851, df = 5, p-value = 0.006605
```

Finding χ^2 from a table of data

- Consider the effect of a treatment on later arrest (From Ashley Kendall, 2016)

Condition	Arrested	Not Arrested
Control	14	21
Treatment	3	23

R code

```
ak.df <- data.frame(Control=c(14,21),Treated =c(3,23))
rownames(ak.df) <- c("Arrested","Not Arrested")
ak.df #show the data frame
chisq.test(ak.df)  #Test it using the Yates continuity correction
```

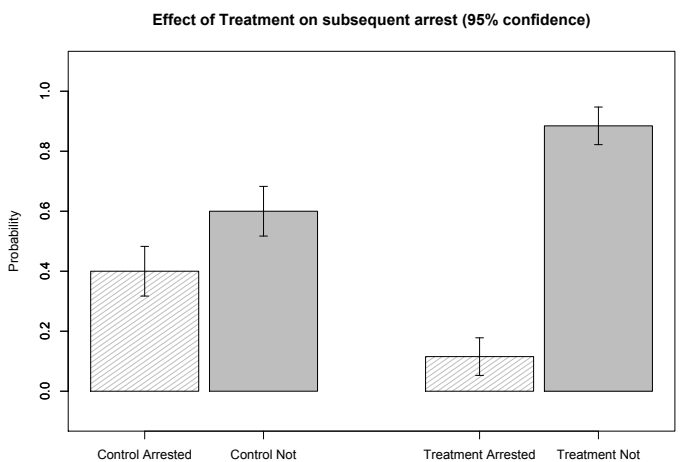
```
> ak.df #show the data frame
              Control Treated
Arrested           14       3
Not Arrested       21      23
> chisq.test(ak.df)  #Test it using the Yates continuity correctio
        Pearson's Chi-squared test with Yates' continuity correcti
data:  ak.df
X-squared = 4.6791, df = 1, p-value = 0.03053
```

NORTHWESTERN UNIVERSITY

Graph the tabled data showing confidence intervals of proportions

R code

```
ak.df <- data.frame(Control=c(14,21),Treated =c(3,23))
ak.p <- t(t(ak.df)/colSums(ak.df)) #convert to probabilities
standard.error  <- sqrt(ak.p[1,] * ak.p[2,]/colSums(ak.df))
stats <- data.frame(mean=as.vector(ak.p),
                          se=rep(standard.error,each=2))
rownames(stats) <- c("Control Arrested","Control Not",
                        "Treatment Arrested","Treatment Not")
error.bars(stats=stats,bars=TRUE,space=c(.1,.1,1,.1),
 density=c(20,-10,20,-10),ylab="Probability",
 xlab="Control vs Treatment",
 main ="Effect of Treatment on subsequent arrest (95% confidence)")
```




```
round(stats,2)
                      mean   se
Control Arrested      0.40 0.08
Control Not           0.60 0.08
Treatment Arrested    0.12 0.06
Treatment Not         0.88 0.06
```

NORTHWESTERN UNIVERSITY

Multiple regression and the general linear model

1. Use the sat.act data example
2. Do the linear model
3. Summarize the results

R code

```
mod1 <- lm(SATV ~ education + gender + SATQ,data=my.data)
summary(mod1,digits=2)
```

```
Call:
lm(formula = SATV ~ education + gender + SATQ, data = my.data)
Residuals:
    Min      1Q  Median      3Q     Max
-372.91  -49.08    2.30   53.68  251.93
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 180.87348   23.41019   7.726 3.96e-14 ***
education     1.24043    2.32361   0.534  0.59363
gender       20.69271    6.99651   2.958  0.00321 **
SATQ          0.64489    0.02891  22.309  < 2e-16 ***
Signif. codes:  0 Ô***Õ 0.001 Ô**Õ 0.01 Ô*Õ 0.05 Ô.Õ 0.1 Ô Õ 1
Residual standard error: 86.24 on 683 degrees of freedom
  (13 observations deleted due to missingness)
Multiple R-squared: 0.4231,     Adjusted R-squared: 0.4205
F-statistic:   167 on 3 and 683 DF,  p-value: < 2.2e-16
```

Zero center the data before examining interactions

In order to examine interactions using multiple regression, we must first "zero center" the data. This may be done using the `scale` function. By default, `scale` will standardize the variables. So to keep the original metric, we make the scaling parameter FALSE.

R code

```
csat <- data.frame(scale(my.data,scale=FALSE))
describe(csat)  #centered not standardized data
```

	vars	n	mean	sd	median	trimmed	mad	min	max
gender	1	700	0	0.48	0.35	0.04	0.00	-0.65	0.35
education	2	700	0	1.43	-0.16	0.14	1.48	-3.16	1.84
age	3	700	0	9.50	-3.59	-1.73	5.93	-12.59	39.41
ACT	4	700	0	4.82	0.45	0.30	4.45	-25.55	7.45
SATV	5	700	0	112.90	7.77	7.22	118.61	-412.23	187.77
SATQ	6	687	0	115.64	9.78	7.04	118.61	-410.22	189.78

Note that we need to take the output of scale (which comes back as a matrix) and make it into a data.frame if we want to use the linear model on it.

Zero center the data before examining interactions

R code

```
csat <- data.frame(scale(my.data,scale=FALSE))
mod2 <-  lm(SATV  ~ education * gender * SATQ,data=csat)
summary(mod2)
```

```
Call:
all:
lm(formula = SATV ~ education * gender * SATQ, data = csat)

Residuals:
    Min      1Q  Median      3Q     Max
-372.53  -48.76    3.33   51.24  238.50

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)           0.773576   3.304938   0.234  0.81500
education             2.517314   2.337889   1.077  0.28198
gender               18.485906   6.964694   2.654  0.00814 **
SATQ                  0.620527   0.028925  21.453  < 2e-16 ***
education:gender      1.249926   4.759374   0.263  0.79292
education:SATQ       -0.101444   0.020100  -5.047 5.77e-07 ***
gender:SATQ           0.007339   0.060850   0.121  0.90404
education:gender:SATQ 0.035822   0.041192   0.870  0.38481
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

Residual standard error: 84.69 on 679 degrees of freedom
  (13 observations deleted due to missingness)
Multiple R-squared:  0.4469,        Adjusted R-squared:  0.4412
F-statistic: 78.37 on 7 and 679 DF,  p-value: < 2.2e-16
```

Compare model 1 and model 2 using `anova`

Test the difference between the two linear models

R code

```
anova(mod1,mod2)
```

```
Analysis of Variance Table
Analysis of Variance Table

Model 1: SATV ~ education + gender + SATQ
Model 2: SATV ~ education * gender * SATQ
  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1    683 5079984
2    679 4870243  4    209742 7.3104 9.115e-06 ***
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1
```

lmCor as an alternative to lm

R code

```
mod3 <- lmCor(SATV ~ education + gender + SATQ,data=my.data)
mod3s <- lmCor(SATV ~ education + gender + SATQ,data=my.data,
          std=FALSE)
```

```
mod3
mod3a
Call: lmCor(y = SATV ~ education + gender + SATQ, data = my.data)

Multiple Regression from raw data

 DV =  SATV
            slope   se     t       p lower.ci upper.ci  VIF Vy.x
(Intercept)  0.00 0.03  0.00 1.0e+00    -0.06     0.06 1.00 0.00
education    0.02 0.03  0.55 5.8e-01    -0.04     0.07 1.01 0.00
gender       0.09 0.03  3.05 2.4e-03     0.03     0.15 1.04 0.00
SATQ         0.66 0.03 22.67 1.3e-85     0.60     0.72 1.03 0.43

Residual Standard Error =  0.76  with  696  degrees of freedom

 Multiple Regression
        R   R2 Ruw R2uw Shrunken R2 SE of R2 overall F df1 df2
SATV 0.65 0.43 0.4 0.16         0.42     0.03    172.37   3 696
```

lmCor (continued)

Not standardized

```
mod3a
mod3a
Call: lmCor(y = SATV ~ education + gender + SATQ, data = my.data, std = FALSE)

Multiple Regression from raw data

 DV =  SATV
             slope    se     t       p lower.ci upper.ci  VIF Vy.x
(Intercept) 179.88 23.08  7.79 2.4e-14   134.56   225.19 1.00 0.00
education     1.25  2.29  0.55 5.8e-01    -3.23     5.74 1.01 0.00
gender       21.08  6.90  3.05 2.4e-03     7.52    34.63 1.04 0.00
SATQ          0.65  0.03 22.67 1.3e-85     0.59     0.70 1.03 0.43

Residual Standard Error =  85.7  with  696  degrees of freedom

 Multiple Regression
       R   R2  Ruw R2uw Shrunken R2 SE of R2 overall F df1 df2        p
SATV 0.65 0.43 0.34 0.12        0.42     0.03    172.37   3 696 1.47e-83
```

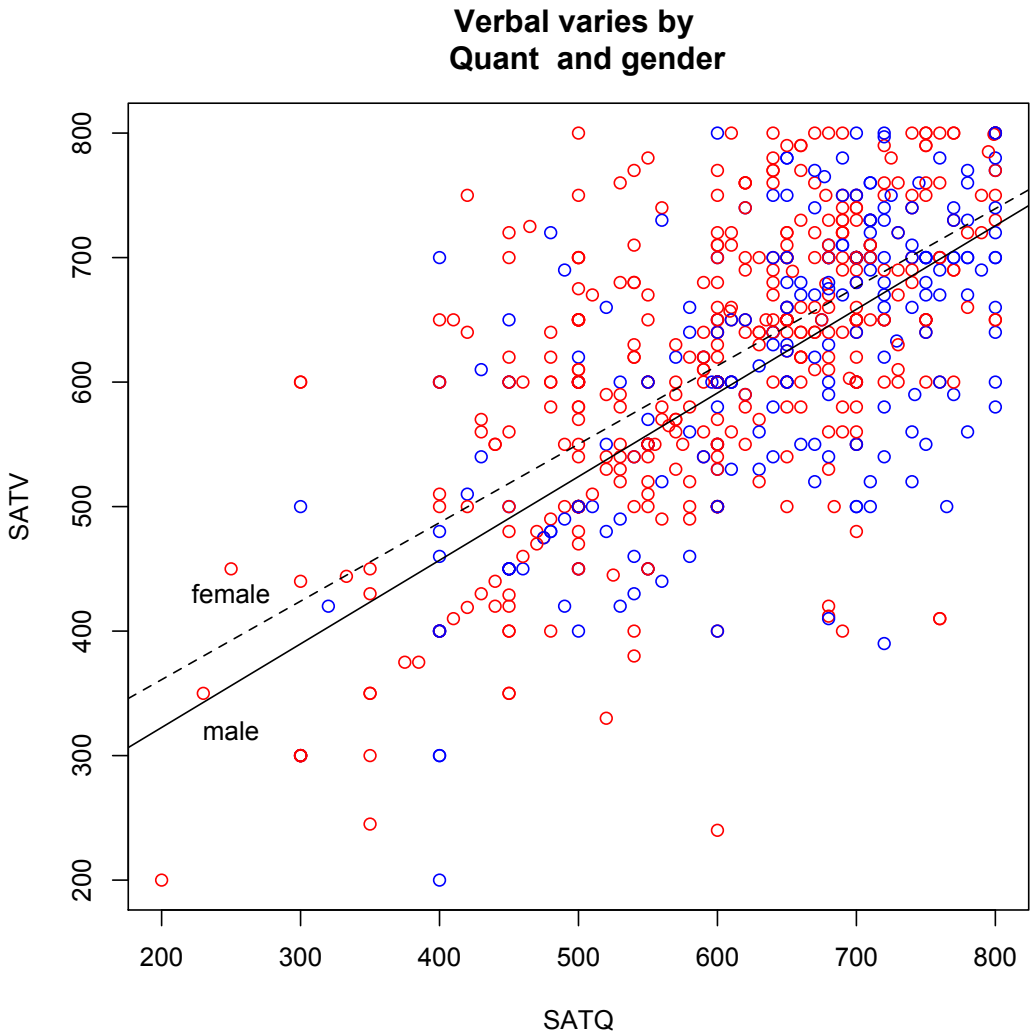
lmCor standardizes by default, particular useful when looking at interactions

R code

```
mod4 <-  lmCor(SATV  ~ education * gender * SATQ,data= my.data)
```

```
mod4
Call: lmCor(y = SATV ~ education * gender * SATQ, data = my.data)

Multiple Regression from raw data

 DV =  SATV
                      slope   se     t       p lower.ci upper.ci  VIF Vy.x
(Intercept)            0.00 0.03  0.00 1.0e+00    -0.06     0.06 1.00 0.00
education              0.03 0.03  1.10 2.7e-01    -0.02     0.09 1.06 0.00
gender                 0.08 0.03  2.75 6.0e-03     0.02     0.14 1.07 0.00
SATQ                   0.64 0.03 21.81 1.1e-80     0.58     0.69 1.07 0.41
education*gender       0.01 0.03  0.37 7.1e-01    -0.05     0.07 1.08 0.00
education*SATQ        -0.15 0.03 -5.11 4.2e-07    -0.21    -0.09 1.08 0.04
gender*SATQ            0.00 0.03  0.10 9.2e-01    -0.05     0.06 1.08 0.00
education*gender*SATQ  0.03 0.03  0.86 3.9e-01    -0.03     0.08 1.14 0.00

Residual Standard Error =  0.75  with  692  degrees of freedom

 Multiple Regression
        R   R2  Ruw R2uw Shrunken R2 SE of R2 overall F df1 df2        p
SATV 0.67 0.45 0.38 0.15        0.44     0.03     80.98   7 692 1.14e-85
```

Show the regression lines by gender



First plot all the data.

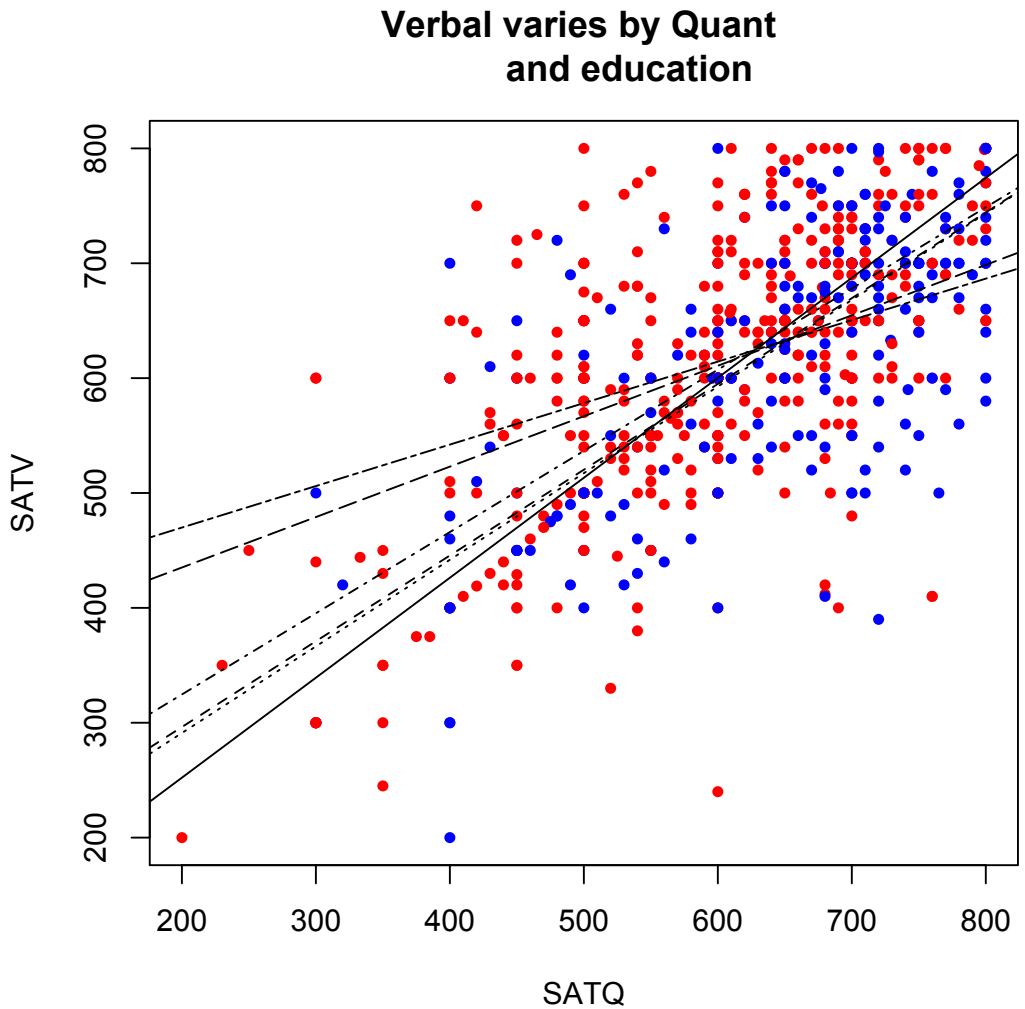
Then add the regression lines.

Then put a title on the whole thing.

R code

```
#first plot the data points
with(my.data,plot(SATV~SATQ,
 col=c("blue","red")[gender]))
#add the regression lines
by(my.data,my.data$gender,
   function(x) abline
  (lm(SATV~SATQ,data=x),
   lty=c("solid","dashed"
         )[x$gender]))
#add a title
title("Verbal varies by
   Quant  and gender")
 #label the lines
text(250,320,"male")
text(250,430,"female")
```

Show the regression lines by education



Do this again, but for levels of education as the moderator.

R code

```
with(my.data,plot(SATV~SATQ,
    col=c("blue","red")[gender],
    pch=20)) #plot character
by(my.data,my.data$education,
   function(x) abline
 (lm(SATV~SATQ,data=x),
 lty=c("solid", "dashed","dotted",
   "dotdash", "longdash",
  "twodash")[(x$education+1)]))

 title("Verbal varies by Quant
         and education")
```

Questions?

Efron, B. (1979). Bootstrap methods: Another look at the jackknife. *The Annals of Statistics*, 7(1-26).

Efron, B. and Gong, G. (1983). A leisurely look at the bootstrap, the jackknife, and cross-validation. *The American Statistician*, 37(1):36–48.