Review

- Getting start and getting help
- Calculator and Operators

  + * / - ^ < <= > >= == !=

- Objects:

  vector, factor, matrix, data frame and list

- Control structures

  Conditional statements and Loop

- Data import and save

  read.table(), save.image(), save

From Data to Graphics

Plot()

The following both plot **y** against **x**:

```
plot(y ~ x)   # Use a formula to specify the graph
plot(x, y)
```

Try

```
plot((0:20)*pi/10, sin((0:20)*pi/10), xlab="x", ylab="y")
x<-( 0:20)*pi/10
y<- sin((0:20)*pi/10)
plot(y~x)
```

Dataset that relates to lecture note

Datasets that relate to lecture note

- Download the R datafile: usingR.RData from ISpace
- Place this file in the working directory
- Within the R session, type: **load("usingR.RData")**
- **ls()** or **objects()**

Plot()

Example 1

```
str(elasticband)
plot(distance~stretch,data=elasticband)
```

or

```
attach(elasticband) # R now knows where to find distance & stretch
plot(distance ~ stretch)
```

Example 2

```
plot(ACT ~ Year, data=austpop, type="l")
plot(ACT ~ Year, data=austpop, type="b")
```

Try to find possible type by **?plot**

Plot(): adding lines, points and text on the plot

Example 3

```
plot(ACT~Year,data=austpop,type="n")
points(austpop$Year,austpop$ACT,pch=22,col="red")
lines(austpop$Year,austpop$ACT,type = "l", col="blue")
text(austpop$Year[5],austpop$ACT[5],"fifth")
title(main="austpop",xlab="Year",ylab="ACT")
```

Add a line

```
lines(austpop$Year,austpop$NT,type = "b", col="red")
```

Note: The **points()** function adds points to a plot. The **lines()** function adds lines to a plot. The **text()** function adds text at specified locations.

Plot(): Controlling axis

Example 4

```
windows()
plot(NT~Year,data=austpop,type="n",xaxt='n',xlab='Year',ylab='NT',
main="austpop")
axis(1,at=seq(1910,2000,by=10),cex.axis=0.6)
points(austpop$Year,austpop$NT,pch=22,col="red")
lines(austpop$Year,austpop$NT,type = "l", col="blue")
title(main="austpop")
```

windows() open a graphics window

The **axis()** function gives fine control over axis ticks and labels.

xlim=, ylim= specifies the lower and upper limits of the axes, for example with xlim=c(1, 10) or xlim=range(x)

Plot(): Size, color and choice of plotting symbol

- The parameter **cex** ("character expansion") controls the size
- The **col** ("colour") controls the colour of the plotting symbol.
- The parameter **pch** controls the choice of plotting symbol.
- The parameter lwd controls the thick of line.

Example 5

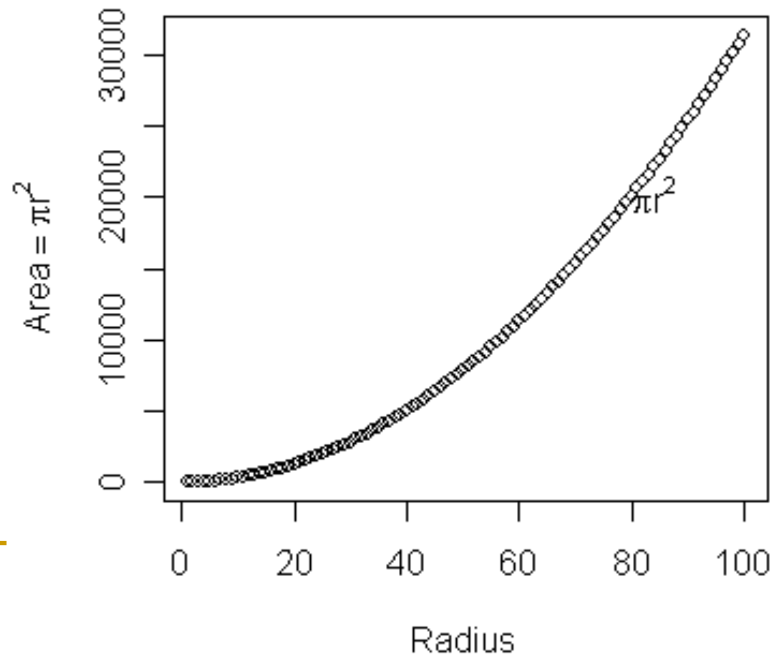
```
plot(1, 1, xlim=c(1, 7.5), ylim=c(0,5), type="n")
# Do not plot points
points(1:7, rep(4.5, 7), cex=1:7, col=1:7, pch=0:6)
text(1:7,rep(3.5, 7), labels=paste(0:6), cex=1:7, col=1:7)
lines(1:7,rep(2,7), lwd=2)
lines(1:7,rep(3,7), lwd=4,lty=3)
```

Try to find other controlling papameter by type '**?par**'

Plotting Mathematical Symbols

- By **expression()**

```
x<-1:100
y<-pi*x^2
plot(x, y, xlab="Radius", ylab=expression(Area == pi*r^2))
text(80,pi*80^2,expression(pi*r^2),c(0,0.4))
```

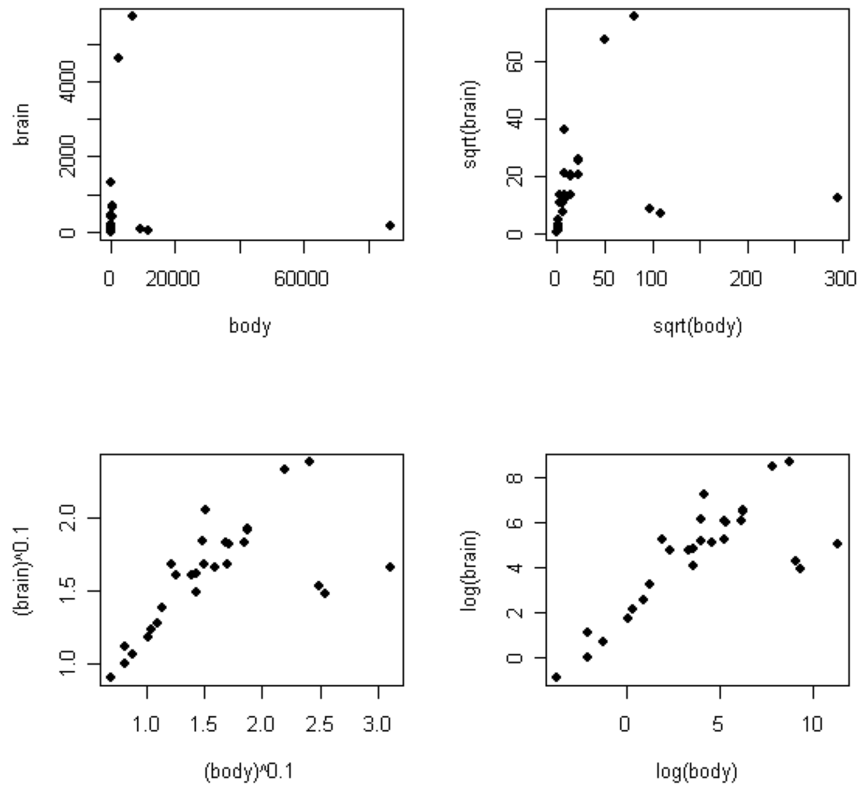


Multiple plots on the one page

- **mfrow():** subsequent plots appear row by row

library(MASS)
data(Animals)

par(mfrow=c(2,2), pch=16)
attach(Animals)
plot(body, brain)
plot(sqrt(body), sqrt(brain))
plot((body)^0.1, (brain)^0.1)
plot(log(body),log(brain))
detach(Animals)



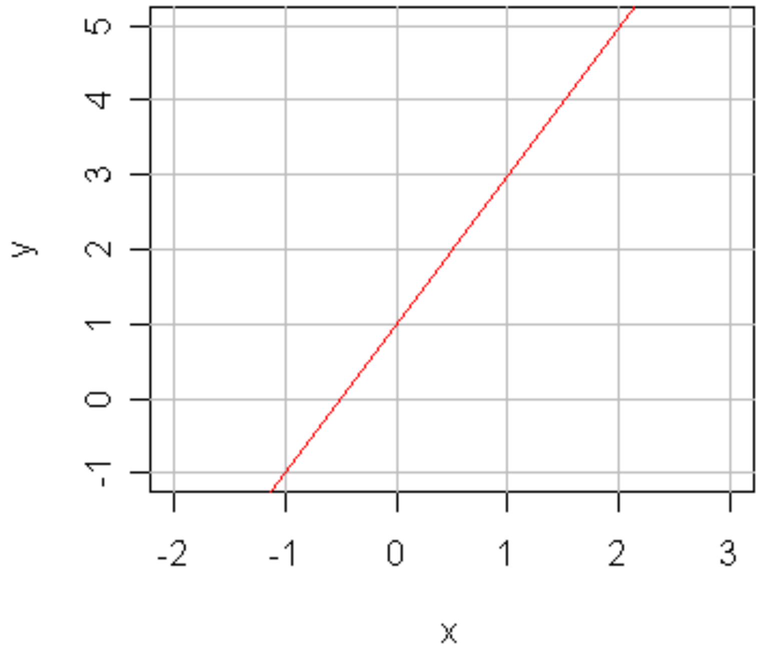
par(mfrow=c(1,1), pch=1) # Restore to 1 figure per page

- **mfcol():** subsequent plots appear column by column

Adding straight lines to plots

- **abline()**

```
plot(c(-2,3), c(-1,5), type = "n", xlab="x", ylab="y")
abline(h=0, v=0, col = "blue")
abline(h = -1:5, v = -2:3, col ="gray")
abline(a=1, b=2, col ="red")
```



Usage

abline(a , b , h, v)

a,b: the intercept and slope, single values

h: the y-value(s) for horizontal line(s).

v: the x-value(s) for vertical line(s).

Plots that show the distribution of data values

- **Histograms:**

Example:

```
library(MASS)
str(crabs)
x <- crabs$FL
y <- crabs$CL
op <- par(mfrow=c(2,1))
hist(x, col="light blue", xlim=c(0,50))
hist(y, col="light blue", xlim=c(0,50))
par(op)
```

- **Breakpoints:**

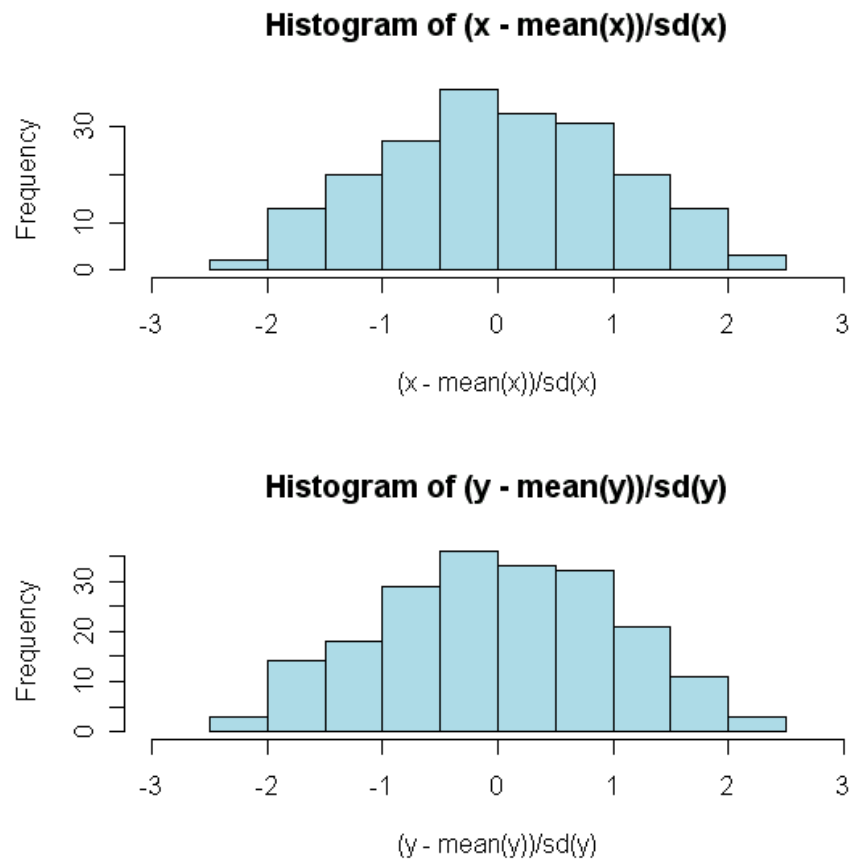
```
x <- crabs$FL
hist(x, col="light blue",breaks=(1:20)*1.5, xlim=c(0,30))
hist(x, col="light blue",breaks=20, xlim=c(0,30))
```

Plots that show the distribution of data values

- **Histograms:**

  Example: After normalization **(x-mean(x)/sd(x))**

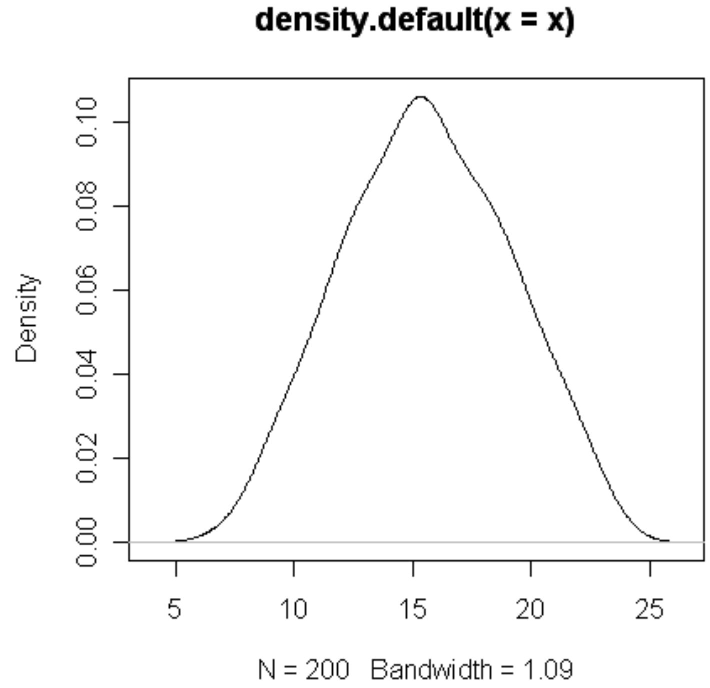
```
x <- crabs$FL
y <- crabs$CL
op <- par(mfrow=c(2,1))
hist( (x - mean(x)) / sd(x),
col = "light blue",xlim = c(-3, 3) )
hist( (y - mean(y)) / sd(y),
col = "light blue",xlim = c(-3, 3) )
par(op)
```



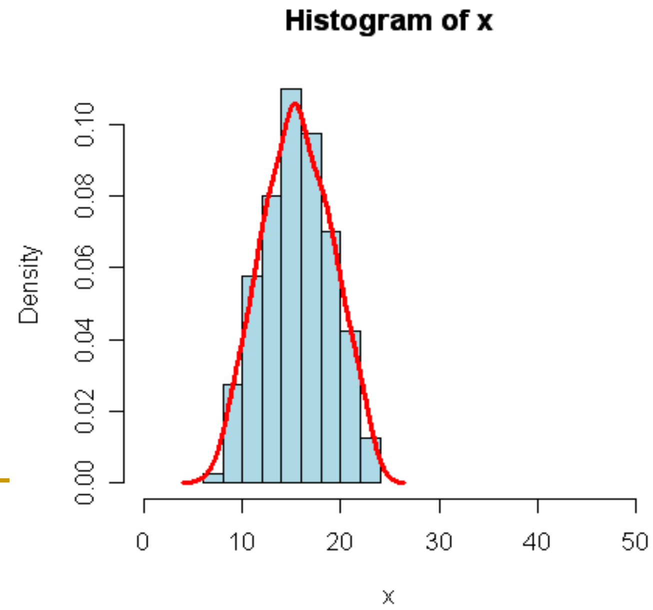
Plots that show the distribution of data values

- **Density Plots: density()**

```
str(crabs)
x <- crabs$FL
y <- crabs$CL
plot(density(x),type="l")
```



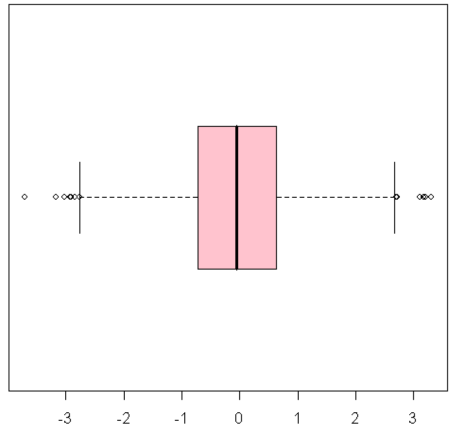

```
hist(x, col="light blue", xlim=c(0,50),
probability = TRUE)
lines(density(x), col = "red", lwd = 3)
```



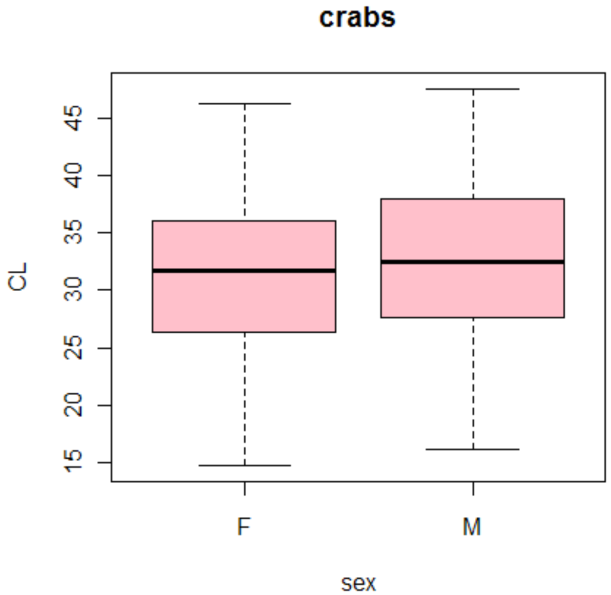
Box plot

- **Boxplot()**

N <- 2000
x <- rnorm(N)
boxplot(x, horizontal = TRUE, col = "pink“)
boxplot(x, col = "pink")



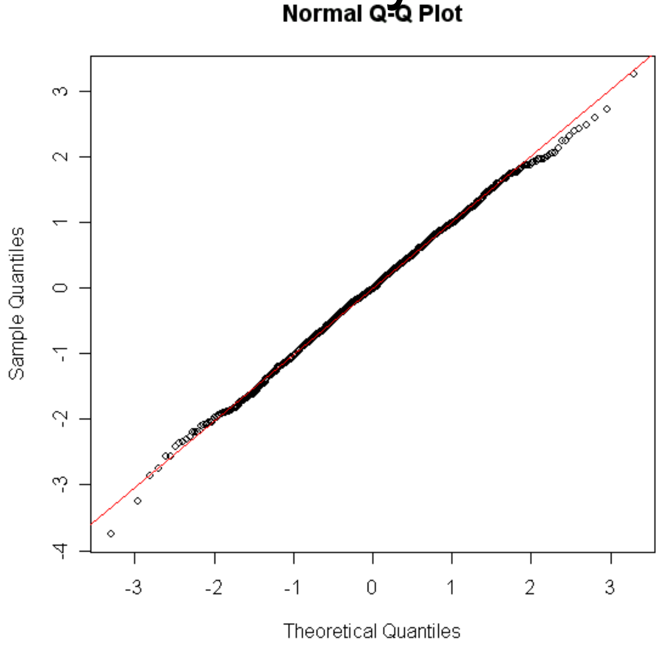
library(MASS)
boxplot(CL~sex,data=crabs,col='pink',xlab='sex',ylab='CL',main='crabs')



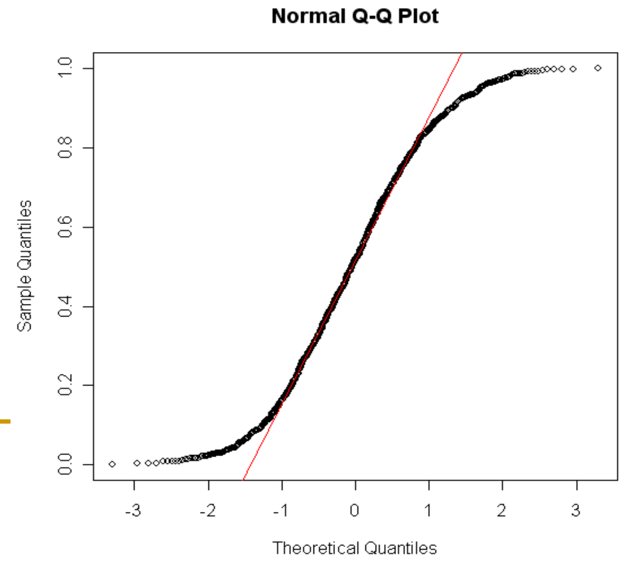
Normal probability plots

- qqnorm() gives a normal QQ plot of the values in y

```
n <- 1000
x <- rnorm(n)
qqnorm(x)
qqline(x, col="red")
```




```
x <- runif(n)
qqnorm(x)
qqline(x, col="red")
```

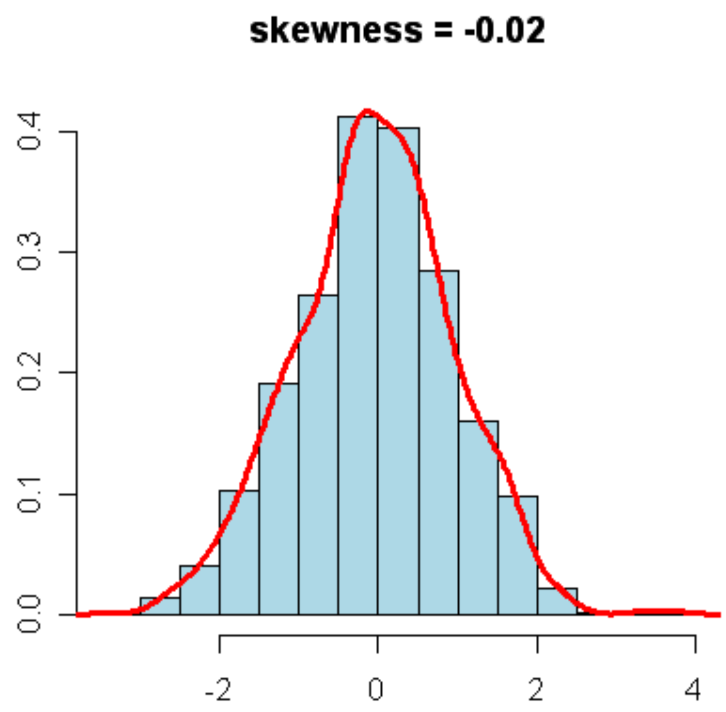


Example:

- Use of hist(), lines(), density(), skewness

```
n <- 1000
x <- rnorm(n)
skewness<- sum((x-mean(x))^3)/((n-1)*
(sd(x)^3))  # #Calculate the "skewness“

hist(x, col="light blue",
probability=TRUE,
main=paste("skewness =",
round(skewness,digits=2)), xlab="",
ylab="")

lines(density(x), col="red", lwd=3)
```

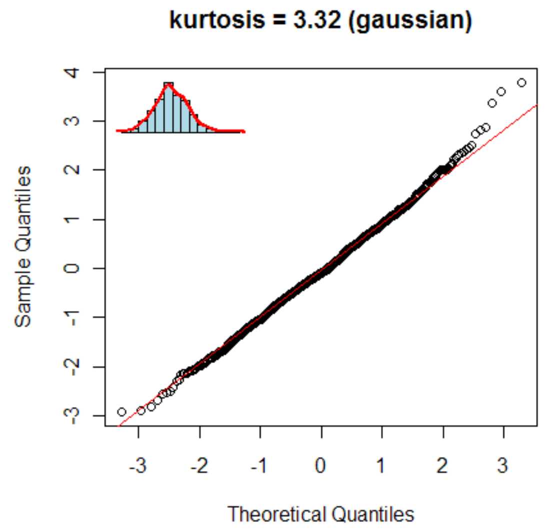
$$skewness = \frac{\sum_{i=1}^{N}(Y_i - \bar{Y})^3}{(N-1)s^3}$$



Example: Use of par(), qqnorm(), qqline(), hist(), lines(), density(), kurtosis

```
n <- 1000
x <- rnorm(n)
kurtosis<- sum((x-mean(x))^4)/((n-1)* (sd(x)^4))
#For the "kurtosis"
qqnorm(x, main=paste("kurtosis =", round(kurtosis, digits=2),
"(gaussian)"))
qqline(x, col="red")

op <- par(fig=c(.02,.5,.5,.98), new=TRUE)
hist(x, probability=T, col="light blue",
xlab="",ylab="", main="“, axes=F)
lines(density(x), col="red", lwd=2)
par(op)
```

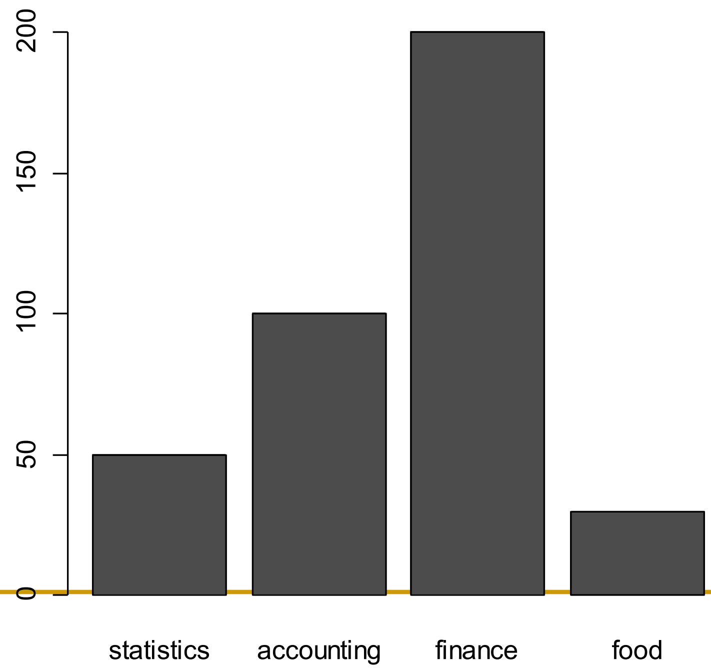
$$kurtosis = \frac{\sum_{i=1}^{N}(Y_i - \bar{Y})^4}{(N-1)s^4}$$



Plot of qualitative univariate variables

- Barplot()

```
barplot(height, ...)
statistics<-50;  accounting<-100;  finance<-200; food<-30
x<-data.frame(statistics,accounting,finance,food)
barplot(as.matrix(x))
```



Barplot()

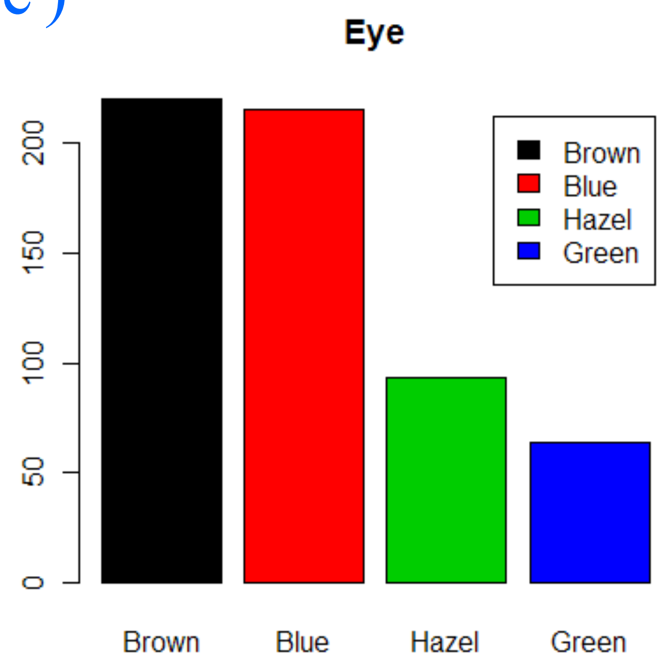
data(HairEyeColor)

```
> HairEyeColor
, , Sex = Male

       Eye
Hair    Brown Blue Hazel Green
  Black    32   11    10     3
  Brown    38   50    25    15
  Red      10   10     7     7
  Blond     3   30     5     8

, , Sex = Female

       Eye
Hair    Brown Blue Hazel Green
  Black    36    9     5     2
  Brown    81   34    29    14
  Red      16    7     7     7
  Blond     4   64     5     8
```

x <- apply(HairEyeColor, 2, sum)

```
> x
Brown  Blue Hazel Green
  220   215    93    64
```

barplot(x, col=1:4,legend.text = TRUE)
title('Eye')



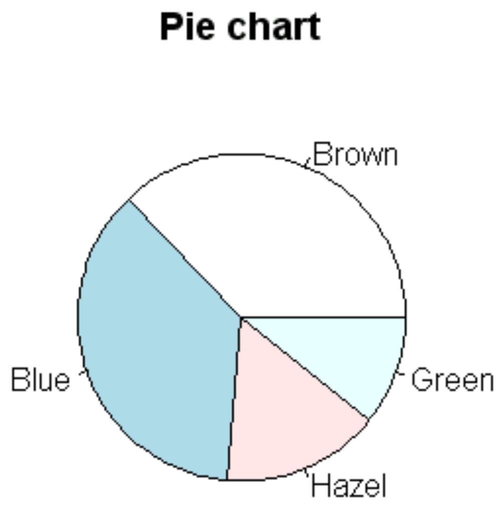
Plot of qualitative univariate variables

- Pie chart

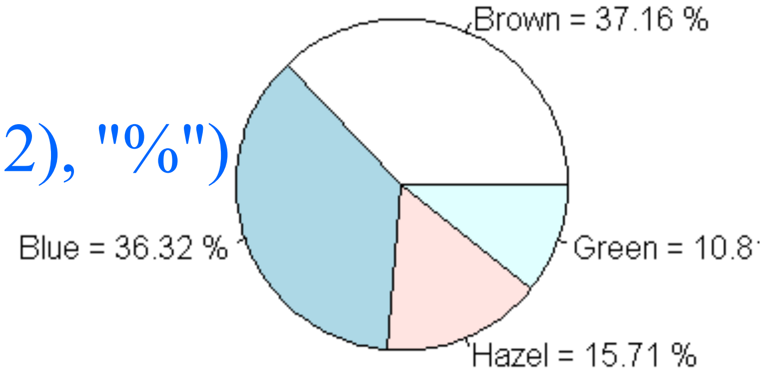
```
x <- apply(HairEyeColor, 2, sum)
```

```
> x
Brown  Blue Hazel Green
  220   215    93    64
```

```
pie(x)
title(main="Pie chart")
```




```
lx <- paste(names(x),"=",round(x/sum(x)*100,2), "%")
pie(x,lx)
```



Plot of Qualitative bivariate data (barplot)

data(HairEyeColor)

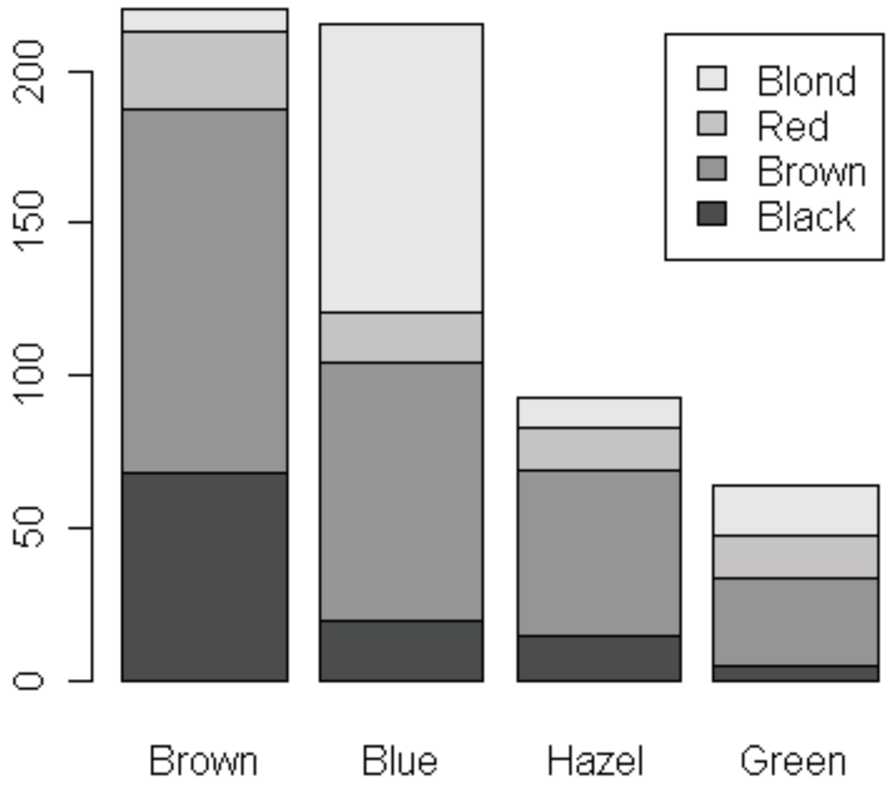
```
> HairEyeColor
, , Sex = Male

       Eye
Hair    Brown Blue Hazel Green
  Black    32   11    10     3
  Brown    38   50    25    15
  Red      10   10     7     7
  Blond     3   30     5     8

, , Sex = Female

       Eye
Hair    Brown Blue Hazel Green
  Black    36    9     5     2
  Brown    81   34    29    14
  Red      16    7     7     7
  Blond     4   64     5     8
```

a <-apply(HairEyeColor, c(1,2), sum)

```
> a
       Eye
Hair    Brown Blue Hazel Green
  Black    68   20    15     5
  Brown   119   84    54    29
  Red      26   17    14    14
  Blond     7   94    10    16
```

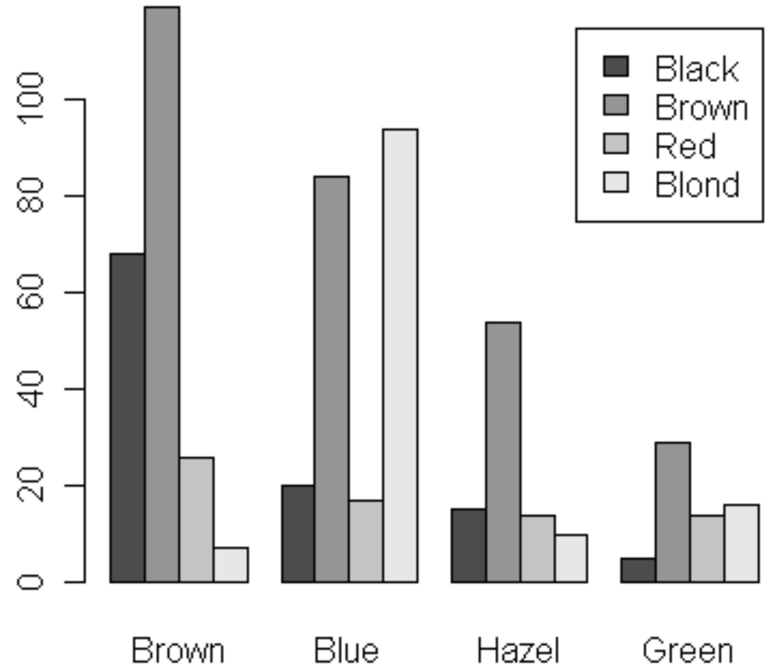
```
> a
         Eye
Hair     Brown Blue Hazel Green
  Black     68   20    15     5
  Brown    119   84    54    29
  Red       26   17    14    14
  Blond      7   94    10    16
```

barplot(a, legend.text=TRUE)



```
> a
        Eye
Hair     Brown Blue Hazel Green
  Black     68   20    15     5
  Brown    119   84    54    29
  Red       26   17    14    14
  Blond      7   94    10    16
```

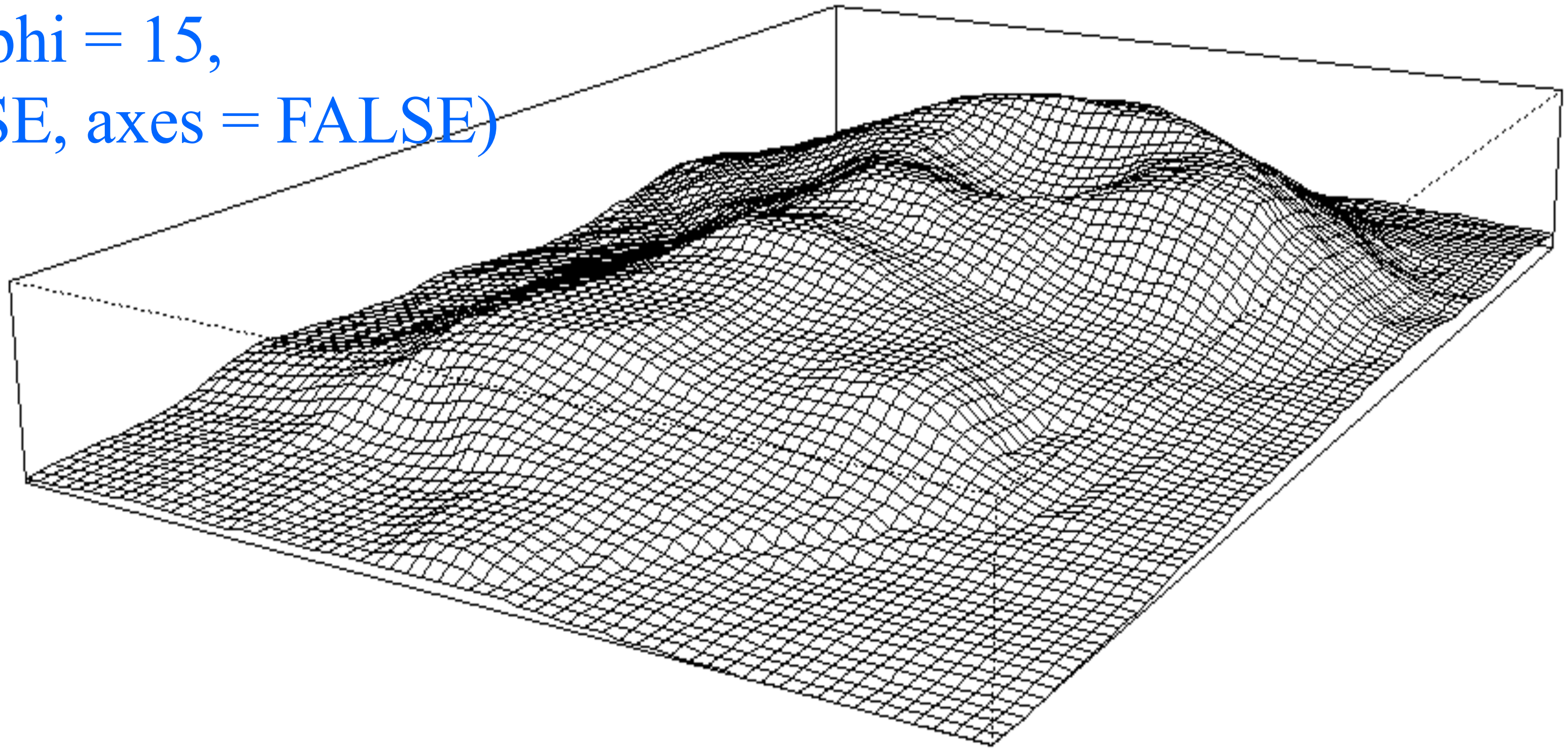
barplot(a,beside = TRUE,legend.text =TRUE)



3 Dimension plot

- Persp ():Draw perspective plots of surfaces over the x–y plane

```
data(volcano)
z <-  volcano
x <- 10 * (1:nrow(z))
y <- 10 * (1:ncol(z))
persp(x, y, z,
theta = 120, phi = 15,
scale = FALSE, axes = FALSE)
```



Data Volcano

Maunga Whau is a volcano in the Auckland volcanic field.
Data set volcano gives topographic information for Maunga Whau on a 10m by 10m grid.
Data is displayed by a matrix with 87 rows and 61 columns, rows corresponding to grid lines running east to west and columns to grid lines running south to north.

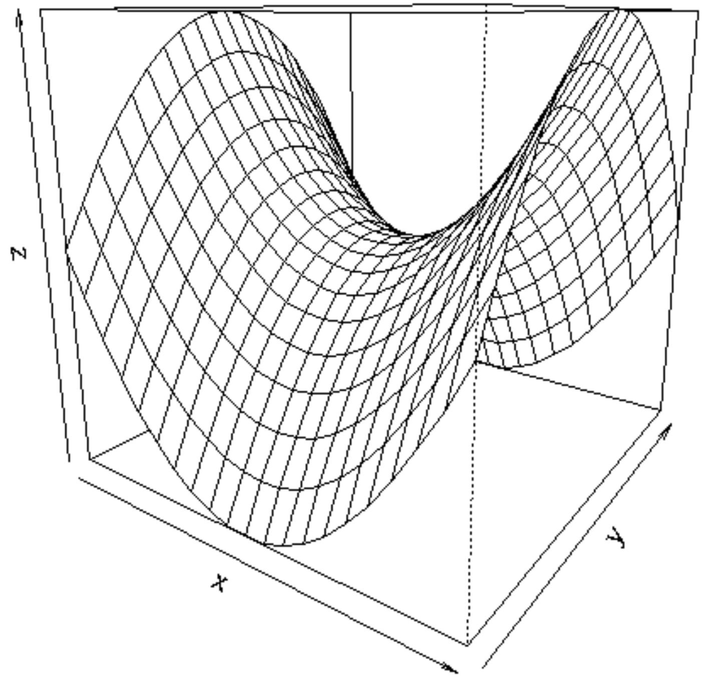
```
data(volcano)
z <-  volcano
```

```
> z[1:10,1:10]
      [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
 [1,]  100  100  101  101  101  101  101  100  100   100
 [2,]  101  101  102  102  102  102  102  101  101   101
 [3,]  102  102  103  103  103  103  103  102  102   102
 [4,]  103  103  104  104  104  104  104  103  103   103
 [5,]  104  104  105  105  105  105  105  104  104   103
 [6,]  105  105  105  106  106  106  106  105  105   104
 [7,]  105  106  106  107  107  107  107  106  106   105
 [8,]  106  107  107  108  108  108  108  107  107   106
 [9,]  107  108  108  109  109  109  109  108  108   107
[10,]  108  109  109  110  110  110  110  109  109   108
```

Example:

- Create a simple surface f(x,y) = x^2 - y^2

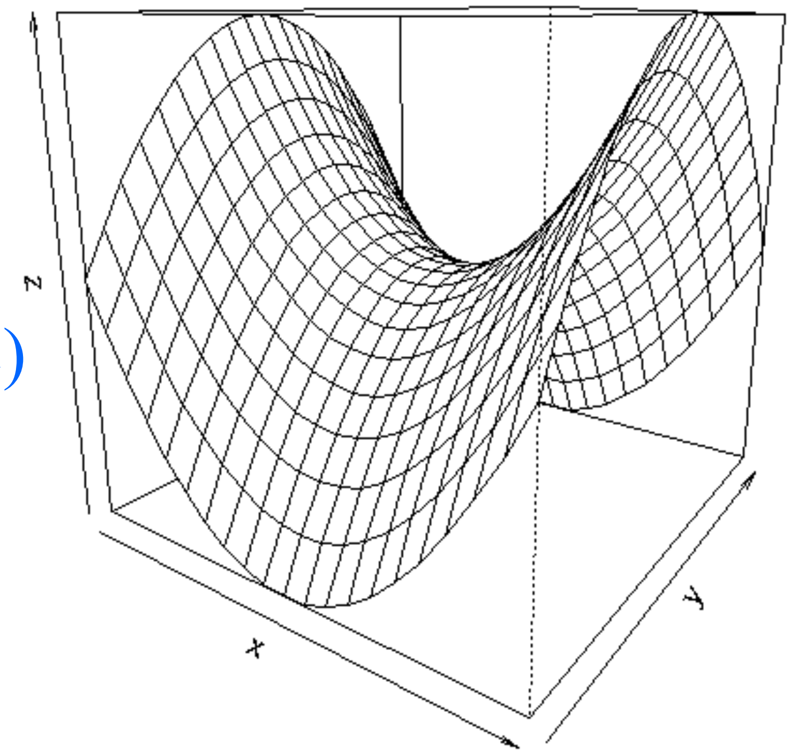
```
nx <- 21
ny <- 21
x <- seq(-1, 1, length = nx)
y <- seq(-1, 1, length = ny)
z<-
```



Example:

- Create a simple surface f(x,y) = x^2 - y^2

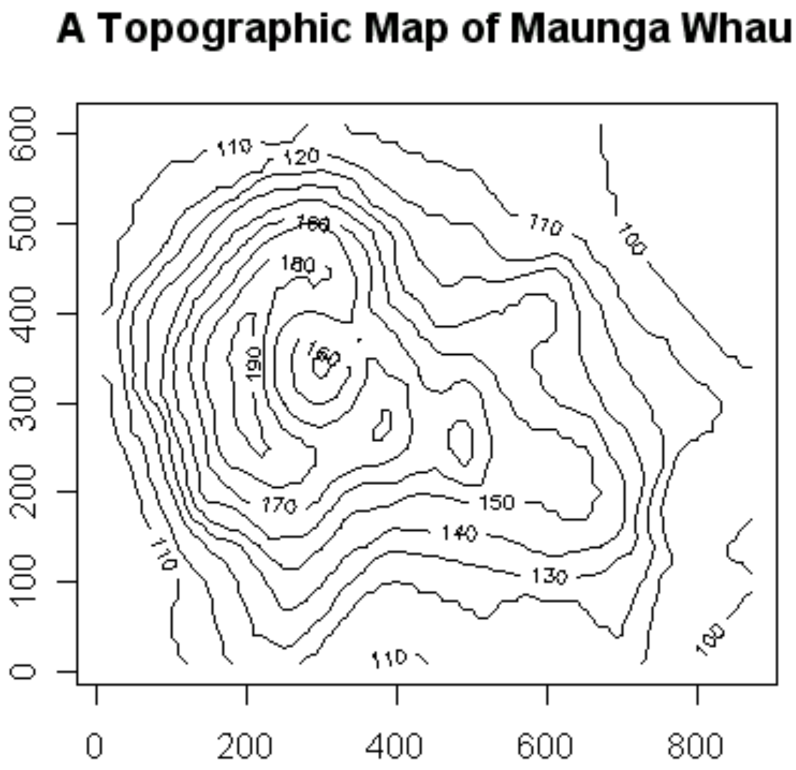
```
nx <- 21
ny <- 21
x <- seq(-1, 1, length = nx)
y <- seq(-1, 1, length = ny)
z <- outer(x, y, function(x,y) x^2 - y^2)
persp(x, y, z, theta = 35)
```



3 Dimension plot

- Contour(): Create a contour plot, or add contour lines to an existing plot

```
data("volcano")
x <- 10*1:nrow(volcano)
y <- 10*1:ncol(volcano)
rx<-range(x)
ry<-range(y)
plot(x = 0, y = 0,
type = "n",
xlim = rx, ylim = ry,
xlab = "", ylab = "")
contour(x, y, volcano,
add = TRUE,)
title("A Topographic Map of Maunga Whau", font = 4)
```



Image() and Contour():

```
data(volcano)
x <- 10*(1:nrow(volcano))
y <- 10*(1:ncol(volcano))
image(x, y, volcano,
col = terrain.colors(100),
axes = FALSE,
xlab = "", ylab = "")
contour(x, y, volcano,
levels = seq(90, 200, by=5),
add = TRUE)
axis(1, seq(100, 800, by = 100))
axis(2, seq(100, 600, by = 100))
title(main = "Maunga Whau Volcano", font.main = 4)
```