STAT 40001/STAT 50001 Statistical Computing

Lecture 6

Department of Mathematics and Statistics

Outline- Descriptive Statistics

- **Qualitative Data**
- **Quantitative Data**
- **Updating R Graphs**
- **Numerical Summary**

R Graphics- Qualitative

The **table** command allows us to look at frequency distribution.
Example: A survey asks people if they smoke or not.
Yes, No, No, Yes, Yes
We can enter this into R with the c() command, and summarize with the table command as follows

```
> x=c("Yes","No","No","Yes","Yes")
> table(x)
```

Note: The **table** command simply adds up the frequency. The Final grades of 25 students in MA 345 is given as below:
C,D,D,D,C,D,C,C, A,C,B, A,B, A,B,C,B,C, A, A,C, A,D,C,A
We will create a table as

```
> x<-c("C","D","D","D","C","D","C","C", "A","C","B", "A",
"B", "A","B","C","B","C", "A", "A","C", "A","D","C","A")
> table(x)
A B C D
7 4 9 5
```

factor

Categorical data is often used to classify data into various levels or factors. For example, the smoking data could be part of a broader survey on student health issues. R has a special class for working with factors which is occasionally important to know as R will automatically adapt itself when it knows it has a factor. To make a factor is easy with the command **factor** or **as.factor.**

```
>x=c("Yes","No","No","Yes","Yes")
>factor(x)
```

Bar Graph

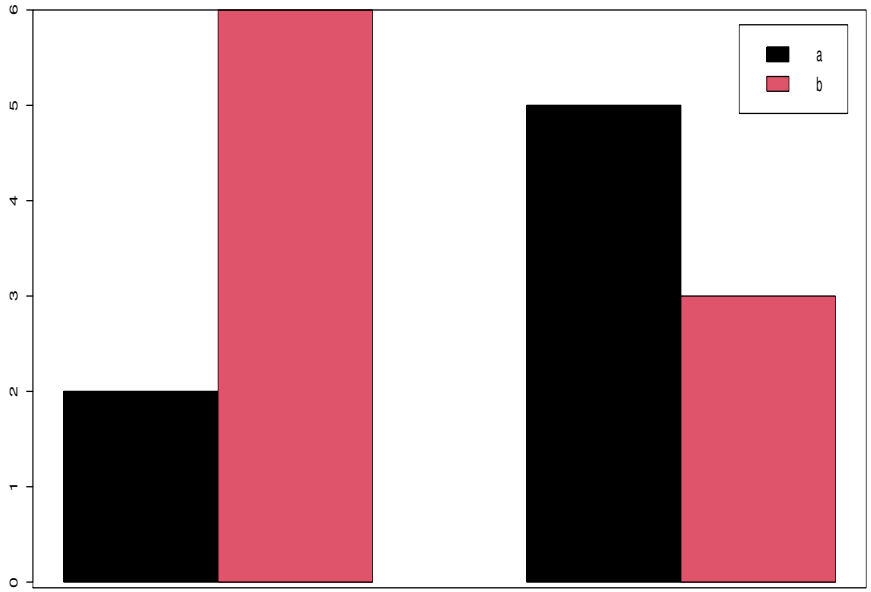
A bar chart draws a bar with height proportional to the count in the table. The height could be given by the frequency, or the proportion. A bar graph can be drawn for frequency data or relative frequency data

```
> grade<-c(12,20,10,3,5)
> names(grade)=c("A", "B", "C", "D", "F")
>barplot(grade,col=c(1,2,3,4,5),main="Grade Distribution")
```

```
grade<-c(12,20,10,3,5)
names(grade)=c("A", "B", "C", "D", "F")
barplot(grade/(sum(grade)),col=c(1,2,3,4,5),
main="Grade Distribution",xlab="Grade",ylab="Relative frequency"
```

Multiple bar graph

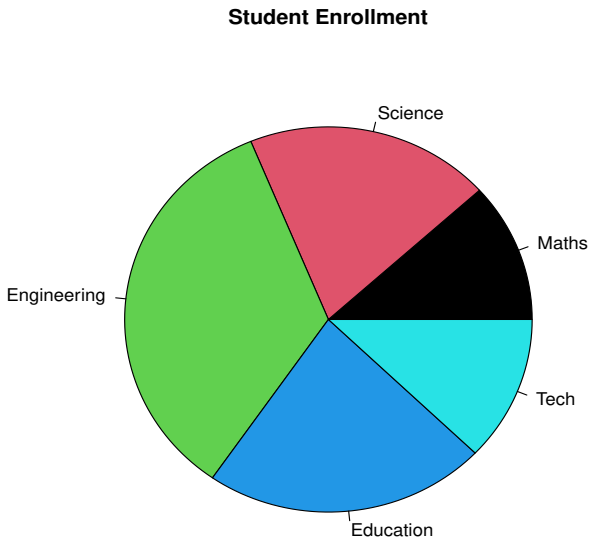
```
x<-c("a","a","b","b","b","b","b","b")
y<-c("a","a" ,"a" ,"a" ,"a","b","b","b")
cc<-cbind(table(x),table(y))
colnames(cc)<-c("x","y")
barplot(cc) # stacked plot
barplot(cc,beside=TRUE,col=c(1,2),legend=rownames(cc))
# side by side plot
```



Pie Chart

A circle divided into sectors that represent the percentages of a population or a sample that belongs to different categories is called a pie chart. To construct a pie chart, we multiply by 360 the relative frequency for each category to obtain the degree measure or size of the angle for the corresponding category.

```
>students<-c(120, 200, 350, 230,125)
>names(students)=c("Maths","Science","Engineering","Education","Tech")
>pie(students,col=c("1","2","3","4","5"),main="Student Enrollment")
```



Bar graph & Pie chart

Consider the blood group of 240 individuals

Blood group	n
A	56
B	75
AB	23
O	86

```
> blood<-c(56,75,23,86)
> pie(blood,col=c("1","2","3","4"),labels=c("A","B","AB","O"))
```

If you have 2 categorical variables, you might be interested in a 2-dimensional contingency table

```
> eyecol<-c(1,2,1,2,2,2,3,3,1,4,2,2,2,3,1)
> sex <- c(1,1,1,2,1,2,1,1,1,2,1,1,1,2,2)
> table(sex,eyecol)
> prop.table(table(sex,eyecol),1) # row percentages
> prop.table(table(sex,eyecol),2) # column percentages
```

Histogram

A histogram is a graphic that gives an idea of the "shape" of a sample, indicating regions where sample observations are concentrated and regions where they are sparse. A histogram is a graph in which classes are marked on the horizontal axis and either the frequencies, relative frequencies, or percentages are represented by the heights on the vertical axis. The number of classes suggested by Sturge's formula is $c = 1 + \log_2(n)$ where n is the number of observations in the data.

```
> x<-c (25,37,20,31,31,21,12,25,36,27,38,16,40,32,33,24,39,26,27,19)
> hist(x)
```

```
> x<-c (25,37,20,31,31,21,12,25,36,27,38,16,40,32,33,24,39,26,27,19)
> hist(x, main="Histogram of Chicago Temp.",xlab="Temperature")
```

```
> hist(x, breaks=15) it suggests 10 breaks
> hist(x, breaks = c(10, 20, 30, 40))# uses these breaks
> hist(x, breaks="scott") # Use "Scott" algorithm
```

Plotting Density Curve

The problems associated with drawing histograms and density functions of continuous variables are much more challenging. The subject of density estimation is an important issue for statisticians. You can get a feel for what is involved by browsing the **?density** help window. The algorithm used in **density.default** disperses the mass of the empirical distribution function over a regular grid. The choice of bandwidth is a compromise between smoothing enough to rub out insignificant bumps and smoothing too much so that real peaks are eliminated. The bandwidth is given as

$$b = \frac{max(x) - min(x)}{2(1 + \log_2 n)}$$

Frequency distribution table

Consider a data set

70,60,66,54,66,60,55,57,67,71,62,69,57,63,69,58,60,52,63,63,61,65,60,60,73

```
> data=c(70,60,66,54,66,60,55,57,67,71,62,69,57,63,69,58,
         60,52,63,63,61,65,60,60,73)
> range(data)
> class_size<- (73-52)/5 # Here 5 represents desired numbe
> breaks<-seq(50,75, by=5)
> prob.cut<-cut(data,breaks, right=FALSE)
> freqtable<-table(prob.cut)
> freqtable
> cbind(freqtable)
> cumfreq<-cumsum(freqtable)
> cumfreq
> cbind(freqtable, cumfreq)
        freqtable cumfreq
[50,55)         2       2
[55,60)         4       6
```

Example

We will access faithful data from the data set and plot density curve for eruptions.

```
>attach(faithful)
>(max(eruptions)-min(eruptions))/(2*(1+log(length(eruptions),base=2)))

> par(mfrow=c(1,2))
> hist(eruptions,15, freq=FALSE,main="Density Curve",col=2)
> lines(density(eruptions,width=0.6,n=200))
> truehist(eruptions,nbins=15, main="Density Curve",col=3)
> lines(density(eruptions,n=200))
```

Note that **truehist** is in the package **MASS**

Graphic Parameters, *par()*

The function **par()** is used to set or get graphical parameters. **par** allows us to plot multiple (x, y)'s in a single graphic. This is accomplished by selecting par(new=T) following each call to plot.

```
x<-seq(-5,5,0.1)
y1<-dnorm(x)
y2<-dcauchy(x)
y3<-0.5*dexp(abs(x))
yrange<-range(y1,y2,y3)
plot(x,y1,xlab="x",ylab="f(x)",lty=1, type="l",xlim=c(-5,5),ylim=yrange,col=1)
par(new=TRUE)
plot(x,y2,xlab="",ylab="",lty=3,type="l",xlim=c(-5,5),ylim=yrange,col=2)
par(new=TRUE)
plot(x,y3,xlab="",ylab="",lty=2,type="l",xlim=c(-5,5),ylim=yrange,col=4)
legend(1,.5,legend=c("N(0,1)","C(0,1)","L(0,1)"),lty=c(1,3,2),col=c(1,2,4))
title(cex=1,"probability density functions of standard Normal, standard Cauchy and
\n standard Laplace distributions")
```

Stem and Leaf Plot

Stem-and-leaf plot is a simple way of summarizing quantitative data. In a stem-and-leaf plot each data value is split into a "stem" and a "leaf". The "leaf" is usually the last digit of the number and the other digits to the left of the "leaf" form the "stem". Usually there is no need to sort the leaves, although computer packages typically do.

```
x=c(78,74,82,66,94,71,64,88,55,80,91,74,82,75,96,78,84,79,71,83)
stem(x)
```

```
The decimal point is 1 digit(s) to the right of the |
5 | 5
6 | 46
7 | 11445889
8 | 022348
9 | 146
```

Functions for calculating summary statistics of vector elements

Entry	Package	Description
min()	base	the minimum value of the numeric vector
max()	base	the maximum value of the numeric vector
range()	base	the range of the numeric vector
mean()	base	the arithmetic mean of the numeric vecto
median()	stats	the median of a numeric vector
quantile()	stats	various sample quantiles of a numeric vec
IQR()	stats	the inter-quartile range of a numeric vecto
fivenum()	stats	Tukey's five-number summary
sd()	stats	the standard deviation of a numeric vecto
var()	stats	the variance of a numeric vector
pmin()	base	the parallel minima of two or more numeri
pmax()	base	the parallel maxima of two or more numer
weighted.mean()	stats	the weighted mean of a numeric vector
mad()	stats	the median absolute difference of a numer

Functions for calculating summary statistics of vector elements

Entry	Package	Description
smean.sd()	Hmisc	the mean and standard deviation of a numeric
wtd.mean()	Hmisc	the weighted mean of a numeric vector
wtd.var()	Hmisc	the weighted variance of a numeric vector
wtd.quantile	Hmisc	the weighted quantiles of a numeric vector
ecdf()	stats	the empirical CDF (ECDF) of a numeric vecto
wtd.ecdf	Hmisc	the weighted ECDF of a numeric vector
wtd.rank	Hmisc	the weighted ranks of a numeric vector, using
describe	Hmisc	concise statistical description of vector, matrix
cor()	stats	the correlation between two numeric vectors o
cov()	stats	the covariance between two numeric vectors o
cov2cor()	stats	Scales a covariance matrix into the correspond
density()	stats	the kernel density estimates of a numeric vecto

Examples

Suppose, CEO yearly compensations are sampled and the following are found (in millions).
12, 0.4, 5, 2, 50, 8, 3, 1, 4, 0.25

```
> sals = scan() # read in with scan
1: 12 .4 5 2 50 8 3 1 4 0.25
11:
Read 10 items
> mean(sals) # the average
[1] 8.565
> var(sals) # the variance
[1] 225.5145
> sd(sals) # the standard deviation
[1] 15.01714
> median(sals) # the median
[1] 3.5
> fivenum(sals) # min,lower hinge, Median, upper hinge, max
[1] 0.25 1.00 3.50 8.00 50.00
> summary(sals)
Min. 1st Qu. Median Mean 3rd Qu. Max.
0.250 1.250 3.500 8.565 7.250 50.000
> mean(sals,trim=1/10) # This computes the 10%trimmed mean
[1]  4.425
> IQR(sals)
[1] 6
```

Examples

```
> data=c(10, 17, 18, 25, 28, 28)
> summary(data)
Min. 1st Qu. Median Mean 3rd Qu. Max.
10.00 17.25 21.50 21.00 27.25 28.00
> quantile(data,.25)
25%
17.25
> quantile(data,c(.25,.75)) # two values of p at once
25% 75%
17.25 27.25
```

Example

```
> x<-c(2,4,5,6,4,5,6,7,8,9,12,23)
> y<-c(5,1,3,5,6,7,8,9,3,21,13,21)
> pmin(x,y)
 [1]  2  1  3  5  4  5  6  7  3  9 12 21
> pmax(x,y)
 [1]  5  4  5  6  6  7  8  9  8 21 13 23
> range(x)
[1]  2 23
> smean.sd(x)
Error: could not find function "smean.sd"

> library(Hmisc)
> x
 [1]  2  4  5  6  4  5  6  7  8  9 12 23
> smean.sd(x)
    Mean        SD
7.583333 5.517877
```

Functions for calculating summary statistics of vector elements

```
summary():    Summary statistics of each column; type of
                  statistics depends on data type
apply() :     Apply a function to each column, works best
                   if all columns are the same data type
tapply():     Divide the data into subsets and apply
              a function to each subset, returns an array
by() :  Similar to tapply(), return an object of class by
ave():  Similar to tapply(), returns a vector the same
                  length as the argument vector
aggregate():  Similar to tapply(), returns a dataframe
sweep() :     Sweep "out" a summary statistic from a
                  dataframe, matrix or array
```

Boxplot

A boxplot is a way of summarizing a set of data measured on an interval scale. It is often used in exploratory data analysis. It is a type of graph which is used to show the shape of the distribution, its central value, and variability. The picture produced consist five number summaries. The median for each dataset is indicated by center line, and the first and third quartiles are the edges of the box. The extreme values (within 1.5 times the inter-quartile range from the upper or lower quartile) are the ends of the lines extending from the IQR. Points at a greater distance from the median than 1.5 times the IQR are plotted individually as asterisks. These points represent potential outliers.

```
>x=c(24,58,61,67,71,73,76,79,82,83,85,87,88,88,92,93,94,97)
>boxplot(x, main="Boxplot of test scores", col=2)
> arrows(1,24,1.2,30)
> text(1.4,31,"This is an Outlier")
```

Example:

Link below provides the number of Atlantic hurricane from 1870 to 2010

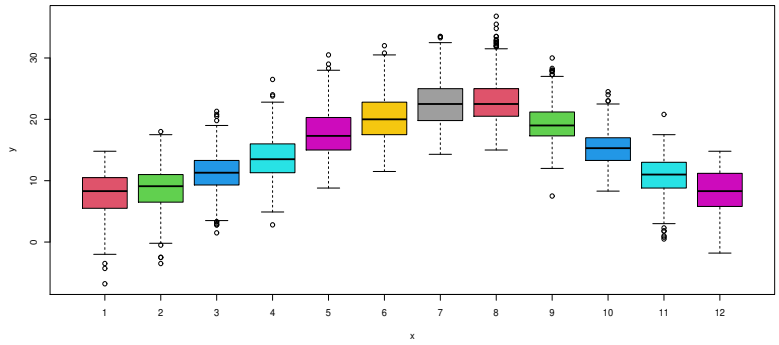
http://www.stat.purdue.edu/ zhanghao/MAS/dataset/SilwoodWeather.txt

We will import the subject data and plot a boxplot for monthly data

```
>temperature="http://www.stat.purdue.edu/~zhanghao/MAS/dataset/SilwoodWeather.txt"
>weather=read.table(temperature,header=T)
>attach(weather)
> names(weather)
[1] "upper" "lower" "rain" "month" "yr"
```

Before we can plot the data we need to declare month to be a factor. At the moment, R just thinks it is a number.

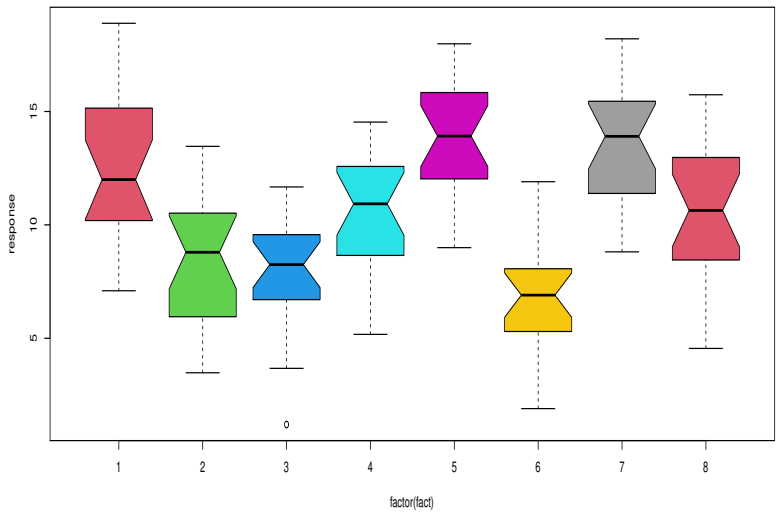
```
>month<-factor(month)
>plot(month,upper, col=c(2,3,4,5,6,7,8,10,11,12,13,14))
```



Side by side Box plot-Example

When there are many levels of a categorical explanatory variable, we need to be cautious about the statistical issues involved with multiple comparisons.The data show the response of yield to a categorical variable (fact) with eight levels representing eight different genotypes of seed (cultivars) used in the trial:

```
> data=read.table("C://Aryal//box.txt", header=T)
> attach(data)
> plot(response~factor(fact), notch=T, col=c((2:8),10))
```



Box plot

Box Plot

A boxplot is a convenient way of graphically depicting groups of numerical data through their quartiles.

R code below is used to draw a box plot.

```
> boxplot(x)
> boxplot.stats(x)$out
```

ggplot-syntex

```
Noun:  Data
     - ggplot(data = df)

Verb: ‘‘geom_" + Plot Type
 - ggplot(data = df) + geom_bar()

Adjectives:  Aesthetics (‘‘aes") (x, y, fill, colour, linetype)
     - ggplot(data = df,aes(x=categorical.var, fill=group.var))
     + geom_bar()

Adverb: stat (e.g. identity, bin)
     -ggplot(data = df, aes(x=categorical.var, fill=group.var))
       + geom_bar(stat = "bin")

Preposition :  position (e.g. fill, dodge, identity)
    -ggplot(data = df, aes(x=categorical.var, fill=group.var)) +
        geom_bar(stat="bin", position = "identity", binwidth=5)
```

Using ggplot2 package

Consider students data available in the Brightspace

```
students = read.csv("students.csv")
str(students)#structure of the data
dim(students)
nrow(students)
ncol(students)
students$Brothers
students$Sisters
with(students, Brothers + Sisters)
students[1:5, c(1, 6, 7)]
```

Categorical Variables are easier using with function

```
data(students)
with(students, table(Sex))
with(students, table(Sex, BloodType))
students = read.csv("students.csv", na.strings=c("","NA"))
with(students, table(Sex, BloodType))
```

Bar Graph

```
library(ggplot2)
data(students)
ggplot(students, aes(x = Level)) + geom_bar()
with(students, table(Sex, Level))
ggplot(students, aes(x = Level, fill = Sex)) + geom_bar()
ggplot(students, aes(x = Level, fill = Sex))
+ geom_bar(position = "dodge")
```

ggplot2-Quantitative Variable

```
library(ggplot2)
data(students)
with(students, fivenum(Height))
ggplot(students, aes(x = Height)) + geom_dotplot()
ggplot(students, aes(x=Height))+geom_histogram(binwidth=2)
ggplot(students, aes(x = Height))
           +geom_histogram(binwidth =2,  col=2, fill=3)
ggplot(students, aes(x = Height)) + geom_density()
ggplot(students, aes(x = Height))+ geom_density(adjust= 4)
ggplot(students, aes(x = Sex, y = Height))+ geom_boxplot()
ggplot(students, aes(x =Sex, y = Height))
              + geom_boxplot(col=c(2,3), fill=c(3,4))
ggplot(students, aes(x = Sex, y=Height))
                    + geom_boxplot() + coord_flip()
```

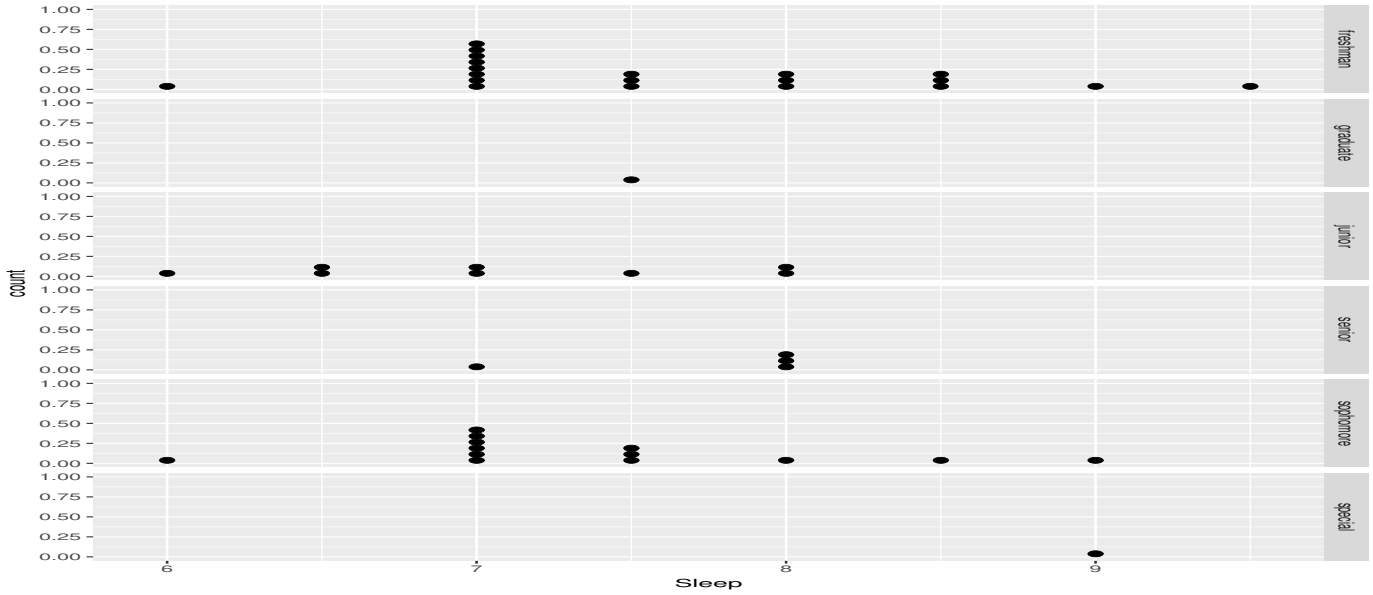
Quantitative Summary

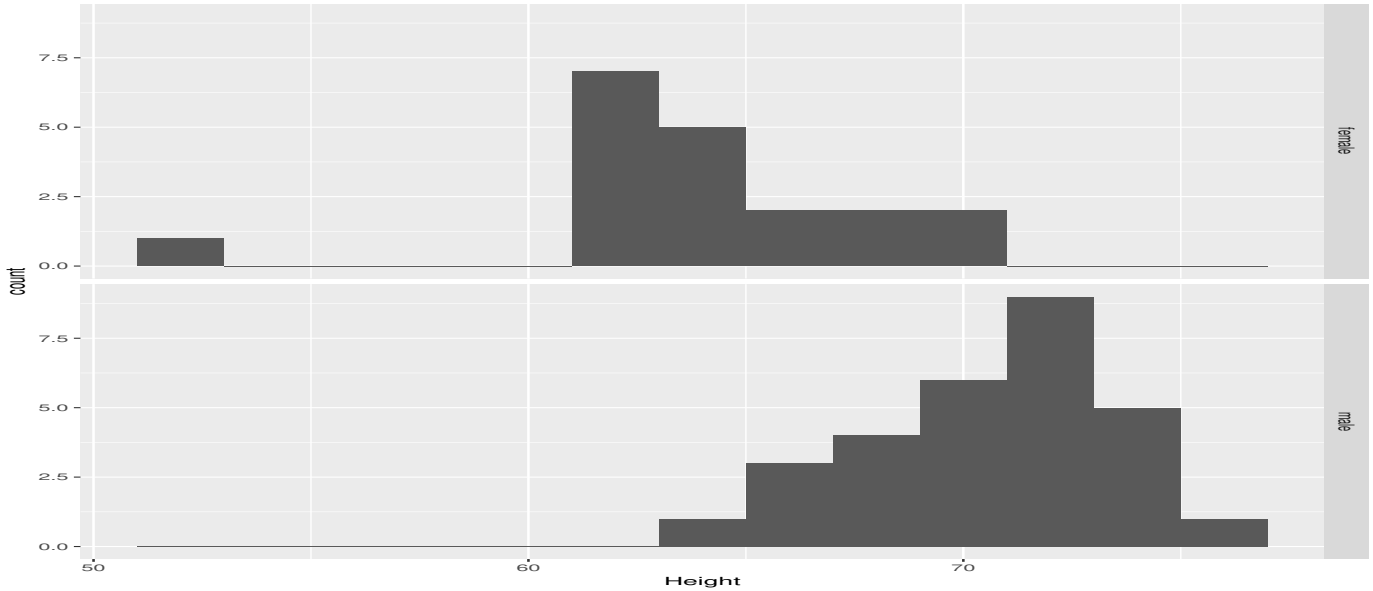
```
data(students)
with(students, mean(Height))
with(students, by(Height, Sex, mean))
with(students, median(Height))
with(students, by(Height, Sex, median))
with(students, fivenum(Height))
with(students, mean(Height[Sex == "female"]))
with(students, median(Height[Sex == "female"]))
with(students, sd(Height[Sex == "female"]))
```

Quantitative Data

```
library(ggplot2)
data(students)
ggplot(students, aes(x = Sleep)) + geom_dotplot(dotsize = 0.4)
ggplot(students, aes(x = Sleep)) + geom_dotplot(dotsize = 0.4)
+ facet_grid(Level ~.)
with(students, by(Sleep, Level, mean, na.rm = TRUE))
ggplot(students, aes(x = Height))+geom_histogram(binwidth = 2)
+ facet_grid(Sex ~.)
```

Some plots


count
Sleep
freshman
graduate
junior
senior
sophomore
special


count
Height
female
male

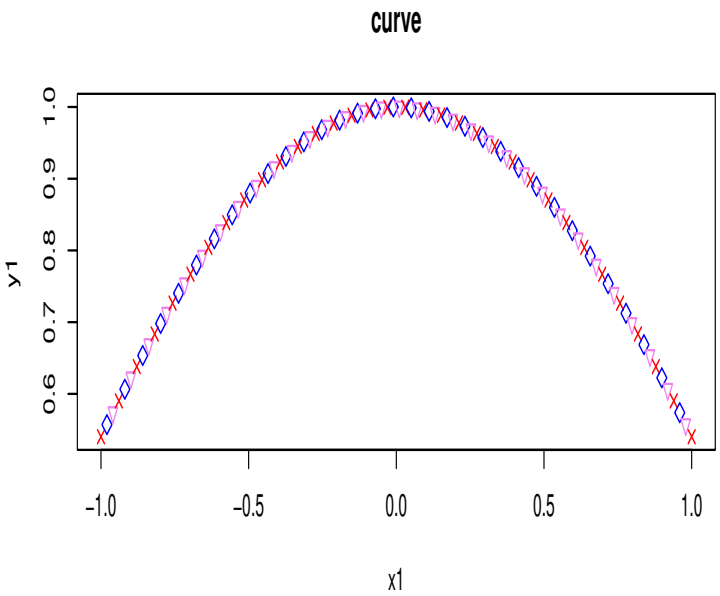
More Graphs

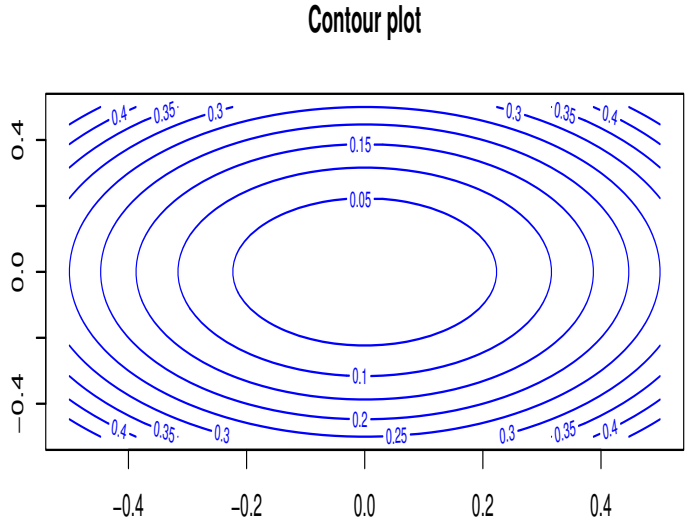
```
> x1 = seq(-2*pi, 2*pi, length = 100)
> y1 = cos(x1)
> plot(x1, y1, pch = 19, col= "red")

> par(mfrow=c(2,2))
> #plot2: With some customization
> plot(x1,y1,pch=c(4,5,6),col=c(2,3,4), main="curve")
> # Contour Plot
> y = x = seq(-pi, pi, length = 50)
> f = outer(x,y, function(x,y) (x^2 + y^2))
> contour(x, y, f, col = 'blue', main = "Contour plot")
> # image plot
> image(x, y, f, main = "Image plot")

> #3D surface plot
> persp(x,y,f,theta=45,col="blue",main="3D surface plot")
```

Some plots


curve
y1
x1
1.0
0.9
0.8
0.7
0.6
−1.0
−0.5
0.0
0.5
1.0


Contour plot
0.4
0.35
0.3
0.15
0.05
0.1
0.2
0.25
0.4
0.0
−0.4
−0.4
−0.2
0.0
0.2
0.4

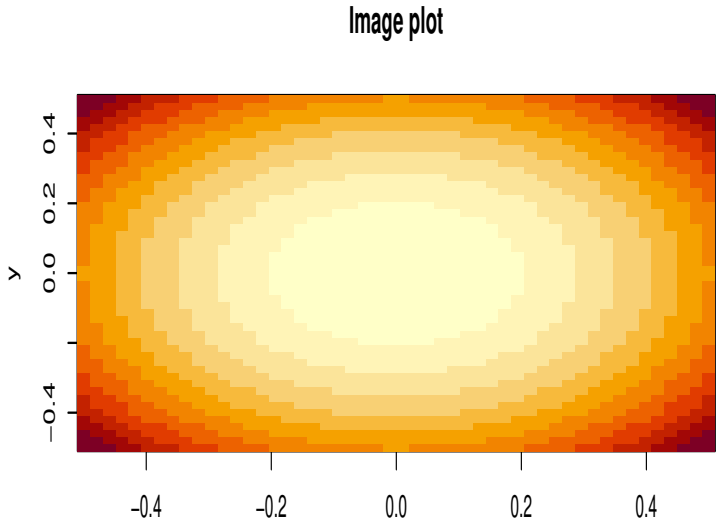
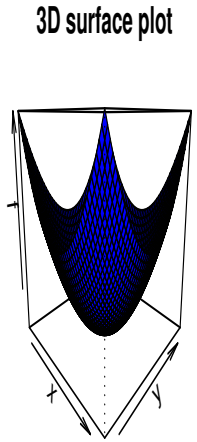


Image plot
y
0.4
0.2
0.0
−0.4
−0.4
−0.2
0.0
0.2
0.4


3D surface plot
x
y

Piedonut

Example: The "acs" data included in package "moonBook" is demographic and laboratory data of 857 patients with acute coronary syndrome(ACS).

```
library(ggplot2)
library(moonBook)
library(webr)
data(acs)
PieDonut(acs,aes(pies=Dx,donuts=smoking))
PieDonut(acs,aes(Dx,smoking),explode=1)
PieDonut(acs,aes(Dx,smoking),explode=1,explodeDonut=TRUE)
```