R 統計軟體

軟體使用入門

教學大綱

- R軟體發展歷史
- 主要特色及功能
- 基礎操作
  - 操作介面
  - 搜尋幫助
  - 物件介紹及操作
  - 常用函數
  - 基礎繪圖
- 進階使用
  - 撰寫程式及函數
  - 基礎資料分析

R 統計軟體發展歷史

- R 統計軟體最初是由Ross Ihaka及Robert Gentleman兩人以統計分析及繪圖為目的，仿S語言的架構為基礎而發展出來的統計軟體，可視為改進版本的S語言。大部分的S語言程式碼可直接或稍做修改後就在R上面執行
- R屬於GNU計畫中的一個項目，目前是由 R Development Core Team維護及發展
- 目前R最新的版本為4.0.2版(2020年6月)
- https://cran.r-project.org/

R的特色及功能

- 有效的資料處理及存取能力
- 方便的矩陣操作與運算能力
- 完整而連貫的資料分析能力
- 強大的繪圖功能
- 簡單且發展完善的程式語言環境(S 語言)
- 免費

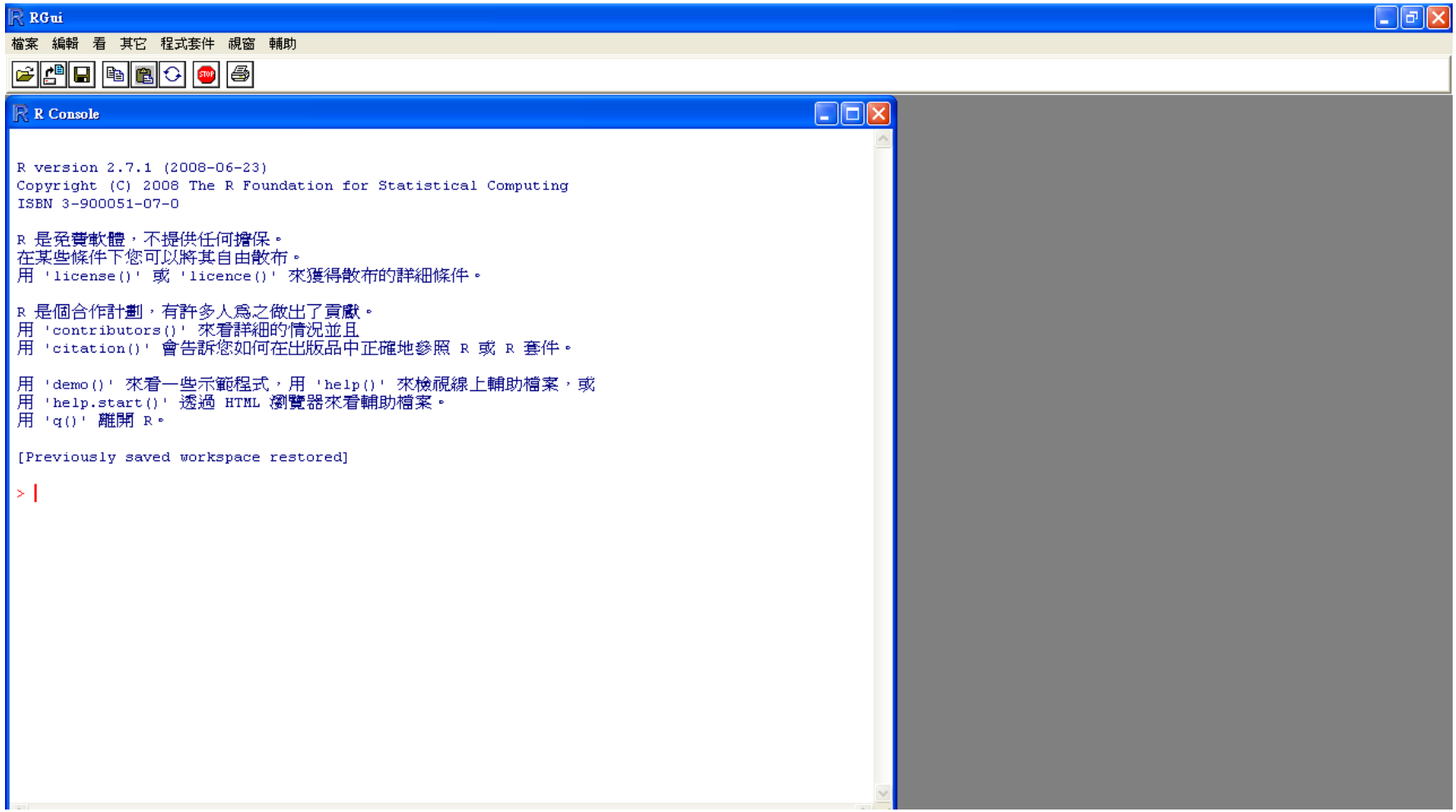
軟體下載

- Google 搜尋 “R” 第一個顯示即是R統計軟體網頁
- The R Project for Statistical Computing
- CRAN
- 選擇下載點:http://cran.csie.ntu.edu.tw/
  - Windows →base→Download R 4.0.2 for Windows
  - Mac OS X → R-4.0.2.pkg

- RStudio
  - https://www.rstudio.com/products/rstudio/download/#download

- 當R 啟動時, 有7 個常用之packages 會自
- 動載入:
- - base: 基本函式(IO, 敘述統計, etc.)
- - stats: 常用統計分析(t.test, anova, etc.)
- - methods: 定義classes of objects
- - utils: 基本程式編寫工具
- - graphics: 基本繪圖工具
- - grDevices:基本繪圖介面
- - datasets: 數據範例

基礎操作


RGui
檔案 編輯 看 其它 程式套件 視窗 輔助
R Console
R version 2.7.1 (2008-06-23)
Copyright (C) 2008 The R Foundation for Statistical Computing
ISBN 3-900051-07-0
R 是免費軟體，不提供任何擔保。
在某些條件下您可以將其自由散布。
用 'license()' 或 'licence()' 來獲得散布的詳細條件。
R 是個合作計劃，有許多人爲之做出了貢獻。
用 'contributors()' 來看詳細的情況並且
用 'citation()' 會告訴您如何在出版品中正確地參照 R 或 R 套件。
用 'demo()' 來看一些示範程式，用 'help()' 來檢視線上輔助檔案，或
用 'help.start()' 透過 HTML 瀏覽器來看輔助檔案。
用 'q()' 離開 R。
[Previously saved workspace restored]
>

SETUP THE ENVIRONMENT

- getwd() # Get currently used directory
- setwd("path") # Set up current directory

基礎操作

```
>(10+40)/2+3
28
>10^50/10^30
1e+20
>y<-1/sqrt(2*pi)*exp(-1/2)
>y
0.2419707
>sigma<-1
>mu<-0
>x<-2
>1/(sqrt(2*pi)* sigma)*exp(-((x - mu)^2/(2*sigma^2)))
0.05399097
>x<-rnorm(n=32,mean=80,sd=10)
      #產生32個來自平均值為80標準偏差為10的常態分布的隨機數
> x
x+5 #向量x中所有的數值+5
x
hist(x)   #畫x的直方圖
?Syntax  #查詢R基本術語
```

操作介面

- 編寫程式:「檔案」→「建立新的命令稿」 或直接於「>」後編寫
- 空一行或用分號 「 ; 」 將指令分開
- 套用已寫好之程式: 「檔案」→「開啟命令稿件」
- 修改或繼續編寫程式: 「檔案」→「開啟命令稿件」
- 程式套件(package)載入: 「程式套件」→「載入程式套件」
- 清理視窗: 右鍵→「清除視窗」
- 「←」、「→」或「=」表輸入
- 前面已執行完的指令:「↑」逐一顯示
- +: 程式未完結就換行會顯示「 + 」提醒，欲結束按「Esc」
- 英文字母大小寫視為**不同**的符號
- # 井字號之後為註解，程式不會執行
- 結束R程式: 直接關閉或指令「q()」

搜尋幫助

- 「輔助」→「Html輔助」= help.start()
- 「輔助」→「R函式」= help() 及 ?
- 「輔助」→「搜尋輔助」= help.search()

Description: brief description

Usage: for a function, details each of its arguments and the possible options (with the corresponding default values); for an operator gives the typical use.

Argument: for a function, details each of its arguments.

Details: detailed description

Value: if applicable, theh type of object returned by the function or the operator.

See Also: other help pages close or similar to the present one

Examples: some examples which can generally be executed without opening the help with the function **example**

R 工作流程

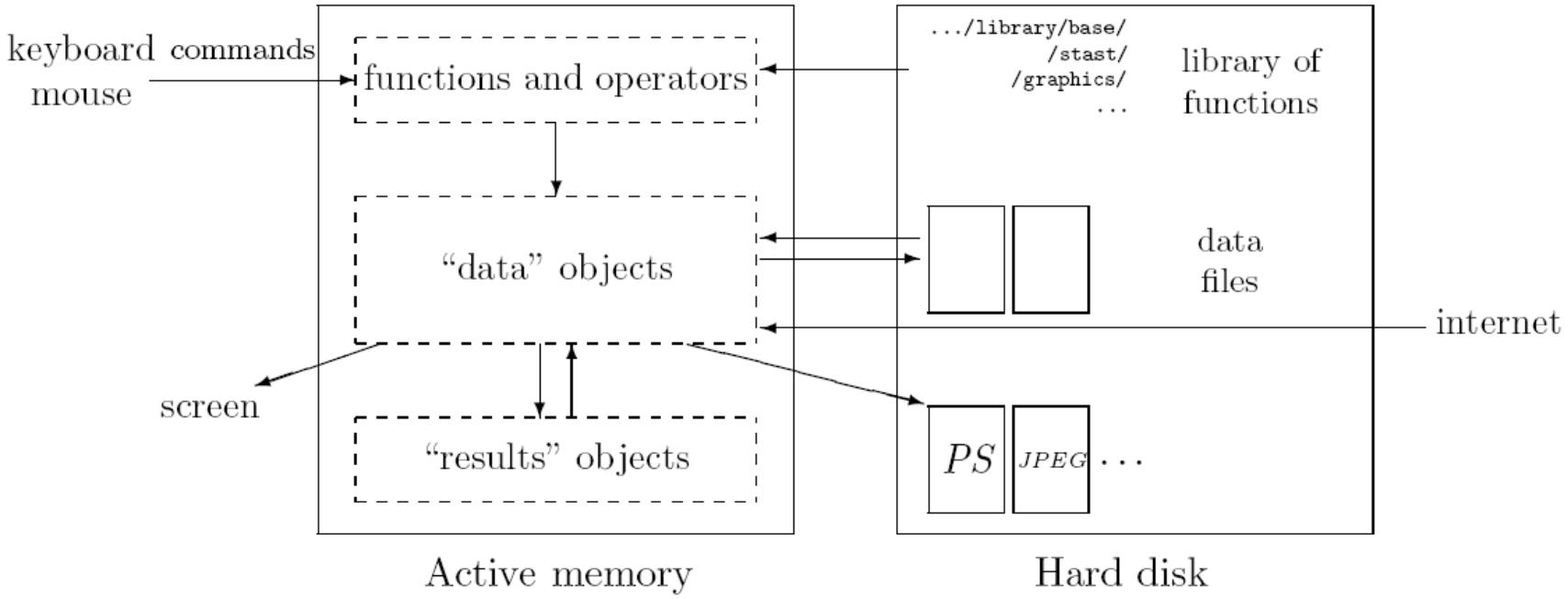


Figure 1: A schematic view of how R works.

變數(variable),資料(data),函數(function),結果(result)等在R程式運行時皆以物件(object)的形式存於電腦記憶體中。我們可以通過運算子(operators)或函數(function)對物件做操作。

物件 (OBJECTS)

object	Modes	several modes possible in the same object
vector	numeric, character, complex or logical	No
factor	numeric or character	No
array matrix	numeric, character, complex or logical	No
matrix	numeric, character, complex or logical	No
data frame	numeric, character, complex or logical	Yes
ts	numeric, character, complex or logical	No
list	function, expression, …	Yes

- 所有的物件(objects)都有兩種基本屬性(intrinsic attributes): 格式(mode)與長度(length)

```
a<-1 ; b<-"sec"; c<-1i ;d<-"TRUE"
mode(a);mode(b);mode(c);mode(d)
ls()     #列出所有物件
rm(a)  #清除物件a
```

運算子(OPERATORS)

Arithmetic		Comparison		Logical	
+	addition	<	lesser than	!x	logical NOT
-	subtraction	>	greater than	x&y	logical AND
*	multiplication	<=	lesser than or equal to	x&&y	id.
/	division	>=	greater than or equal to	xly	logical OR
^	power	==	equal	xlly	id.
%%	modulo	!=	different	xor(x,y)	exclusive OR
%/%	integer division				

```
x<-matrix(1:6,2,3) #製造一個2*3的矩陣x，其數值為1到6
x
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
x[2,3]==6  # x矩陣第2row第3column的值是否等於6
x[x<=3]    # 列出x矩陣內小於或等於3的數值
x[x!=6]    # 列出x矩陣內不等於6的數值
x[x<=3 & x!=2]  #列出x矩陣內小於或等於3且不等於2的值
```

函數(FUNCTION)

- function.name(object, argument, option)
  函數名稱　　物件　　指令　　選項
  #args(function.name)　查詢該函數的指令
- 數學及簡單函數
  sum(),mean(),max(),length()
- 產生隨機變數
  rnorm(),runiform(),rbinom()
- 初統常用分析函數
  t.test(),aov(),lm()

資料格式

1. 向量（vector） => 一維且有序的同屬性元素

```
> c(1,2,3)
[1] 1 2 3
```

2. 陣列（array） => 多維度的向量

```
> array(c(1,2,3),c(1,3))
     [,1] [,2] [,3]
[1,]    1    2    3
```

3. 矩陣（matrix） => nxn 的陣列稱為矩陣

```
> matrix(c(1:4),nrow=2,ncol=2)
     [,1] [,2]
[1,]    1    3
[2,]    2    4
```

4. 列表（list） => 一維且有序的多屬性元素

```
> list(a=1,b=TRUE,c="test",d=c(1,2,3))
$a
[1] 1
$b
[1] TRUE
$c
[1] "test"
$d
[1] 1 2 3
```

5. 框架（frame） => 可以用來存放像是資料庫的表格格式

```
> data.frame(c(1,2,3),c(2,3,4),c(3,4,5))
  c.1..2..3. c.2..3..4. c.3..4..5.
1          1          2          3
2          2          3          4
3          3          4          5
```

物件_序列(VECTOR)

- n1:n2, seq(), c(), rep(), sequence()

```
s1<-1:10 ; s1   #產生一個1到10的序列;

seq(from=1,to=5,by=0.5)  #產生一個序列從1到5間隔為0.5的序列;
seq(from=1,to=5,length=5)  #產生一個序列從1到5長度為5的序列;

s2<-c(1,3,5); s2  #產生數值序列1,3,5
s3<-c("a","b","c"); s3 #產生文字序列1,3,5

rep(1,10)   #產生數值1重複10次的序列;
rep("M",10) #產生文字序列 重複"M" 10次

sequence(c(3,5))  #產生1到3接連1到5的序列;
```

產生隨機序列

3.4.2 Random sequences

law	function
Gaussian (normal)	`rnorm(n, mean=0, sd=1)`
exponential	`rexp(n, rate=1)`
gamma	`rgamma(n, shape, scale=1)`
Poisson	`rpois(n, lambda)`
Weibull	`rweibull(n, shape, scale=1)`
Cauchy	`rcauchy(n, location=0, scale=1)`
beta	`rbeta(n, shape1, shape2)`
'Student' (t)	`rt(n, df)`
Fisher–Snedecor (F)	`rf(n, df1, df2)`
Pearson (χ^2)	`rchisq(n, df)`
binomial	`rbinom(n, size, prob)`
multinomial	`rmultinom(n, size, prob)`
geometric	`rgeom(n, prob)`
hypergeometric	`rhyper(nn, m, n, k)`
logistic	`rlogis(n, location=0, scale=1)`
lognormal	`rlnorm(n, meanlog=0, sdlog=1)`
negative binomial	`rnbinom(n, size, prob)`
uniform	`runif(n, min=0, max=1)`
Wilcoxon's statistics	`rwilcox(nn, m, n)`, `rsignrank(nn, n)`

```
dnorm(x, mean = 0, sd = 1, log = FALSE)
pnorm(q, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
qnorm(p, mean = 0, sd = 1, lower.tail = TRUE, log.p = FALSE)
rnorm(n, mean = 0, sd = 1)
```

rnorm() → 產生常態分布的隨機變數

dnorm() → probability density

pnorm() → cumulative probability function

qnorm() → the value of quantile

```
rnorm(n=30,mean=0,sd=1)
dnorm(1)== 1/sqrt(2*pi)*exp(-1/2)
pnorm(1.645, mean=0,sd=1)
qnorm(0.95,mean=0,sd=1)
```

$$f(x) = 1/(\sqrt{(2\,\pi)}\,\sigma)\; e^{-((x-\mu)^2/(2\,\sigma^2))}$$

物件_因子(FACTOR)

```
f1<-factor(1:3); f1;
f2<-factor(1:3,level=1:5); f2; #產生三個因子1,2,3 有五個等級
f3<-factor(rep(1:3,5)); f3;
f4<-factor(c(3,5),level=1:5); f4

gl(3, 5)  #產生一組factor, 有3個等級，每個等級重複5次;
gl(3,5,label=c("a","b","c") ) # 同上，另將此三個等級分別命名為
"a","b","c";
gl(3,5,length=30)
gl(2,10)
gl(2,1,length=20)  #區分不同
expand.grid(h=c(60,80),w=c(100,300),sex=c("Male","Female"))
```

物件_資料框(DATA.FRAME)

- 從Excel建立資料→檔案→存成.csv檔

male	female
176	156
168	162
175	157
181	163
177	170
165	161
172	154
170	155
173	162
169	167
186	163
163	160
175	162
174	159
169	158
170	163
172	160
176	161
172	158
171	160

物件_資料框(DATA.FRAME)

- 輸入外部資料(.txt檔或.csv檔)
- read.table()
- read.csv()　#預設讀取.csv檔

```
test1<-read.table("test.csv", header=T, sep=",")
#讀取C:\test.csv檔案，有標題，分隔符號為 “,”
test2<-read.csv("test.csv", header=T , col.names=c("M","F"))
#讀取C:\test.csv檔案，有標題，將column 1 ,2分別命名為 “M”及 “F”
```

- 外部輸入資料為data.frame物件

物件_資料框(DATA.FRAME)

- data.frame() 自行產生資料框物件

```
x<-1:4 ; n<-10; M<-c(10,35); y<-2:4
data.frame(x,n)
data.frame(x,M)
data.frame(x,y)
z<-c("a","b","c","d")
data.frame(x,n,row.names=z)
```

資料輸入

- scan() 逐行讀入資料
  - 讀取外部資料

```
data2<-scan("test.csv", sep=", " ,skip=1); data2
```

  - 直接輸入資料

```
data3<-scan()
1 2 3 4 5
6 7 8 9 10

data3
```

儲存資料框物件

- 將資料存成.txt或.csv檔
- write.table()

```
write.table(test2,file="test2.csv", sep=",")
#輸出物件test資料框物件到 test2.csv

write.table(test2,file="test.dat",sep=" ",row.names=FALSE,
col.names=FALSE);
#輸出物件test資料框物件到 test2.csv
```

物件_矩陣(MATRIX)

- 產生矩陣

```
m1<-matrix(1,nr=2,nc=3); m1
m2<-matrix(c(1,2,3,4,5,6),nr=2,nc=3); m2
m3<-matrix(c(1,2,3,4,5,6),2,3,byrow=T);m3
m4<-c(1,2,3,4,5,6); dim(m4)
dim(m4)<-c(2,3); m4
```

- 矩陣操作

```
cbind(m1,m2)
rbind(m1,m2)
m2[,2] ; m2[2,2]
m5<-matrix(c(2,0,0,2),2,2);
m6<-solve(m5); m6  # inverse of m5
m5%*%m6                # matrix multiplication
diag(m5); diag(m5)<-3; m5
```

物件_EXPRESSION

- Expression 為一連串對R有意義的文字所組成

```
> x<-3; y<-2.5; z<-1
> exp1<-expression(x/(y+exp(z)))
> exp1
expression(x/(y+exp(z)))
> eval(exp1)
0.5749019
> D(exp1, "x")  #對x偏微分
1/(y + exp(z))
> D(exp1, "y") #對x偏微分
-(x/(y + exp(z))^2)
```

物件操作

- [ ] index
- :: access variables in a name space
- @ $ component / slot extraction
- attach()
- names()

```
test1<-read.table("c:/test.csv", header=T, sep=",");
test1[1,] ; test1[,2];
test1[,2, drop=F ]
test1[-1,]
test1[test1>=170]
```

練習

- 產生2組長度為10的隨機序列，然後將此兩個序列合併成為1*2的矩陣
- 模擬1組電腦選號的樂透號碼
- 將2008奧運比賽台灣棒球隊的打擊成績輸入R
- 輸入後更改陳金鋒的姓名為 “不動的第四棒”
- 列出打擊率為零的球員，再將其更改為0.01
- 將更改後的資料框輸出成.csv檔

常用函數

sum(x)	sum of the elements of **x**
prod(x)	product of the elements of **x**
max(x)	maximum of the elements of **x**
min(x)	minimum of the elements of **x**
which.max(x)	returns the index of the greatest element of **x**
which.min(x)	returns the index of the smallest element of x
range(x)	in. than c(min(**x**),max(**x**))
length(x)	number of elements of **x**
mean(x)	mean of the elements of **x**
median(x)	median of the elements of **x**
var(x) or cov(x)	variance of the elements of **x** (calculated on n-1);if **x** is a matrix or a data frame, the variance-covariance matrix is calculated
cor(x)	correlation matrix of x if it is a matrix or a data frame(1 if **x** is a vector)
var(x,y) or cov(x,y)	covariance between **x** and **y**, or between the columns of **x** and those of **y** if they are matrices or data frames
cor(x,y)	linear correlation between **x** and **y**, or correlation matrix if they are matrices or data frames

常用函數

round(x,n)	rounds the elements of **x** to n decimals
rev(x)	reverses the elements of **x**
sort(x)	sorts the elements of **x** in increasing order:rev(sort(**x**))
rank(x)	ranks of the elements of **x**
log(x,base)	computes the logarithm of x with base **base**
scale(x)	if **x** is matrix, centers and reduces the data; to center only use the option **center=FALSE**, to reduce only **scale=FALSE** (by default **center=TRUE, sacle=TRUE**)
pmin(x,y,⋯)	a vector which ith element is the minimum of **x[1], y[1],**⋯
pmax(x,y,⋯)	id. for the maximun
cumsum(x)	a vector which ith element is the sum from **x[1]** to **x[i]**
sumprod(x)	id. For the product
cummin(x)	id. For the minimun
cummax(x)	id. For the maximun
match(x,y,⋯)	returns a vector of the same length than **x** with the element of **x** which are in **y**(**NA** otherwise)
which(x==a)	return a vector of the indicies of x if the comparison operation is true (**TRUE**), in this example the values of I for which **x[i]**==a(the argument of this function must be a variable of mode logical

常用函數

choose(n,k)	computes the combinations of k event among n repetitions = n! / [(n-k)!k!]
na.omit(x)	suppresses the observations with missing data(NA) (suppresses the corresponding line if **x** is a matrix or a data frame)
na.fail(x)	returns an error message if **x** contains at least one NA
unique(x)	if **x** is vector or a data frame, return a similar object but with the duplicate elements suppressed
table(x)	returns a table with the numbers of the differents values of **x** (typically for integers or factors)
table(x,y)	contingency table of **x** and y
subset(x,…)	returns a selection of **x** with respect to criteria (…, typically comparison: **x$v1 < 10**); if **x** is a data frame, the option **select** gives the variables to be kept (or dropped using a minus sign)
sample(x,size)	resample randomly and without replacement **size** elements in the vector x, the option **replace = TRUE** allows to resample with replacement

- When the same or similar tasks need to be performed multiple times; for all elements of a list; for all columns of an array; etc.
  - Monte Carlo Simulation
  - Cross-Validation (delete one and etc)

```
for(i in 1:10) {
  print(i*i)
}

i=1
while(i<=10) {
  print(i*i)
  i=i+sqrt(i)
}
```

LOOPS

撰寫程式及函數

```
x<-numeric(10)
for(i in 1:10)
{
  x[i]<-i*rnorm(1,mean=0,sd=5)
}
x

for(i in 1:10)
{
  if (x[i]<0)
   {x[i]<-0}
 else
   {x[i]<-1}
}
fun<-function(x,mu,sigma)
{ 1/(sqrt(2*pi)* sigma)*exp(-((x - mu)^2/(2*sigma^2))) }
fun(1,0,1)
```

基礎資料分析

- 阿匾將錢匯往國外100次，且其中有20次匯往英屬維京群島，若檢方隨機抽查阿匾的10次匯款紀錄，則: 至少檢查到3筆匯往國外的紀錄的機率為?

```
binomial<-function(x,n,p)
{choose(n,x)*p^x*(1-p)^(n-x)}
p<-20/100
n<-10
sum(binomial(3:10,n,p))

binomial(2,10,0.2)
dbinom(2,10,0.2)
1-pbinom(2,10,0.2)
```

基礎資料分析

假設南韓人與臺灣人的平均壽命分別為75歲及80歲，標準偏差均為10， 今分別隨機挑出30名南韓人與台灣人(男女各半)，試作t-test 比較南韓人與台灣人平均壽命是否有差異?

```
T<-rnorm(30,mean=80,sd=10)
K<-rnorm(30,mean=75,sd=10)
sex<-c(rep("Man",15),rep("Female",15))
data1<-data.frame(sex,T,K)

ttest1<-t.test(T,K,alternative="two.sided",var.equal=T,conf.level=0.95)
print(ttest1)
ttest2<-t.test(T,K,alternative="greater",var.equal=T,conf.level=0.95)
print(ttest2)
ttest3<-t.test(T~sex,alternative="two.sided",var.equal=T,conf.level=0.95)
#注意 taiwan~sex 表示taiwan這向量內的值根據sex因子分類做t-test
print(ttest3)
```

基礎資料分析

- 迴歸分析

```
regression<-function(x,y)
{
 ex.lm<-lm(y~x)
 print(summary(ex.lm))
 print(anova(ex.lm))
 win.graph()
 plot(x,y)
 abline(lm(y~x))
 res<-ex.lm$residuals  #residual plot
 yhat<-predict(ex.lm)
 win.graph()
 plot(yhat,res,main="residuals against fit value")
 abline(h=0)
 win.graph()
 qqnorm(res)
 qqline(res)
}
x<-rnorm(20)
y<-2*x+rnorm(20)
regression(x,y)
```

LISTS

- vector: an ordered collection of data of the same type.

```
> a = c(7,5,1)
> a[2]
[1] 5
```

- list: an ordered collection of data of arbitrary types.

```
> doe = list(name="john",age=28,married=F)
> doe$name
[1] "john“
> doe$age
[1] 28
```

- Typically, vector elements are accessed by their index (an integer), list elements by their name (a character string). But both types support both access methods.

LAPPLY, SAPPLY, APPLY

- When the same or similar tasks need to be performed multiple times for all elements of a list or for all columns of an array.
  - May be easier and faster than “for” loops
- lapply(li, *function*)
  - To each element of the list li, the function *function* is applied.
  - The result is a list whose elements are the individual *function* results.

```
> li = list("klaus","martin","georg")
> lapply(li, toupper) # To upper case
> [[1]]
> [1] "KLAUS"
> [[2]]
> [1] "MARTIN"
> [[3]]
> [1] "GEORG"
```

APPLY

apply(arr, margin, fct)

Apply the function fct along some dimensions of the array arr, according to margin, and return a vector or array of the appropriate size.

```
> x
     [,1] [,2] [,3]
[1,]    5    7    0
[2,]    7    9    8
[3,]    4    6    7
[4,]    6    3    5
> apply(x, 1, sum)
[1] 12 24 17 14
> apply(x, 2, sum)
[1] 22 25 20
> apply(x, 1, sort)
     [,1] [,2] [,3] [,4]
[1,]    0    7    4    3
[2,]    5    8    6    5
[3,]    7    9    7    6
```

```
> apply(x, 2, sort)
     [,1] [,2] [,3]
[1,]    4    3    0
[2,]    5    6    5
[3,]    6    7    7
[4,]    7    9    8
```

LAPPLY, SAPPLY, APPLY

- sapply(li, fct)
  - Apply a function over a list or vector

```
> li = list("klaus","martin","georg")
> sapply(li, toupper)
[1] "KLAUS"  "MARTIN" "GEORG"

> fct = function(x) { return(c(x, x*x, x*x*x)) }
> sapply(1:5, fct)
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    1    4    9   16   25
[3,]    1    8   27   64  125
```

FUNCTIONS AND OPERATORS

Functions do things with data
“Input”: function arguments (0,1,2,…)
“Output”: function result (exactly one)

Example:

```
add = function(a,b)
{ result = a+b
  return(result) }
```

Operators:

Short-cut writing for frequently used functions of one or two arguments.
Examples: + - * / ! & | %%

FUNCTIONS AND OPERATORS

- Functions do things with data
  - "Input": function arguments (0,1,2,…)
  - "Output": function result (exactly one)

Exceptions to the rule:

- Functions may also use data that sits around in other places, not just in their argument list: "scoping rules"*
- Functions may also do other things than returning a result. E.g., plot something on the screen: "side effects"

* Lexical scope and Statistical Computing.
R. Gentleman, R. Ihaka, *Journal of Computational and Graphical Statistics*, 9(3), p. 491-508 (2000).

READING DATA FROM FILES

- The read.table() function
  - To read an entire data frame directly, the external file will normally have a special form.
  - The first line of the file should have a name for each variable in the data frame.
  - Each additional line of the file has its first item a row label and the values for each variable.

	Price	Floor	Area	Rooms	Age	Cent.heat
01	52.00	111.0	830	5	6.2	no
02	54.75	128.0	710	5	7.5	no
03	57.50	101.0	1000	5	4.2	no
04	57.50	131.0	690	6	8.8	no
05	59.75	93.0	900	5	1.9	yes
...						

- numeric variables and nonnumeric variables (factors)

READING DATA FROM FILES

- HousePrice <- read.table("houses.data", header=TRUE)

Price	Floor	Area	Rooms	Age	Cent.heat
52.00	111.0	830	5	6.2	no
54.75	128.0	710	5	7.5	no
57.50	101.0	1000	5	4.2	no
57.50	131.0	690	6	8.8	no
59.75	93.0	900	5	1.9	yes

- The data file is named ‘input.dat’.
  - Suppose the data vectors are of equal length and are to be read in in parallel.
  - Suppose that there are three vectors, the first of mode character and the remaining two of mode numeric.
- The scan() function
  - inp<- scan("test.dat", list("",0,0)) # scan file into list
  - To separate the data items into three separate vectors, use assignments like

    label <- inp[[1]]; x <- inp[[2]]; y <- inp[[3]]
  - inp <- scan("test.dat", list(id="", x=0, y=0)); inp$id; inp$x; inp$y

STORING DATA

- Every R object can be stored into and restored from a file with the commands “save” and “load”.
- This uses the XDR (external data representation) standard of Sun Microsystems and others, and is portable between MS-Windows, Unix, Mac.

```
>save(x, file="x.Rdata")
>load("x.Rdata")
```

基礎繪圖

- demo(graphics) #展示圖例
- High-level plotting function: 產生新的繪圖
- Low-level plotting function: 對已繪製完成的圖片增加點,線條或說明
- par(): 調整繪圖的參數

基礎繪圖

plot(x)	plot of the values of x (on the y-axis) ordered on the x-axis
plot(x,y)	bivariate plot of x (on the x-axis) and y (on the y-axis)
sunflowerplot(x,y)	id. but the points with similar coordinates are drawn as a flower which petal number represents the number of points
pie(x)	circular pie-chart
boxplot(x)	box-and-whiskers plot
stripchart(x)	plot of the values of x on a line (an alternative to boxplot() for small sample sizes)
coplot(x~y j z)	bivariate plot of x and y for each value (or interval of values) of z
interaction.plot(f1, f2, y)	if f1 and f2 are factors, plots the means of y (on the y-axis) with respect to the values of f1 (on the x-axis) and of f2 (different curves); the option fun allows to choose the summary statistic of y (by default fun=mean)
matplot(x,y)	bivariate plot of the first column of x vs. the first one of y, the second one of x vs. the second one of y, etc.
dotchart(x)	if x is a data frame, plots a Cleveland dot plot (stacked plots line-by-line and column-by-column)
fourfoldplot(x)	visualizes, with quarters of circles, the association between two dichotomous variables for different populations (x must be an array with dim=c(2, 2, k), or a matrix with dim=c(2, 2) if k = 1)
assocplot(x)	Cohen-Friendly graph showing the deviations from independence of rows and columns in a two dimensional contingency table

基礎繪圖_GRAPHICAL FUNCTION

mosaicplot(x)	'mosaic' graph of the residuals from a log-linear regression of a contingency table
pairs(x)	if x is a matrix or a data frame, draws all possible bivariate plots between the columns of x
plot.ts(x)	if x is an object of class "ts", plot of x with respect to time, x may be multivariate but the series must have the same frequency and dates
ts.plot(x)	id. but if x is multivariate the series may have different dates and must have the same frequency
hist(x)	histogram of the frequencies of x
barplot(x)	histogram of the values of x
qqnorm(x)	quantiles of x with respect to the values expected under a normal law
qqplot(x, y)	quantiles of y with respect to the quantiles of x

LATTICE GRAPHS

- *trellis graphs* - graphs that display a variable or the relationship between variables, conditioned on one or more other variables.
- *graph_type*(*formula*, data=)
- **~x|A**
  - Display numeric variable x for each level of factor A
- **y~x | A*B**
  - Display the relationship between numeric variables y and x separately for every combination of factor A and B levels.
- **~x**
  - Display numeric variable x alone.

GRAPH_TYPE(FORMULA, DATA=) (~X|A), (Y~X | A*B), (~X)

graph_type	description	formula examples
barchart	bar chart	x~A or A~x
bwplot	boxplot	x~A or A~x
cloud	3D scatterplot	z~x*y|A
contourplot	3D contour plot	z~x*y
densityplot	kernal density plot	~x|A*B
dotplot	dotplot	~x|A
histogram	histogram	~x
levelplot	3D level plot	z~y*x
parallel	parallel coordinates plot	data frame
splom	scatterplot matrix	data frame
stripplot	strip plots	A~x or x~A
xyplot	scatterplot	y~x|A
wireframe	3D wireframe graph	z~y*x

points(x, y)	adds points (the option type= can be used)
lines(x, y)	id. but with lines
text(x, y, labels,...)	adds text given by labels at coordinates (x,y); a typical use is:plot(x, y, type="n"); text(x, y, names)
mtext(text,side=3, line=0,...)	adds text given by text in the margin specified by side (see axis() below); line specifies the line from the plotting area
segments(x0, y0, x1, y1)	draws lines from points (x0,y0) to points (x1,y1)
arrows(x0, y0, x1, y1, angle= 30, code=2)	id. with arrows at points (x0,y0) if code=2, at points (x1,y1) if code=1, or both if code=3; angle controls the angle from the shaft of the arrow to the edge of the arrow head
abline(a,b)	draws a line of slope b and intercept a
abline(h=y)	draws a horizontal line at ordinate y
abline(v=x)	draws a vertical line at abcissa x
abline(lm.obj)	draws the regression line given by lm.obj (see section 5)
rect(x1, y1, x2,y2)	draws a rectangle which left, right, bottom, and top limits are x1, x2, y1, and y2, respectively

LOW-LEVEL PLOTTING COMMANDS

polygon(x, y)	draws a polygon linking the points with coordinates given by x and y
legend(x, y,legend)	adds the legend at the point (x,y) with the symbols given by legend
title()	adds a title and optionally a sub-title
axis(side, vect)	adds an axis at the bottom (side=1), on the left (2), at the top(3), or on the right (4); vect (optional) gives the abcissa (or ordinates) where tick-marks are drawn
box()	adds a box around the current plot
rug(x)	draws the data x on the x-axis as small vertical lines
locator(n, type="n", ...)	returns the coordinates (x; y) after the user has clicked n times on the plot with the mouse; also draws symbols (type="p") or lines (type="l") with respect to optional graphic parameters(...); by default nothing is drawn (type="n")

繪圖控制參數

adj	controls text justification with respect to the left border of the text so that 0 is left-justified, 0.5 is centred, 1 is right-justified, values > 1 move the text further to the left, and negative values further to the right; if two values are given (e.g., c(0, 0)) the second one controls vertical justification with respect to the text baseline
bg	specifies the colour of the background (e.g., bg="red", bg="blue"; the list of the 657 available colours is displayed with colors())
bty	controls the type of box drawn around the plot, allowed values are: "o", "l", "7", "c", "u" ou "]" (the box looks like the corresponding character); if bty="n" the box is not drawn
Cex	a value controlling the size of texts and symbols with respect to the default; the following parameters have the same control for numbers on the axes, cex.axis, the axis labels, cex.lab, the title, cex.main, and the sub-title, cex.sub
col	controls the colour of symbols; as for cex there are: col.axis, col.lab, col.main, col.sub
font	an integer which controls the style of text (1: normal, 2: italics, 3: bold, 4: bold italics); as for cex there are: font.axis, font.lab, font.main, font.sub
las	an integer which controls the orientation of the axis labels (0: parallel to the axes, 1: horizontal, 2: perpendicular to the axes, 3: vertical)

lty	controls the type of lines, can be an integer (1: solid, 2: dashed, 3: dotted, 4: dotdash, 5: longdash, 6: twodash), or a string of up to eight characters (between "0" and "9") which specifies alternatively the length, in points or pixels, of the drawn elements and the blanks, for example lty="44" will have the same effet than lty=2
lwd	a numeric which controls the width of lines
mar	a vector of 4 numeric values which control the space between the axes and the border of the graph of the form c(bottom, left, top, right), the default values are c(5.1, 4.1, 4.1, 2.1)
mfcol	a vector of the form c(nr,nc) which partitions the graphic window as a matrix of nr lines and nc columns, the plots are then drawn in columns (see section 4.1.2)
mfrow	id. but the plots are then drawn in line (see section 4.1.2)
pch	controls the type of symbol, either an integer between 1 and 25, or any single character within "" (Fig. 2)
ps	an integer which controls the size in points of texts and symbols
pty	a character which specifies the type of the plotting region, "s": square, "m": maximal
tck	a value which specifies the length of tick-marks on the axes as a fraction of the smallest of the width or height of the plot; if tck=1 a grid is drawn
tcl	id. but as a fraction of the height of a line of text (by default tcl=-0.5) xaxt if xaxt="n" the x-axis is set but not drawn (useful in conjunction with axis(side=1, ...))
yaxt	if yaxt="n" the y-axis is set but not drawn (useful in conjunction with axis(side=2, ...))

R GRAPHS

- Graphs
  - https://www.statmethods.net/graphs/index.html
- Lattice graphs
  - https://www.statmethods.net/advgraphs/index.html
- ggplots
  - http://www.sthda.com/english/wiki/qplot-quick-plot-with-ggplot2-r-software-and-data-visualization
  - http://www.sthda.com/english/wiki/ggplot2-essentials

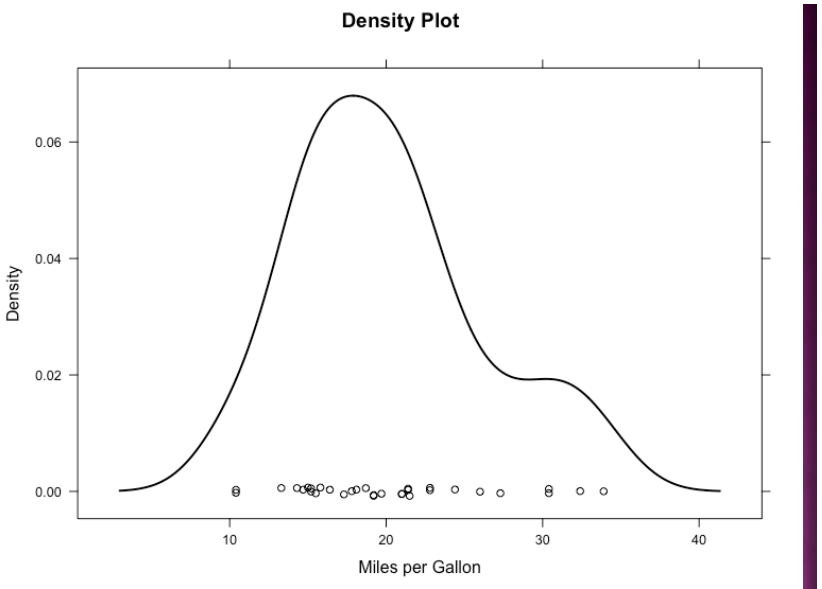
GGPLOT2

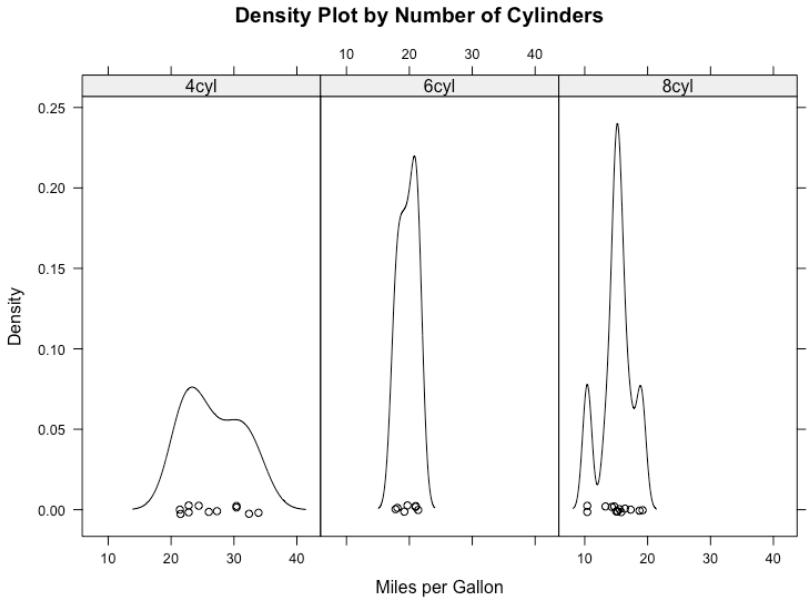
http://www.sthda.com/english/wiki/ggplot2-essentials

- Plot = data + Aesthetics + Geometry.
  - Data is a data frame
  - Aesthetics is used to indicate x and y variables. It can also be used to control the color, the size or the shape of points, the height of bars, etc.
  - Geometry defines the type of graphics (histogram, box plot, line plot, density plot, dot plot, ….)
- There are two major functions
  - qplot() stands for quick plot, which can be used to produce easily simple plots.
  - ggplot() function is more flexible and robust than qplot for building a plot piece by piece.

LATTICE PLOT EXAMPLES

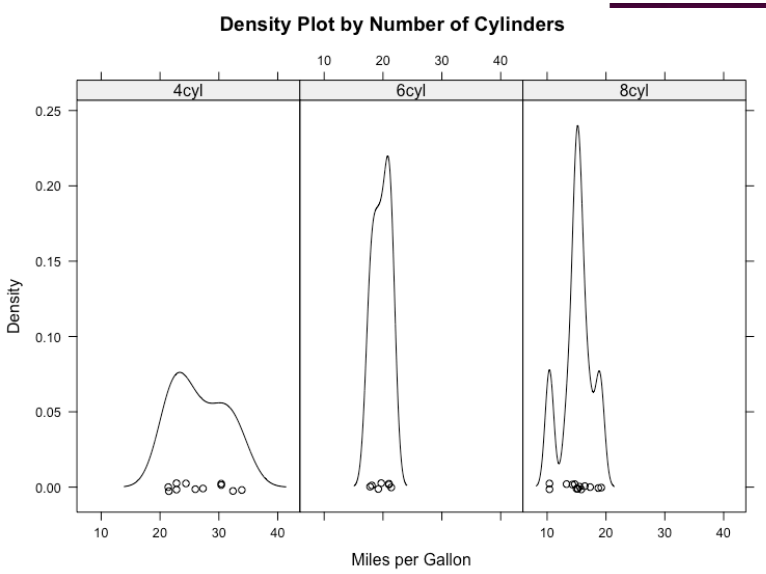
```
# Lattice Examples
library(lattice)
attach(mtcars)
# create factors with value labels
gear.f<-factor(gear,levels=c(3,4,5),
labels=c("3gears","4gears","5gears"))
cyl.f <-factor(cyl,levels=c(4,6,8),
labels=c("4cyl","6cyl","8cyl"))
# kernel density plot
densityplot(~mpg,main="Density Plot",
xlab="Miles per Gallon")
# kernel density plots by factor level
densityplot(~mpg|cyl.f, main="Density
Plot by Number of
Cylinders",xlab="Miles per Gallon")
```



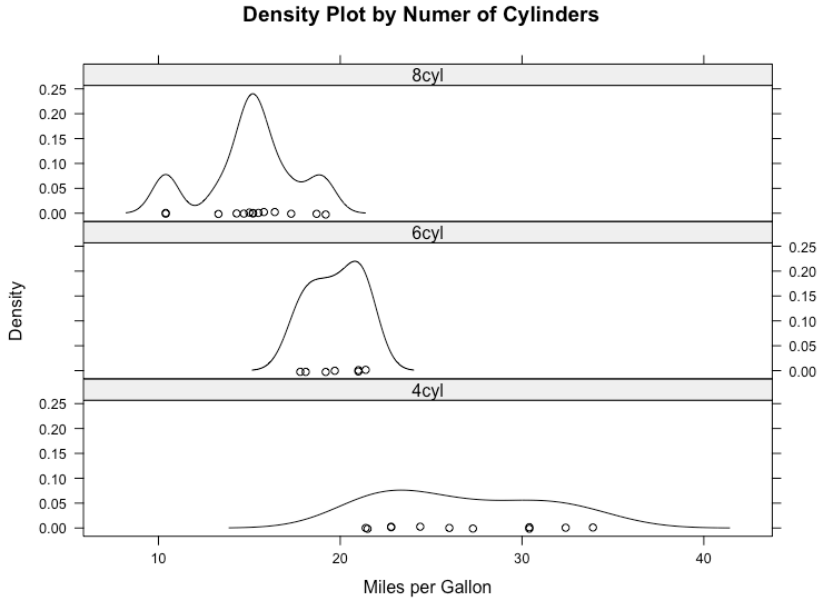


KERNEL DENSITY PLOTS BY FACTOR LEVEL

```
# kernel density plots by factor
level densityplot(~mpg|cyl.f,
main="Density Plot by Number of
Cylinders",xlab="Miles per
Gallon")
```



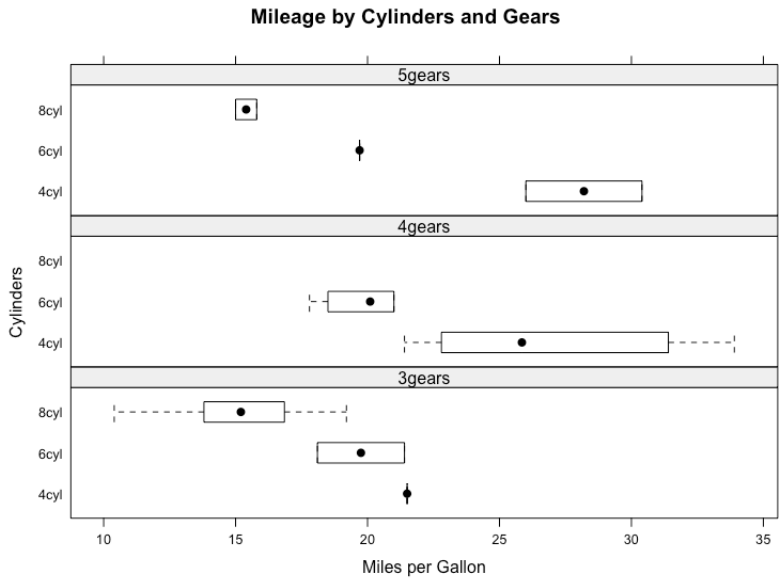

```
# kernel density plots by factor
level (alternate layout)
densityplot(~mpg|cyl.f,
main="Density Plot by Numer of
Cylinders", xlab="Miles per
Gallon",
layout=c(1,3))
```



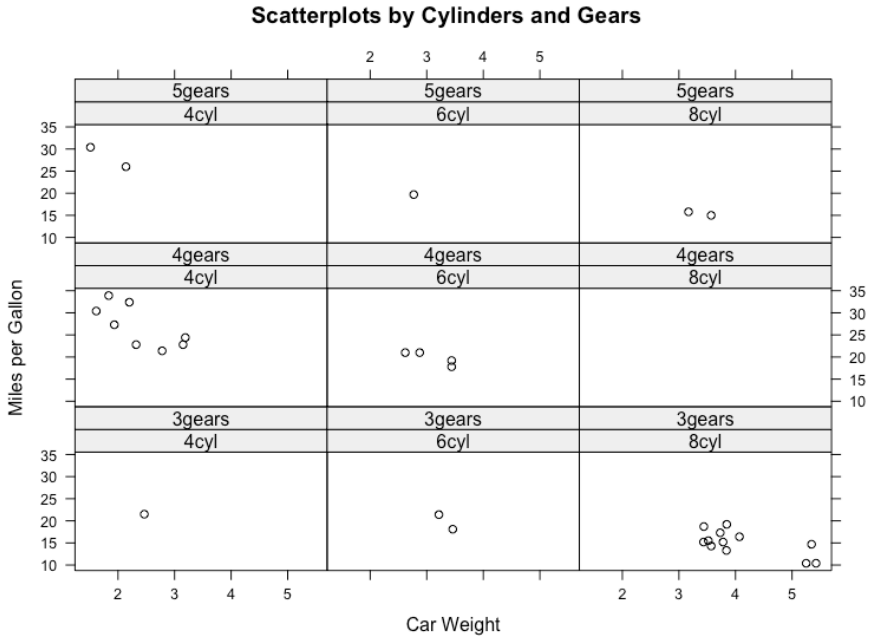
`layout=c(1,3))`

column-widths and the row-heights

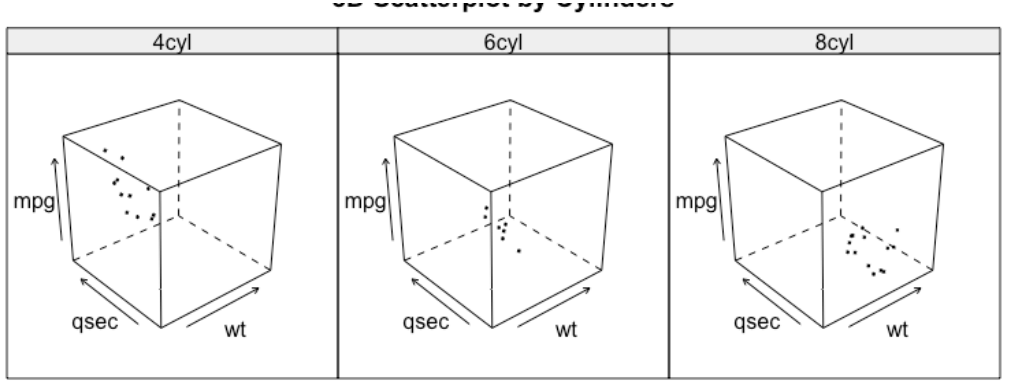
```
# boxplots for each combination of
two factors bwplot(cyl.f~mpg|gear.f,
ylab="Cylinders",          xlab="Miles
per Gallon",          main="Mileage by
Cylinders and Gears", layout=c(1,3))
```



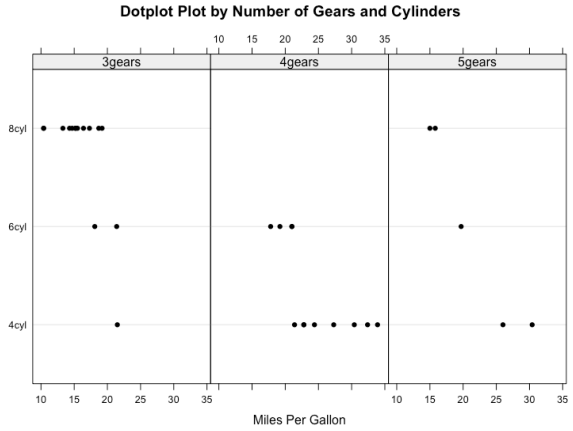

```
# scatterplots for each combination
of two factors
xyplot(mpg~wt|cyl.f*gear.f,
main="Scatterplots by Cylinders and
Gears",          ylab="Miles per
Gallon", xlab="Car Weight")
```



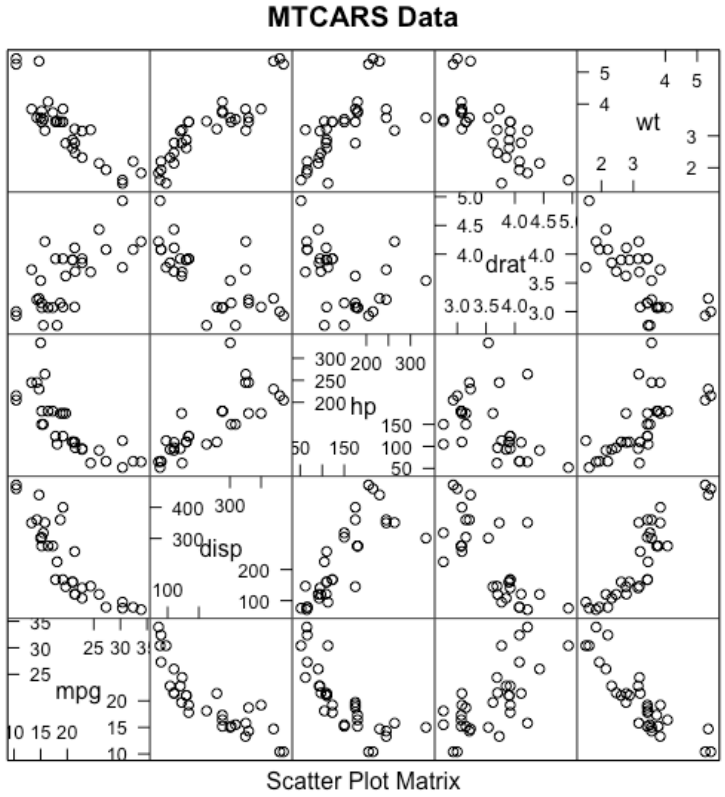
```r
 # 3d scatterplot by factor level
cloud(mpg~wt*qsec|cyl.f,main="3D
Scatterplot by Cylinders")
```




```r
# dotplot for each combination of two
factors
dotplot(cyl.f~mpg|gear.f,main="Dotplot
Plot by Number of Gears and
Cylinders",xlab="Miles Per Gallon")
```




```r
# scatterplot matrix
splom(mtcars[c(1,3,4,5,6)],main="MTCARS
Data")
```



EXPORT PLOTS

- **File formats for exporting plots:**
- **pdf("rplot.pdf")**: pdf file
- **png("rplot.png")**: png file
- **jpeg("rplot.jpg")**: jpeg file
- **postscript("rplot.ps")**: postscript file
- **bmp("rplot.bmp")**: bmp file
- **win.metafile("rplot.wmf")**: windows metafile

CREATE AND SAVING GRAPHS

```
# 1. Open a pdf file
pdf("rplot.pdf")
# 2. Create a plot
plot(x = my_data$wt, y = my_data$mpg, pch = 16, frame = FALSE, xlab = "wt", ylab = "mpg", col = "#2E9FDF")
# Close the pdf file
dev.off()
# 1. Open jpeg file
jpeg("rplot.jpg", width = 350, height = "350")
# 2. Create the plot
plot(x = my_data$wt, y = my_data$mpg, pch = 16, frame = FALSE, xlab = "wt", ylab = "mpg", col = "#2E9FDF")
# 3. Close the file
dev.off()
# Create an eps plot
setEPS()
postscript("whatever.eps")
plot(rnorm(100), main="Hey Some Data")
dev.off()
```

CHART TYPES

- Single variable
  - Dot plot
  - Jitter plot
  - Error bar plot
  - Box-and-whisker plot
  - Histogram
  - Kernel density estimate
  - Cumulative distribution function

(note: examples using qplot library from R)

CHART TYPES

- **Dot plot**

```
> f500.ca <- subset(f500, state_location == "CA")
> f500.ca$state_location <- factor(f500.ca$state_location)
> qplot(revenues, state_location, data=f500.ca)
```

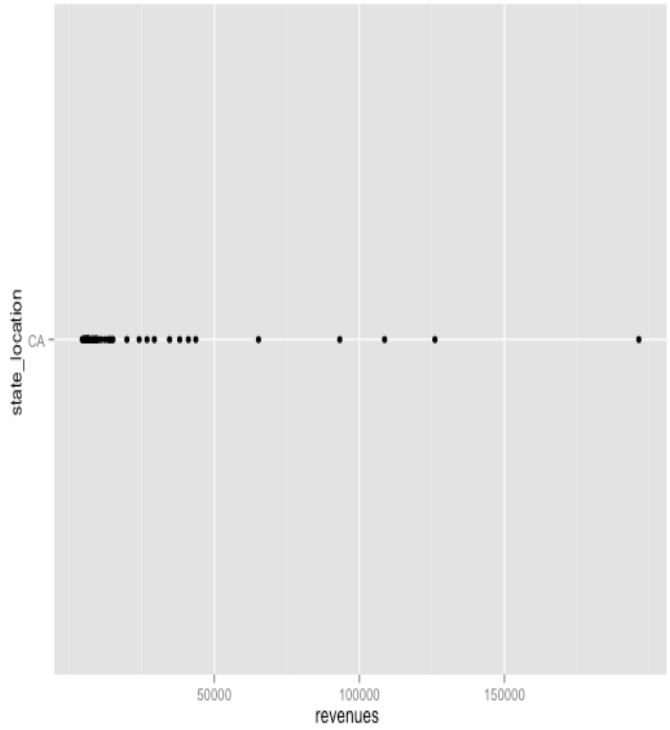


CHART TYPES

- **Jitter plot**

```
> qplot(revenues, state_location, data=f500.ca, geom="jitter")
```

- Noise added to the y-axis to spread the points

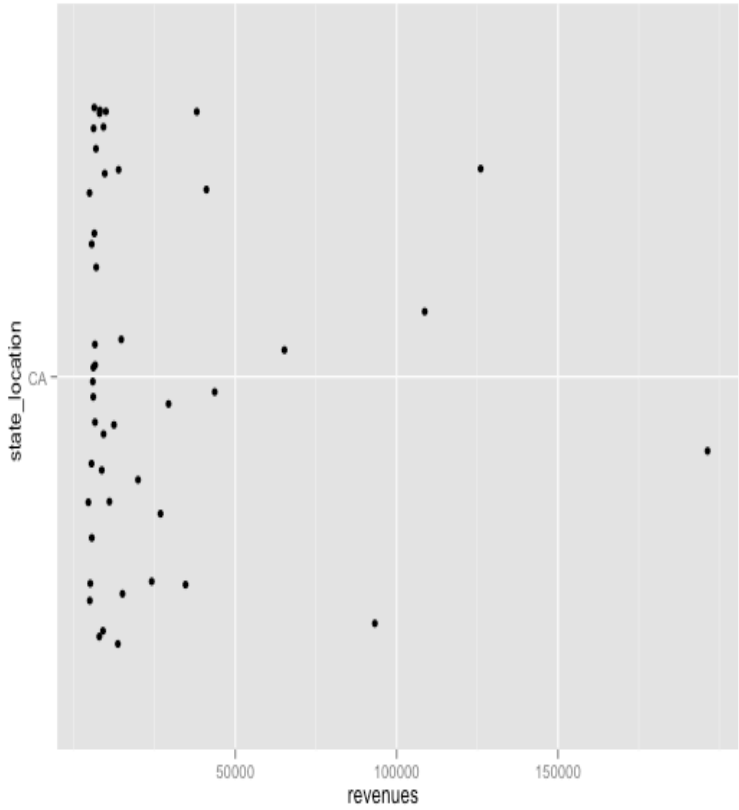


CHART TYPES

- Error bars: usually based on confidence intervals (CI). 95% CI means 95% of points are in the range, so 2.5% of points are above or below the bar.
- Not necessarily symmetric:

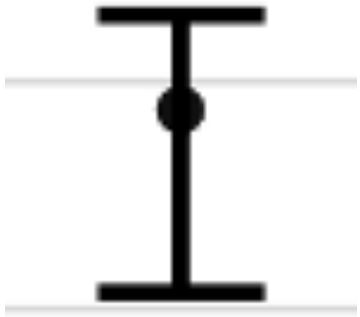


CHART TYPES

- **Box-and-whisker plot** : a graphical form of 5-number summary (Tukey)

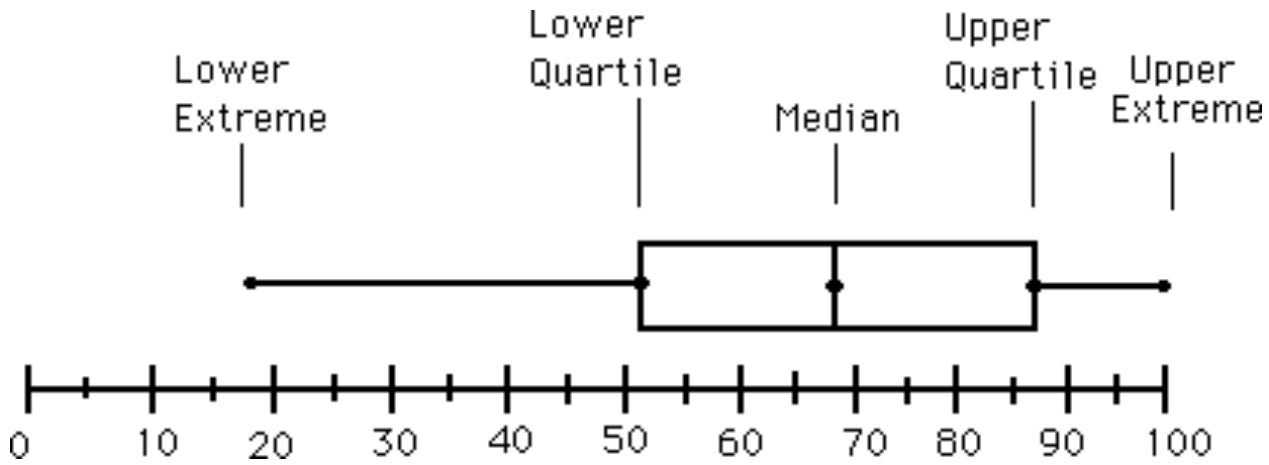


CHART TYPES

- **Histogram**

```
> qplot(revenues, data=f500.ca, geom="histogram")
stat_bin: binwidth defaulted to range/30. Use 'binwidth = x' to adjust this.
```

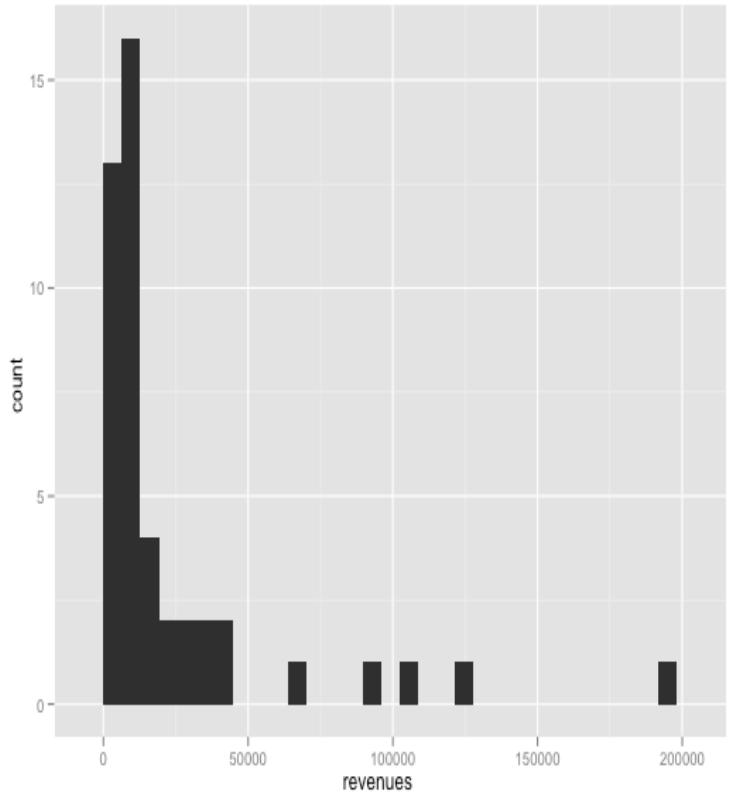


CHART TYPES

- **Kernel density estimate**

```
> qplot(revenues, data=f500.ca, geom="density")
```

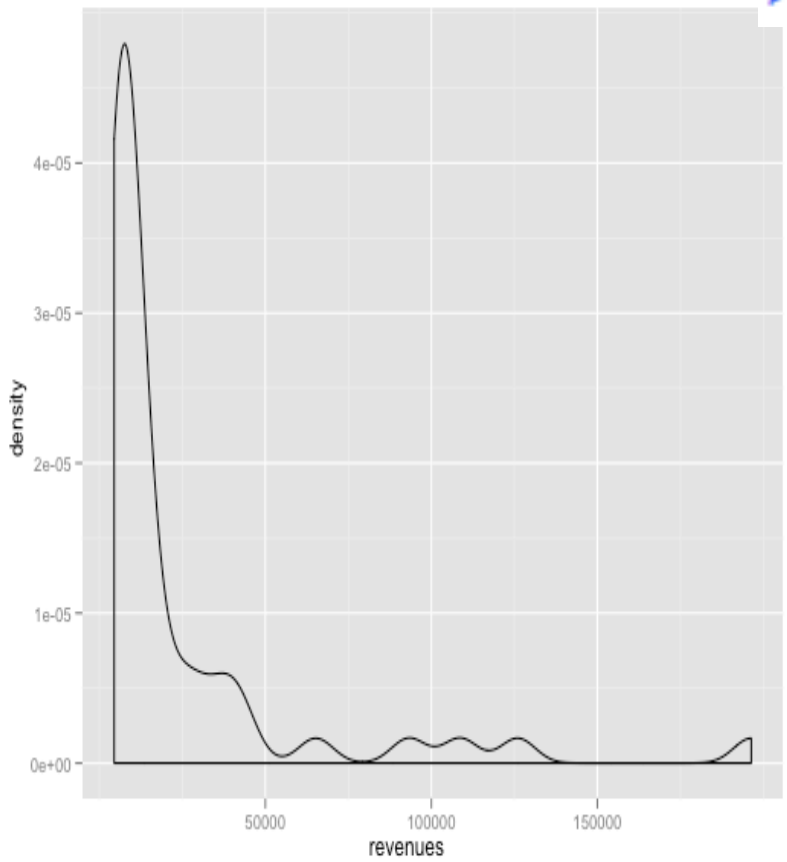


CHART TYPES

- Histogram and Kernel Density Estimates
  - Histogram
    - Proper selection of bin width is important
    - Outliers should be discarded
  - KDE (like a smooth histogram)
    - Kernel function
      - Box, Epanechnikov, Gaussian
    - Kernel bandwidth

CHART TYPES

```
> plot(ecdf(f500.ca$revenues))
```

- **Cumulative distribution function**
- Integral of the histogram – simpler to build than KDE (don't need smoc

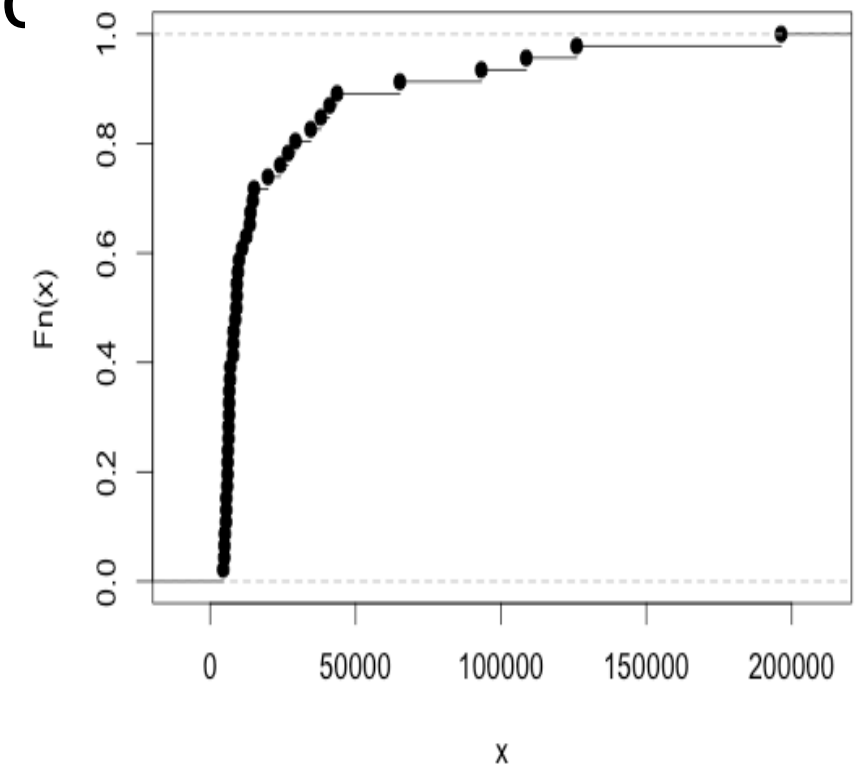


CHART TYPES

- Two variables
  - Bar chart
  - Scatter plot
  - Line plot
  - Log-log plot

CHART TYPES

- **Bar plot:** one variable is discrete

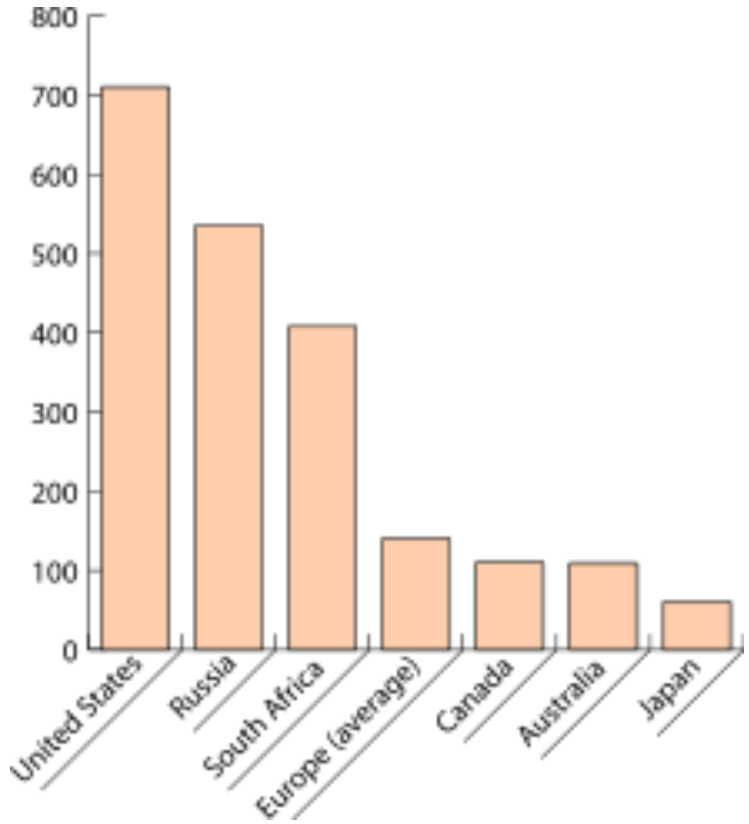


CHART TYPES

- **Scatter plot**

```
> qplot(revenues, profits, data=f500)
```

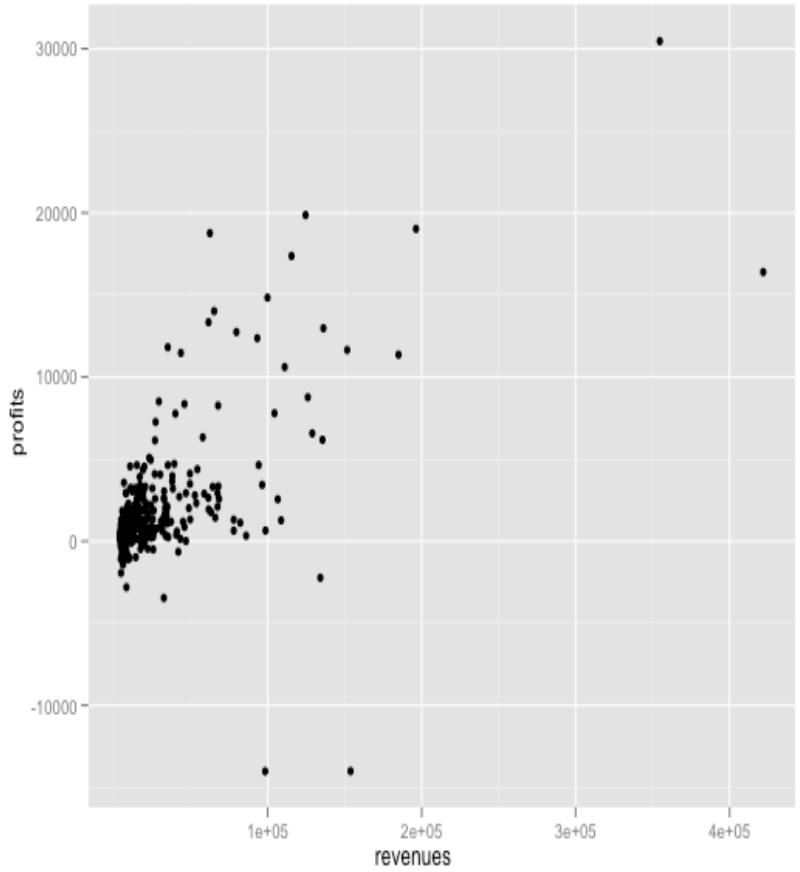


CHART TYPES

- **Line plot**

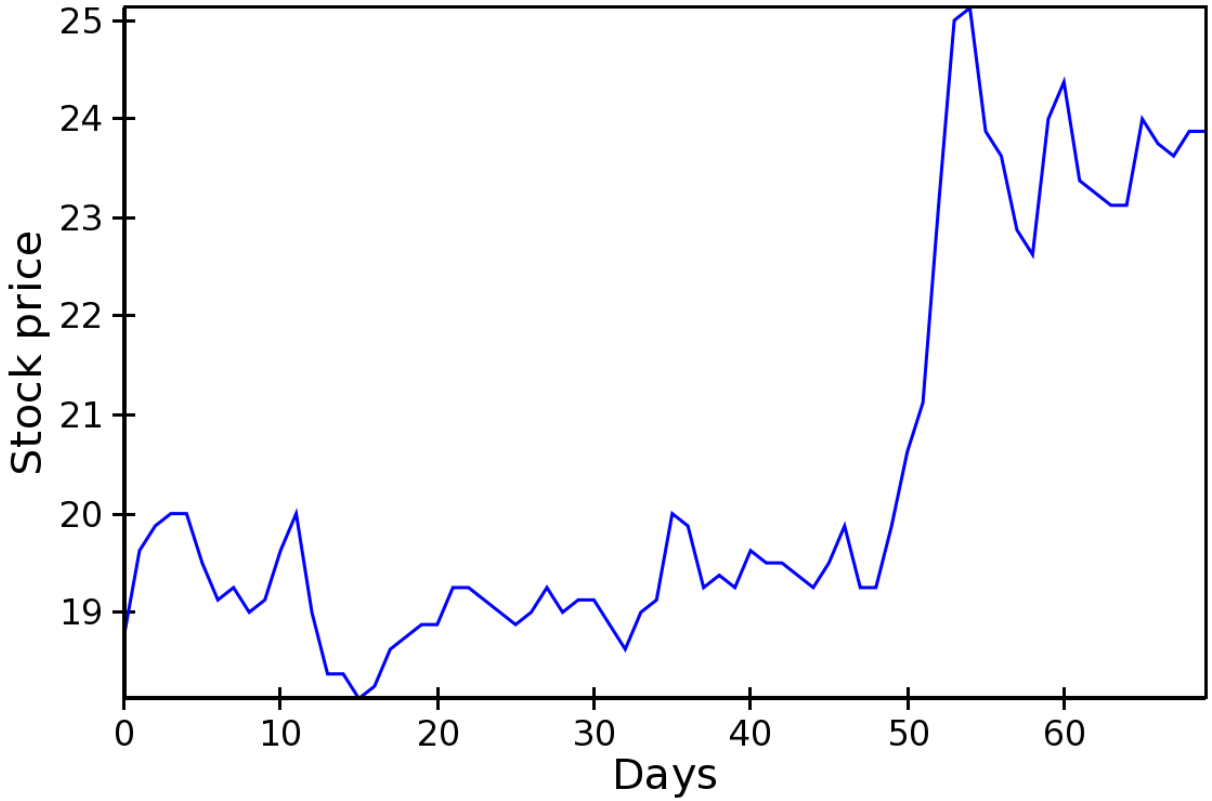
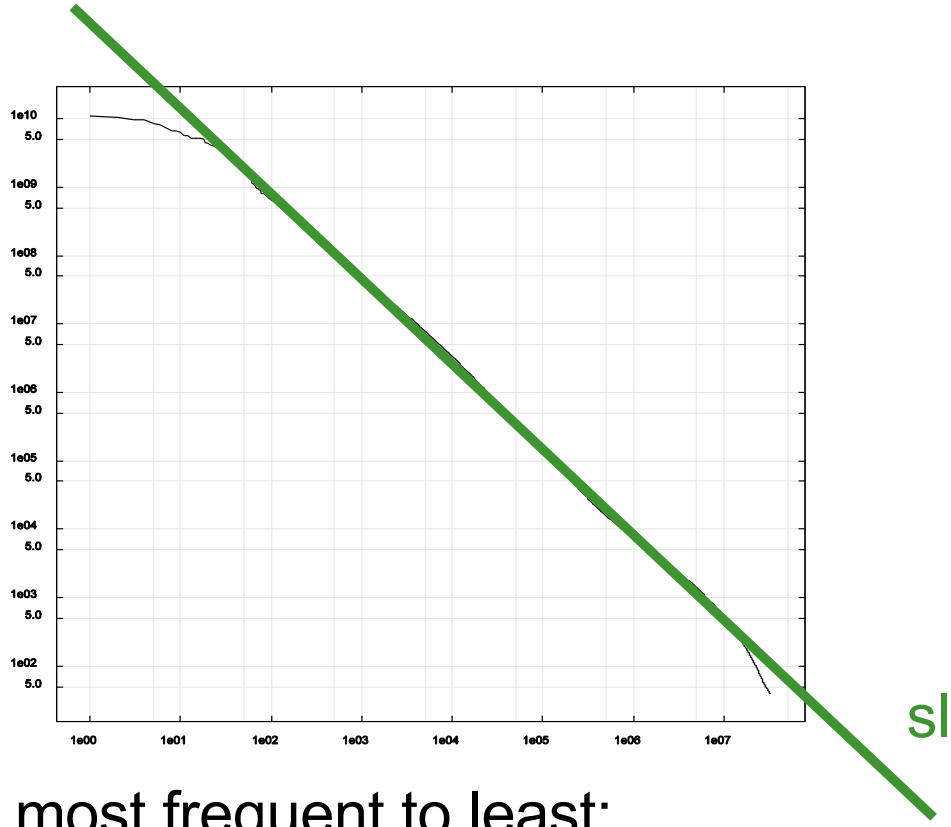


CHART TYPES

- **Log-log plot:** Very useful for power law data

Frequency of words in tweets



slope ~ -1

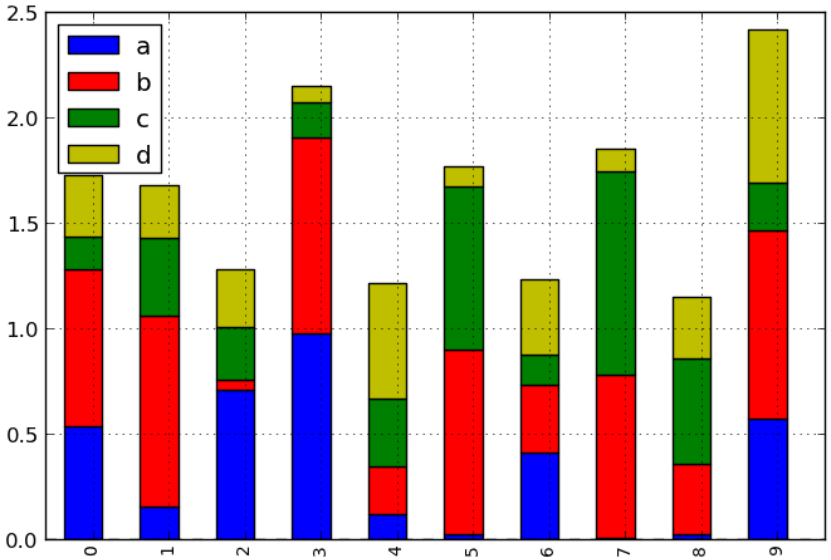
Rank of words in tweets, most frequent to least:
I, the, you,…

CHART TYPES

- More than two variables
  - Stacked plots
  - Parallel coordinate plot

CHART TYPES

- **Stacked plot:** stack variable is discrete:



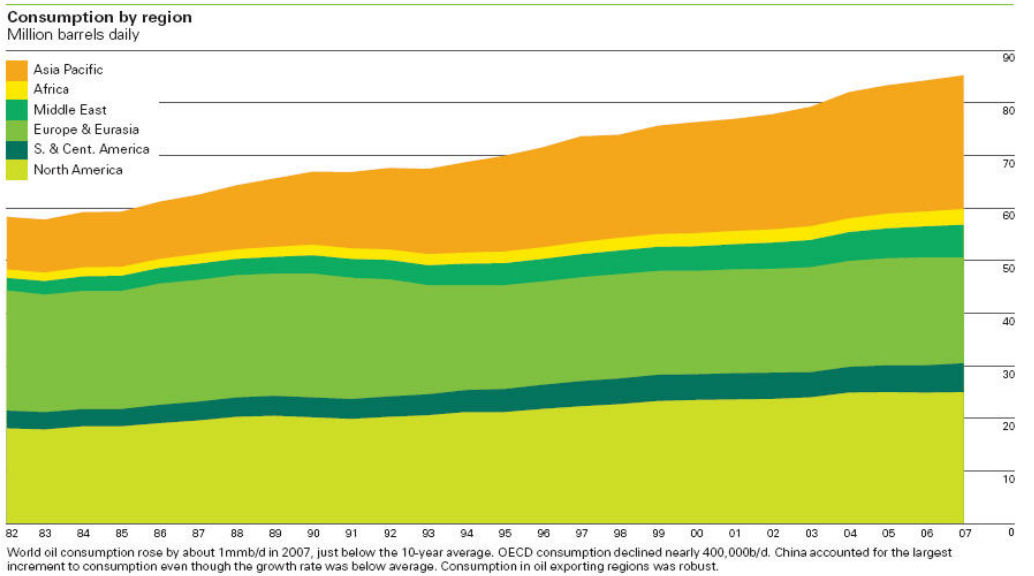
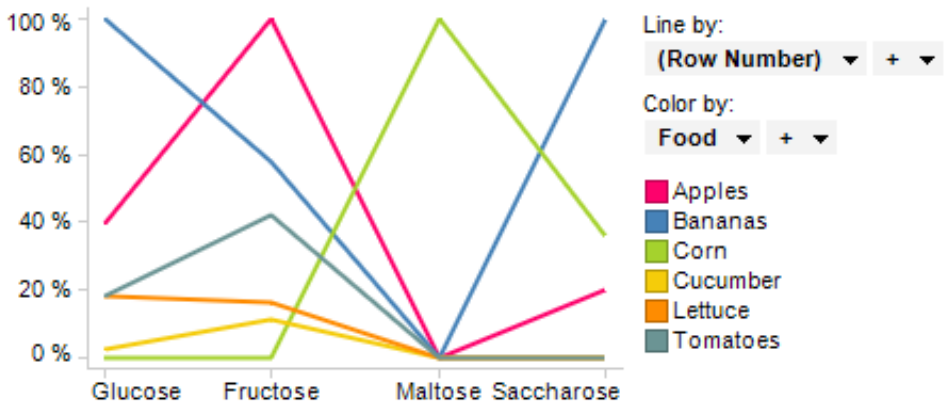
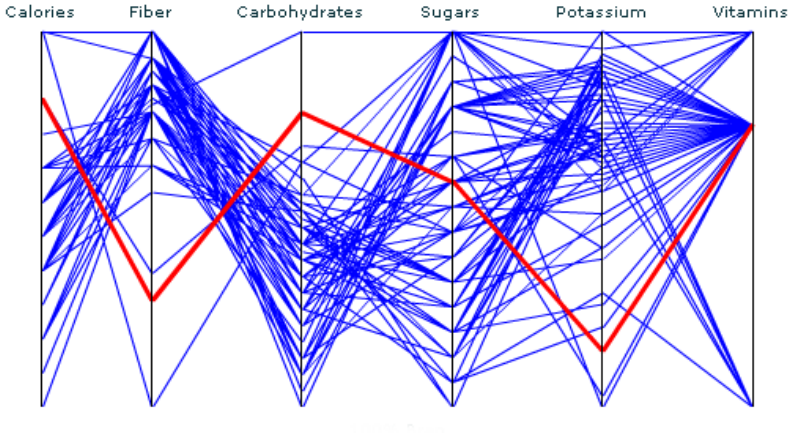


CHART TYPES

- **Parallel coordinate plot:** one discrete variable, an arbitrary number of other variables:





IMPORTING DATA

- How do we get data into R?
- Remember we have no point and click…
- First make sure your data is in an easy to read format such as CSV (Comma Separated Values).
- Use code:
  - `D <- read.csv("path",sep=",",header=TRUE)`

WORKING WITH DATA.

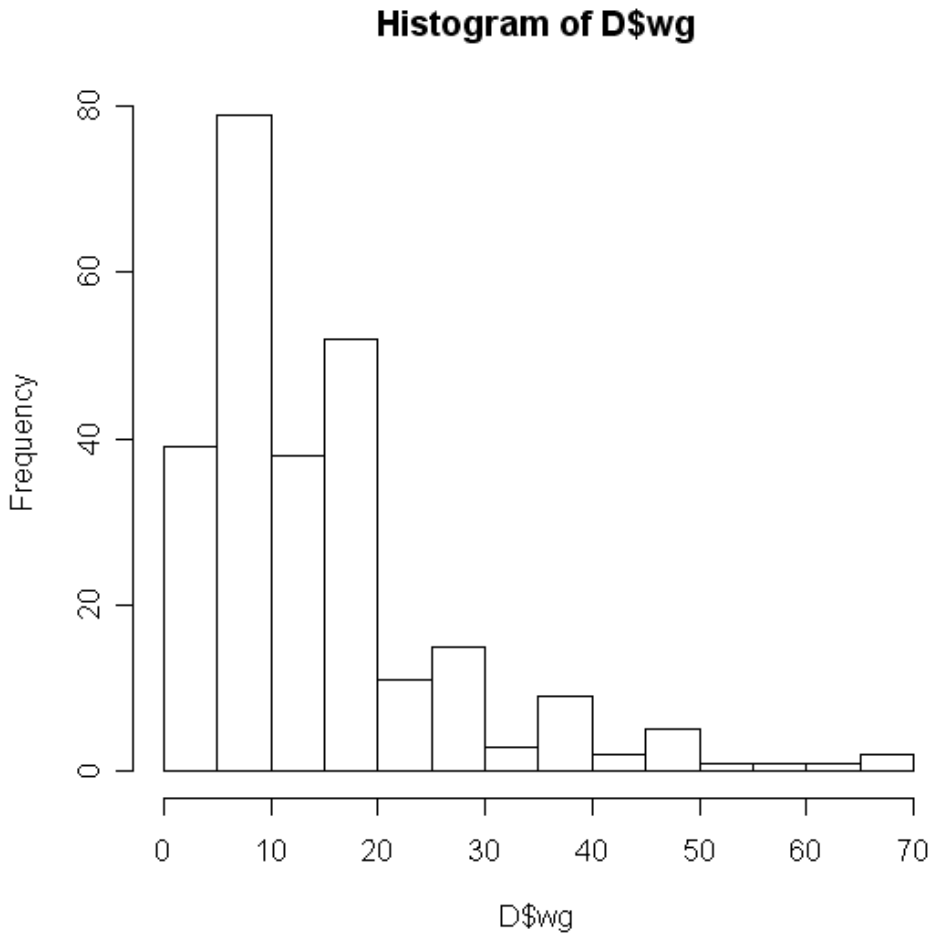
- Accessing columns.
- D has our data in it…. But you can’t see it directly.
- To select a column use `D$column`.

WORKING WITH DATA.

- Subsetting data.
- Use a logical operator to do this.
  - ==, >, <, <=, >=, <> are all logical operators.
  - Note that the “equals” logical operator is two = signs.
- Example:
  - `D[D$Gender == “M”,]`
  - This will return the rows of D where Gender is “M”.
  - Remember R is case sensitive!
  - This code does nothing to the original dataset.
  - `D.M <- D[D$Gender == “M”,]` gives a dataset with the appropriate rows.

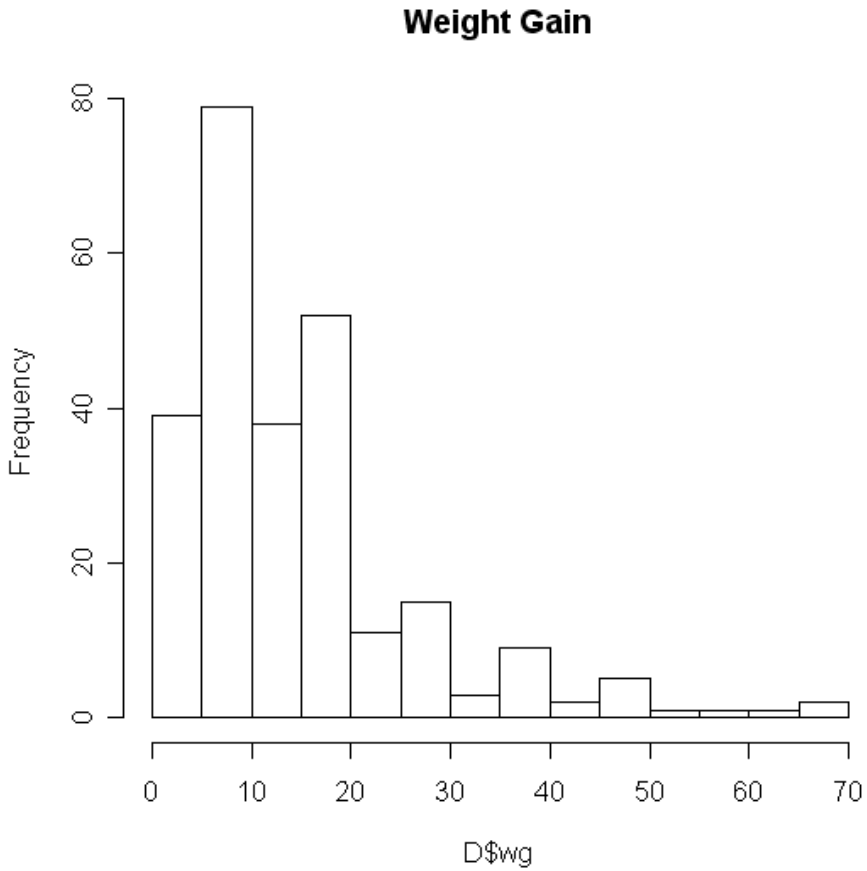
BASIC GRAPHICS

- Histogram
  - `hist(D$wg)`



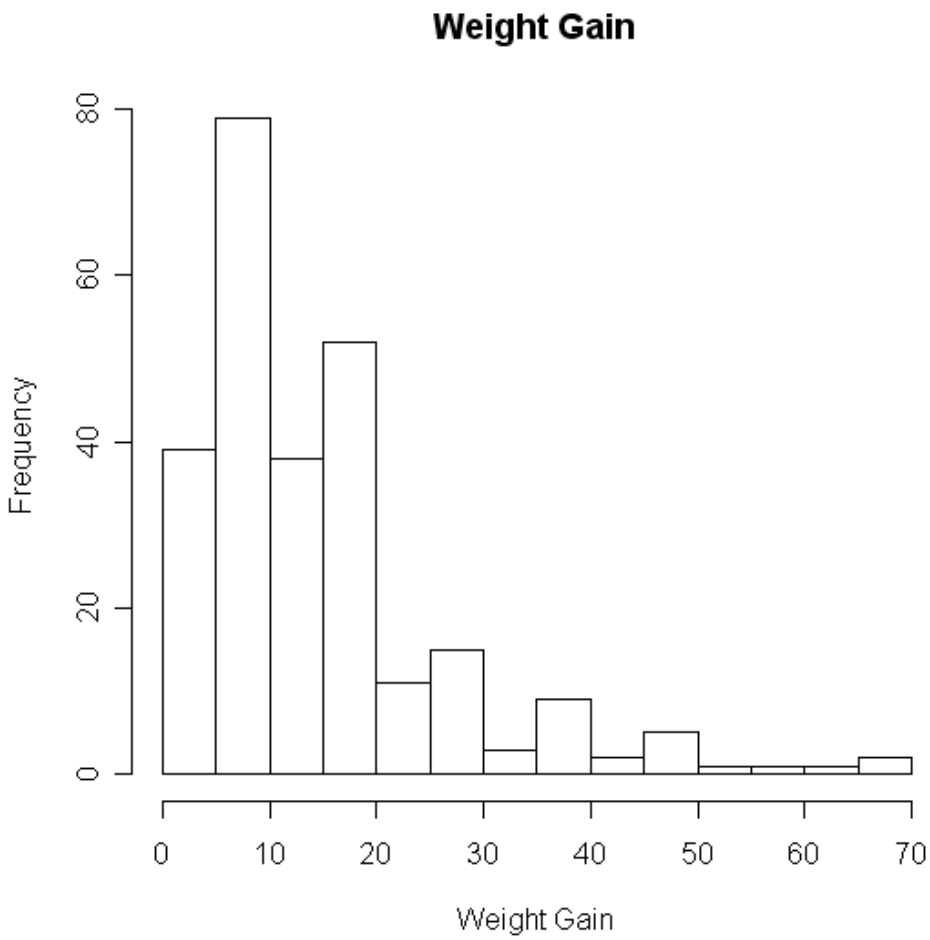
BASIC GRAPHICS

- Add a title…
  - The “main” statement will give the plot an overall heading.
  - `hist(D$wg , main='Weight Gain')`



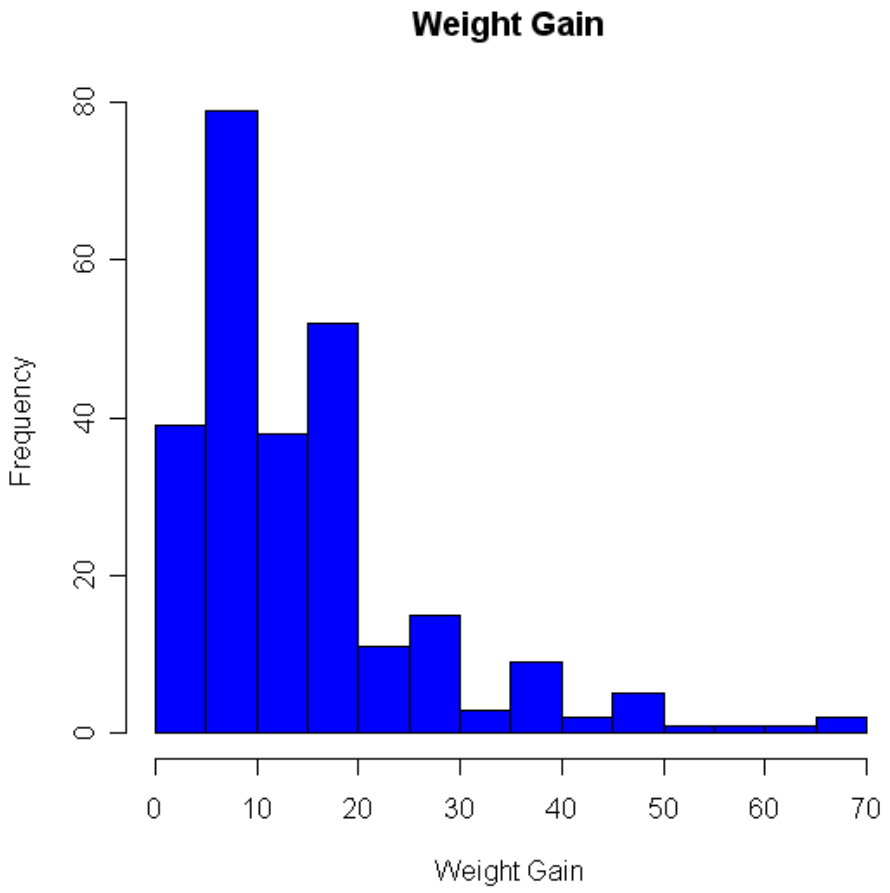
BASIC GRAPHICS

- Adding axis labels…
- Use “xlab” and “ylab” to label the X and Y axes, respectively.
- `hist(D$wg , main='Weight Gain',xlab='Weight Gain', ylab ='Frequency')`

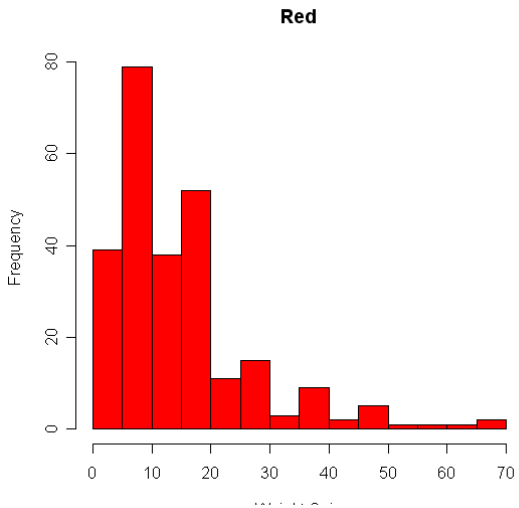


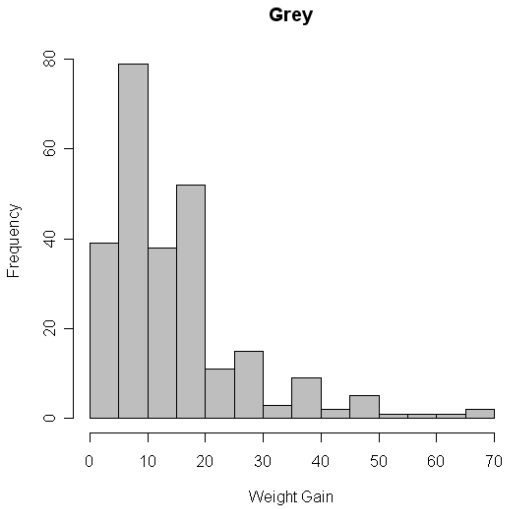
BASIC GRAPHICS

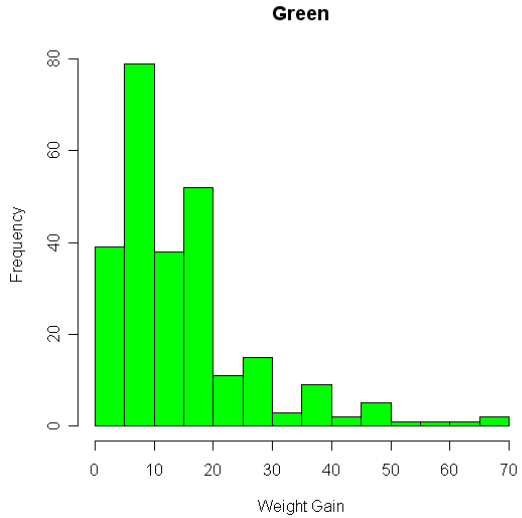
- Changing colors…
- Use the col statement.
  - ?colors will give you help on the colors.
  - Common colors may simply put in using the name.
  - `hist(D$wg, main="Weight Gain",xlab="Weight Gain", ylab ="Frequency", col="blue")`

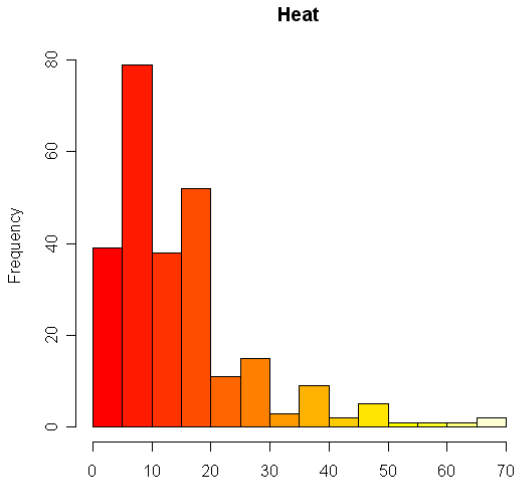


BASIC GRAPHICS - COLORS



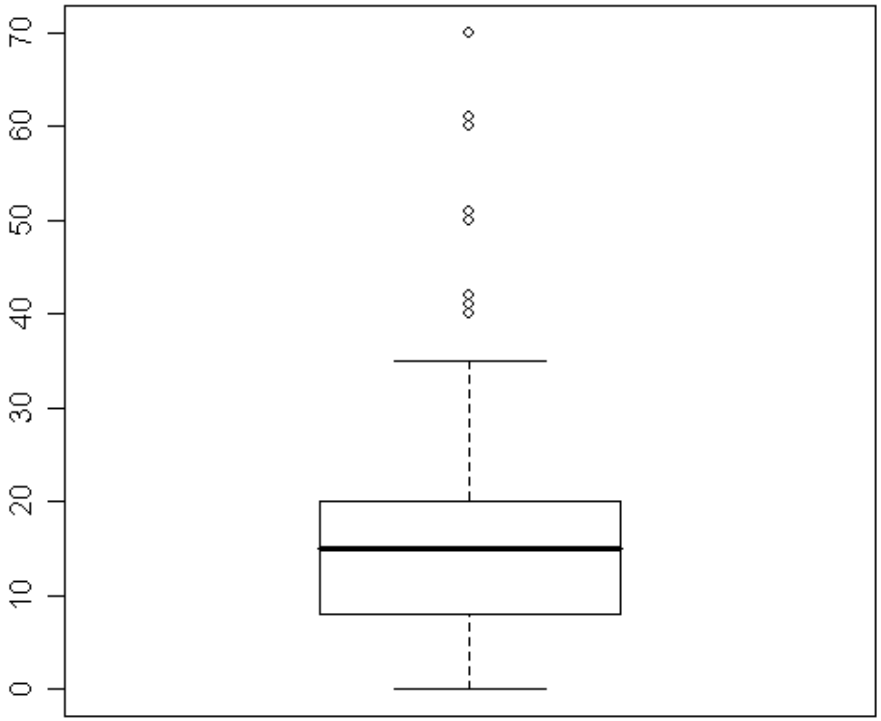






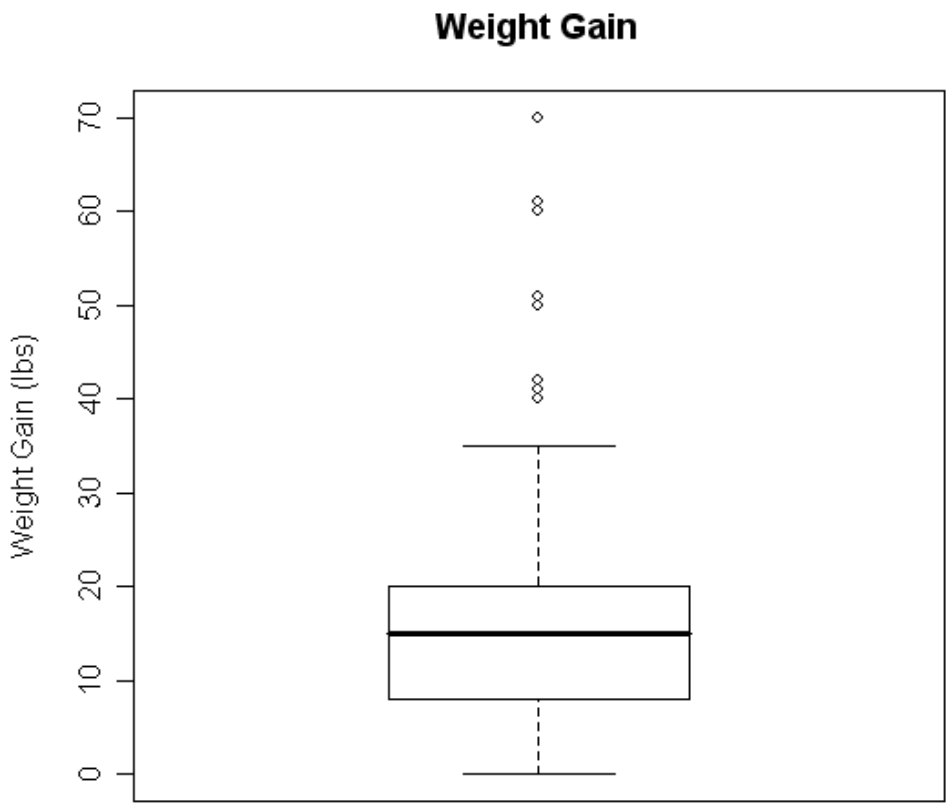
BASIC PLOTS

- Box Plots
- `boxplot(D$wg)`



BOXPLOTS

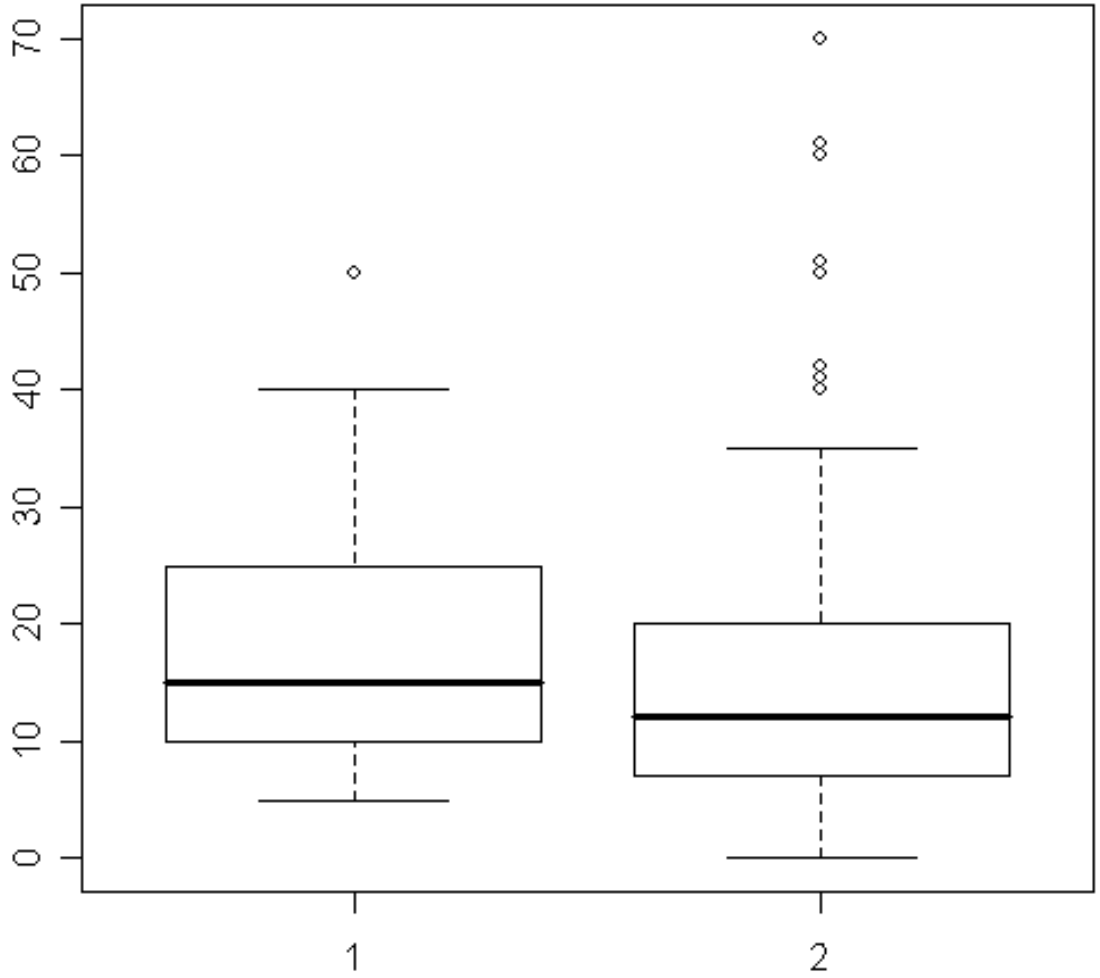
- Change it!
- `boxplot(D$wg,main='Weight Gain',ylab='Weight Gain (lbs)')`



BOX-PLOTS - GROUPINGS

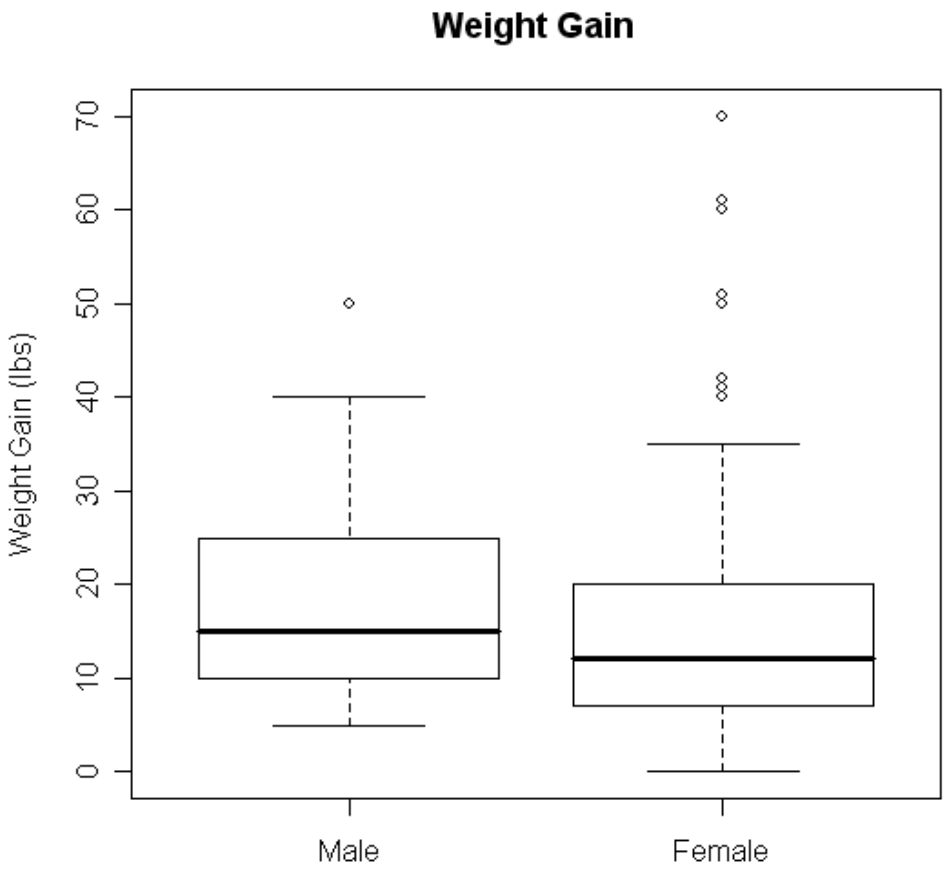
- What if we want several box plots side by side to be able to compare them.
- First Subset the Data into separate variables.
  - `wg.m <- D[D$Gender=="M",]`
  - `wg.f <- D[D$Gender=="F",]`
- Then Create the box plot.
  - `boxplot(wg.m$wg,wg.f$wg)`

BOXPLOTS - GROUPINGS



BOXPLOTS - GROUPINGS

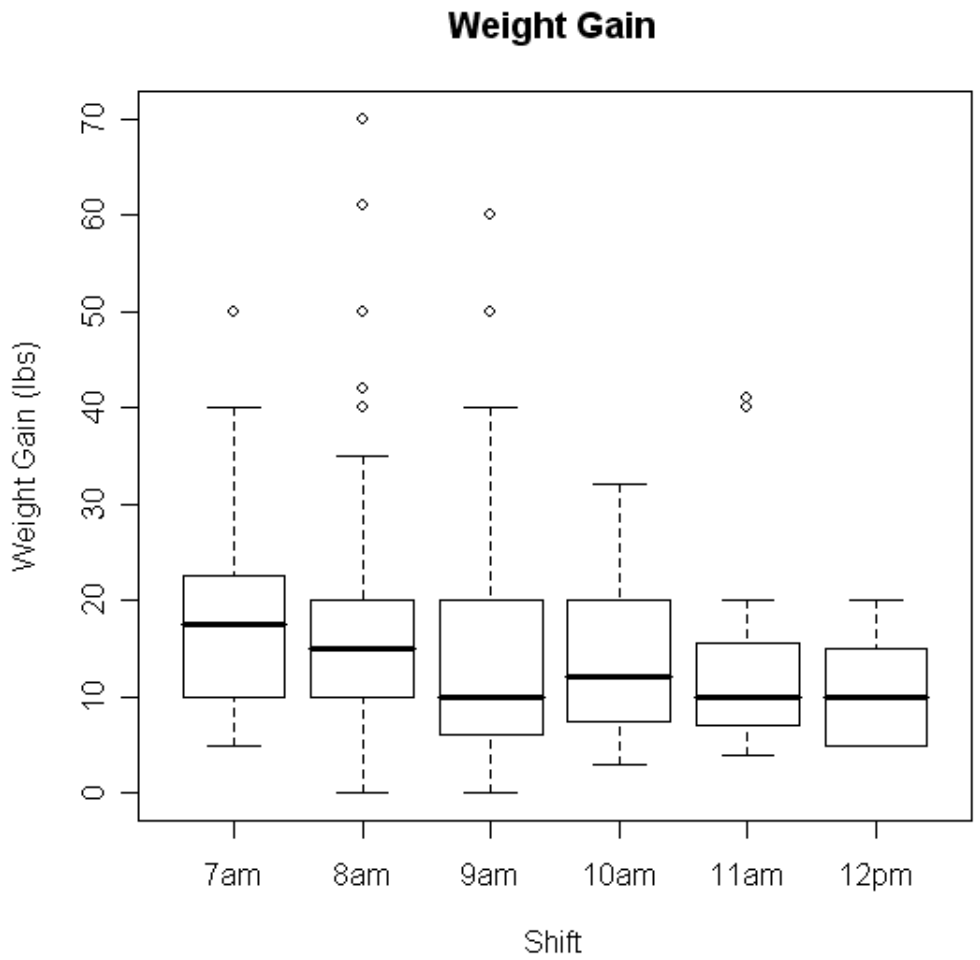
```
boxplot(wg.m$wg,
wg.f$wg, main='Weight
Gain (lbs)',
ylab='Weight Gain',
names =
c('Male','Female'))
```



BOXPLOT GROUPINGS

- Do it by shift
  - `wg.7a <- D[D$Shift=="7am",]`
  - `wg.8a <- D[D$Shift=="8am",]`
  - `wg.9a <- D[D$Shift=="9am",]`
  - `wg.10a <- D[D$Shift=="10am",]`
  - `wg.11a <- D[D$Shift=="11am",]`
  - `wg.12p <- D[D$Shift=="12pm",]`
  - `boxplot(wg.7a$wg, wg.8a$wg, wg.9a$wg, wg.10a$wg, wg.11a$wg, wg.12p$wg, main='Weight Gain', ylab='Weight Gain (lbs)', xlab='Shift', names = c('7am','8am','9am','10am','11am','12pm'))`

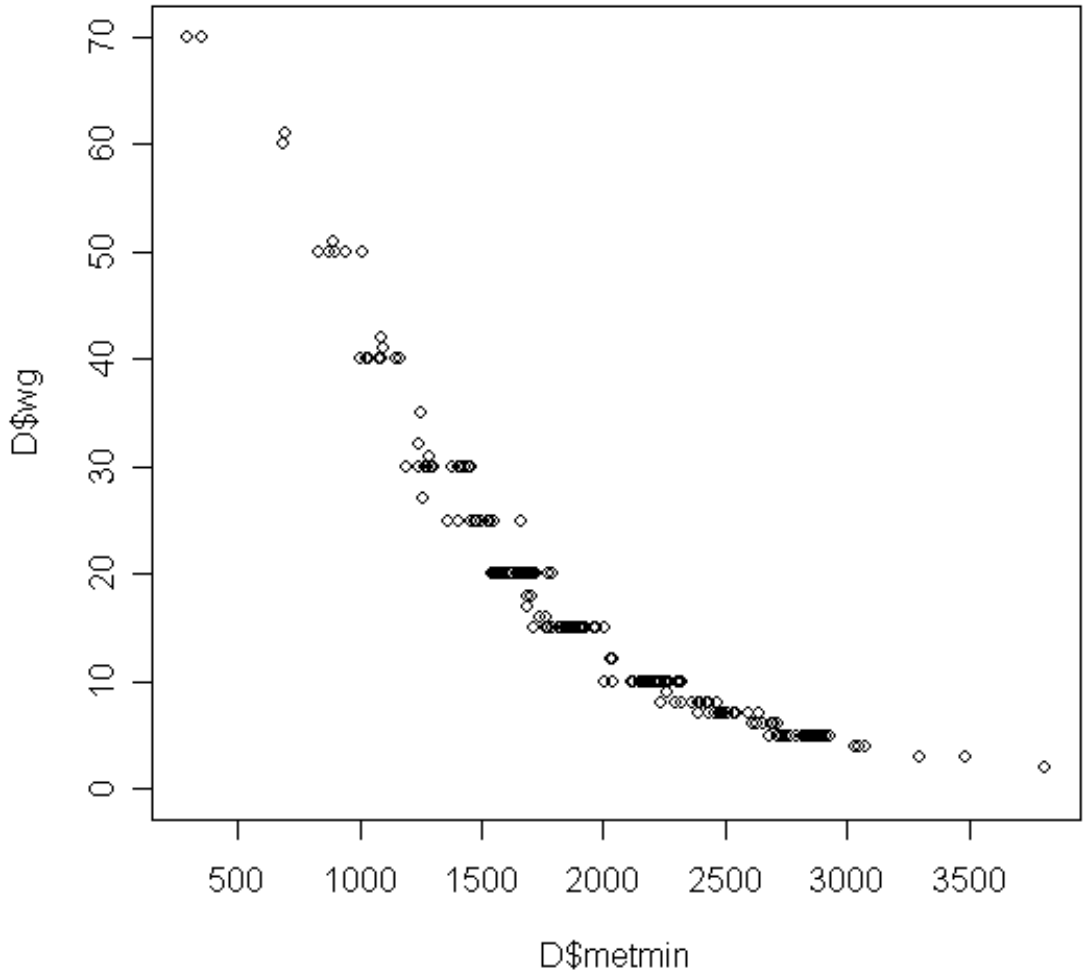
BOXPLOTS GROUPINGS



SCATTER PLOTS

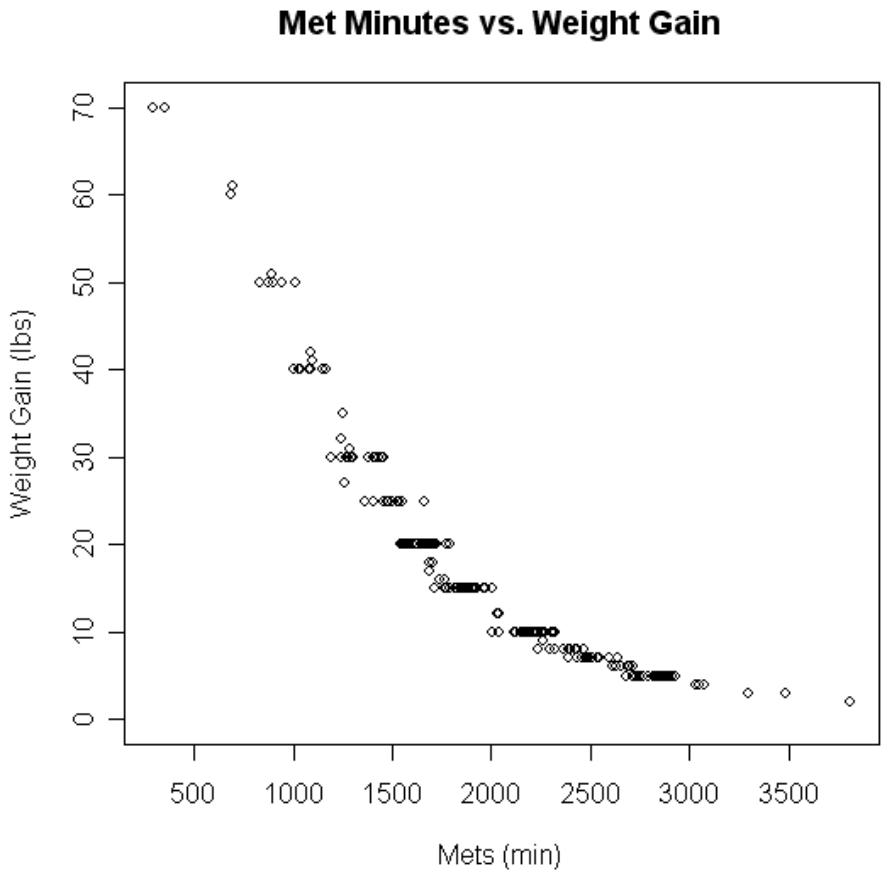
- Suppose we have two variables and we wish to see the relationship between them.
- A scatter plot works very well.
- R code:
  - `plot(x,y)`
- Example
  - `plot(D$metmin,D$wg)`

SCATTER PLOTS

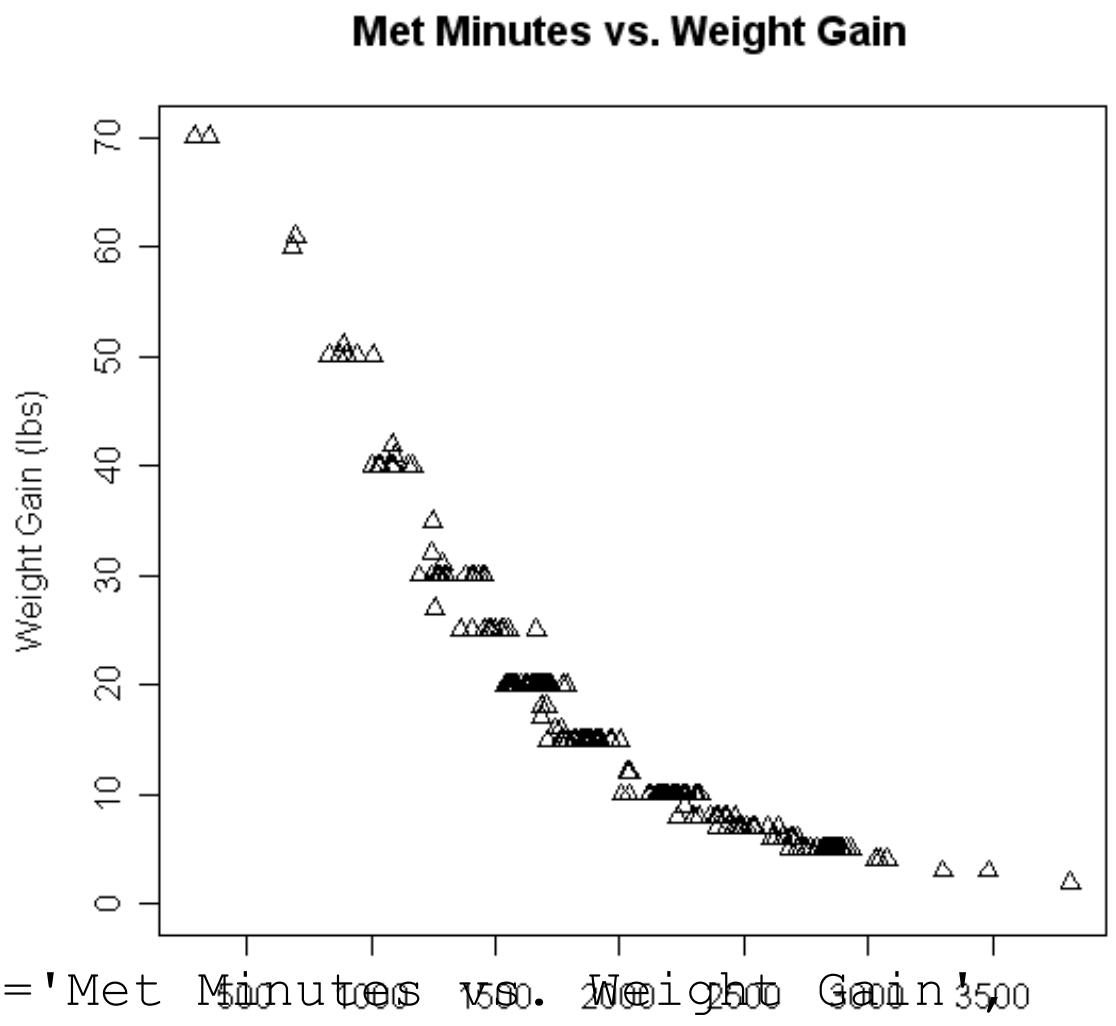


SCATTER PLOTS

```
plot(D$metmin,D$wg,main=
'Met Minutes vs. Weight
Gain',
xlab='Mets
(min)',ylab='Weight Gain
(lbs)')
```



SCATTER PLOTS

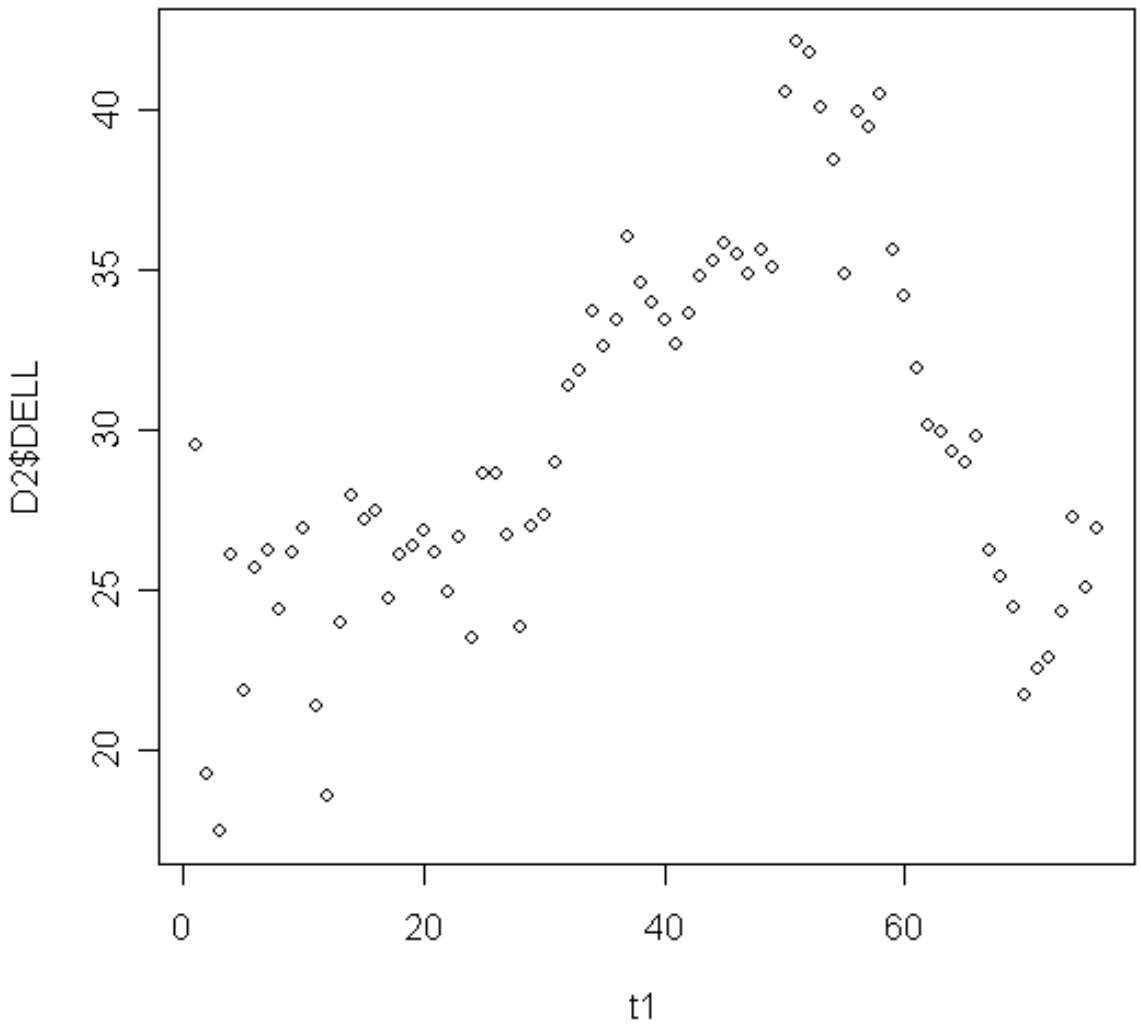



```
plot(D$metmin,D$wg,main='Met Minutes vs. Weight Gain',
 xlab='Mets (min)',ylab='Weight Gain (lbs)',pch=2)
```

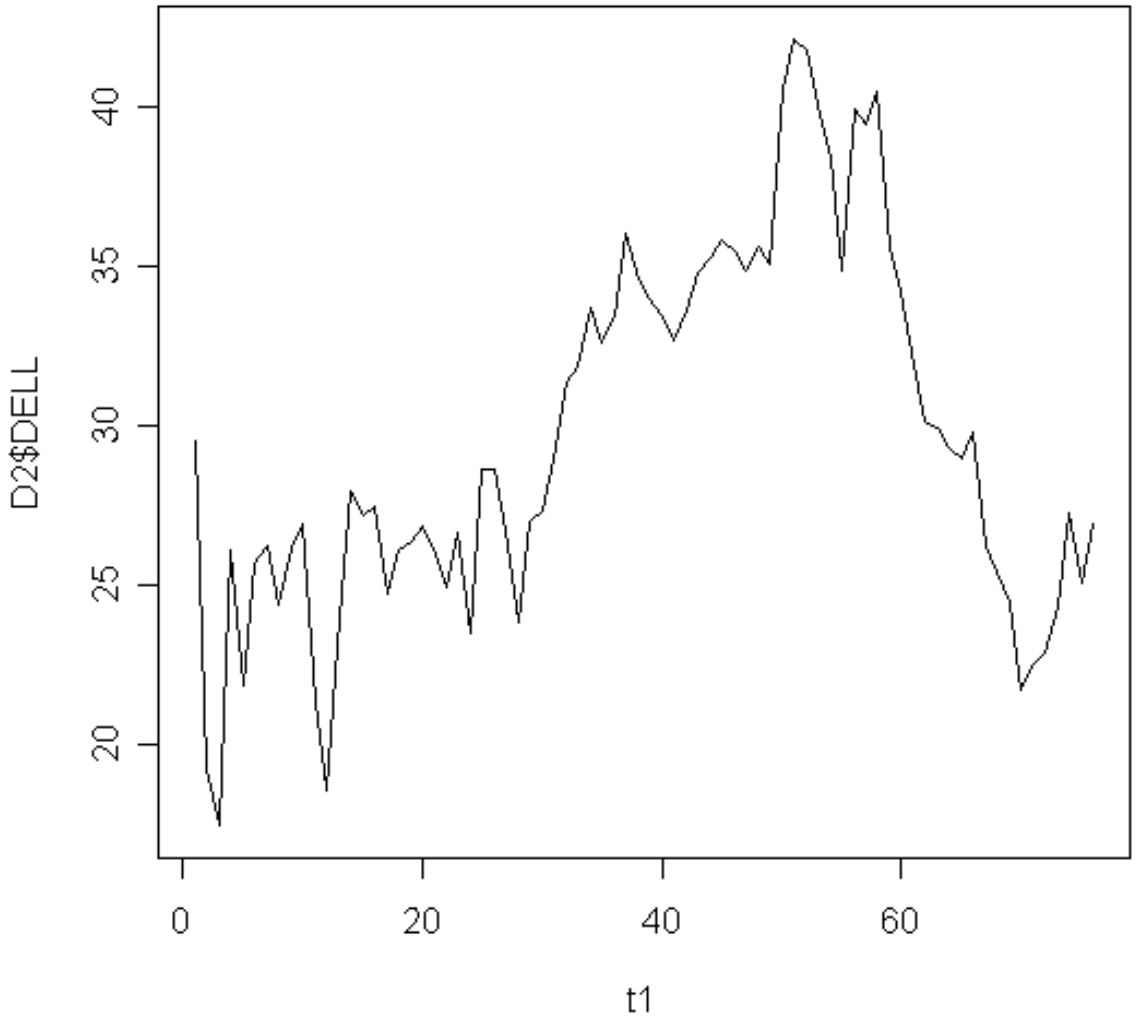
LINE PLOTS

- Often data comes through time.
- Consider Dell stock
  - `D2 <- read.csv("H:\\Dell.csv",header=TRUE)`
  - `t1 <- 1:nrow(D2)`
  - `plot(t1,D2$DELL)`

LINE PLOTS



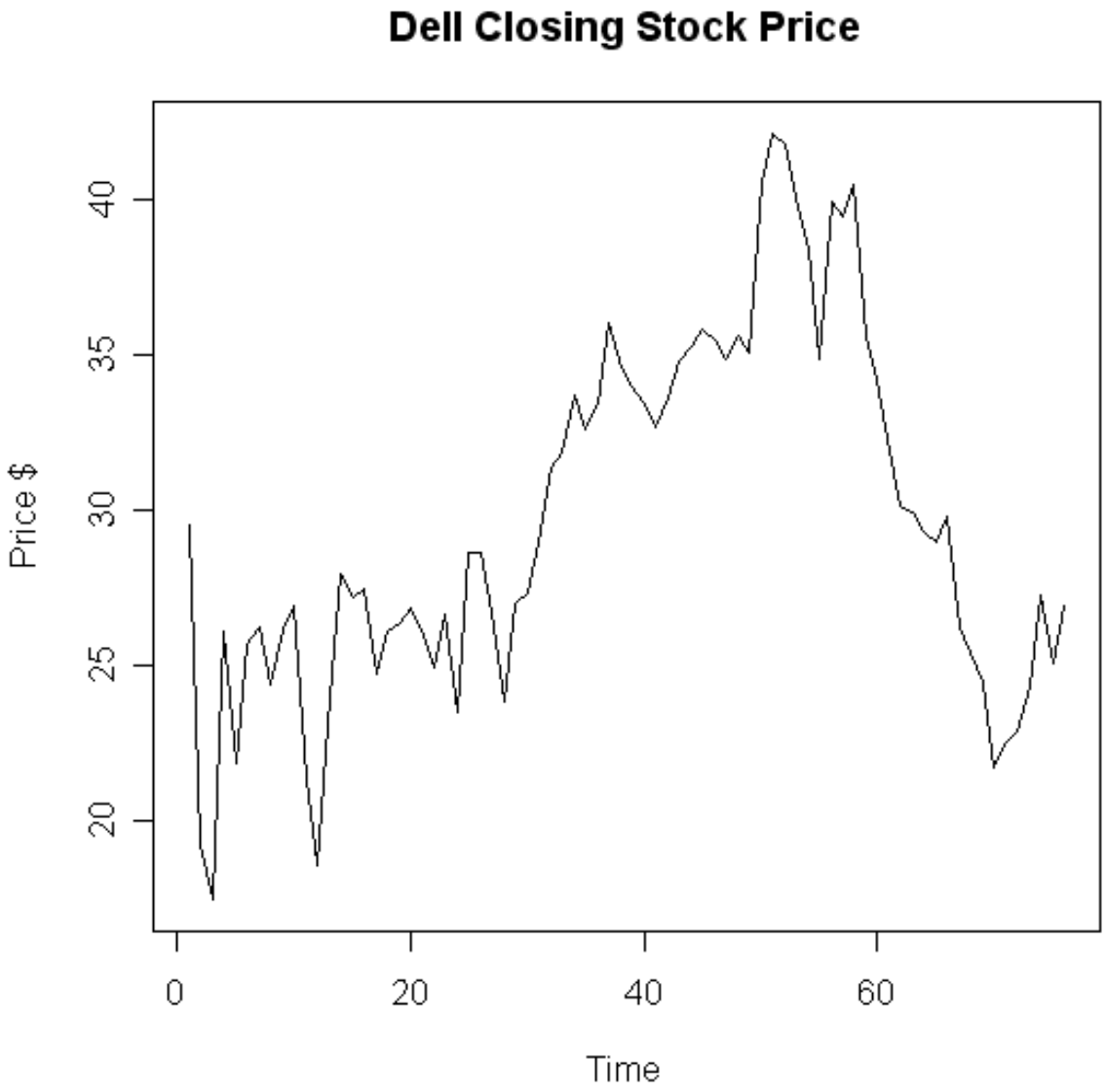
LINE PLOTS




```
plot(t1,D2$DELL,type="l")
```

LINE PLOTS

```
plot(t1,D2$DELL,type="l",main='Dell Closing Stock Price',
xlab='Time',ylab='Price $'))
```



OVERLAYING PLOTS

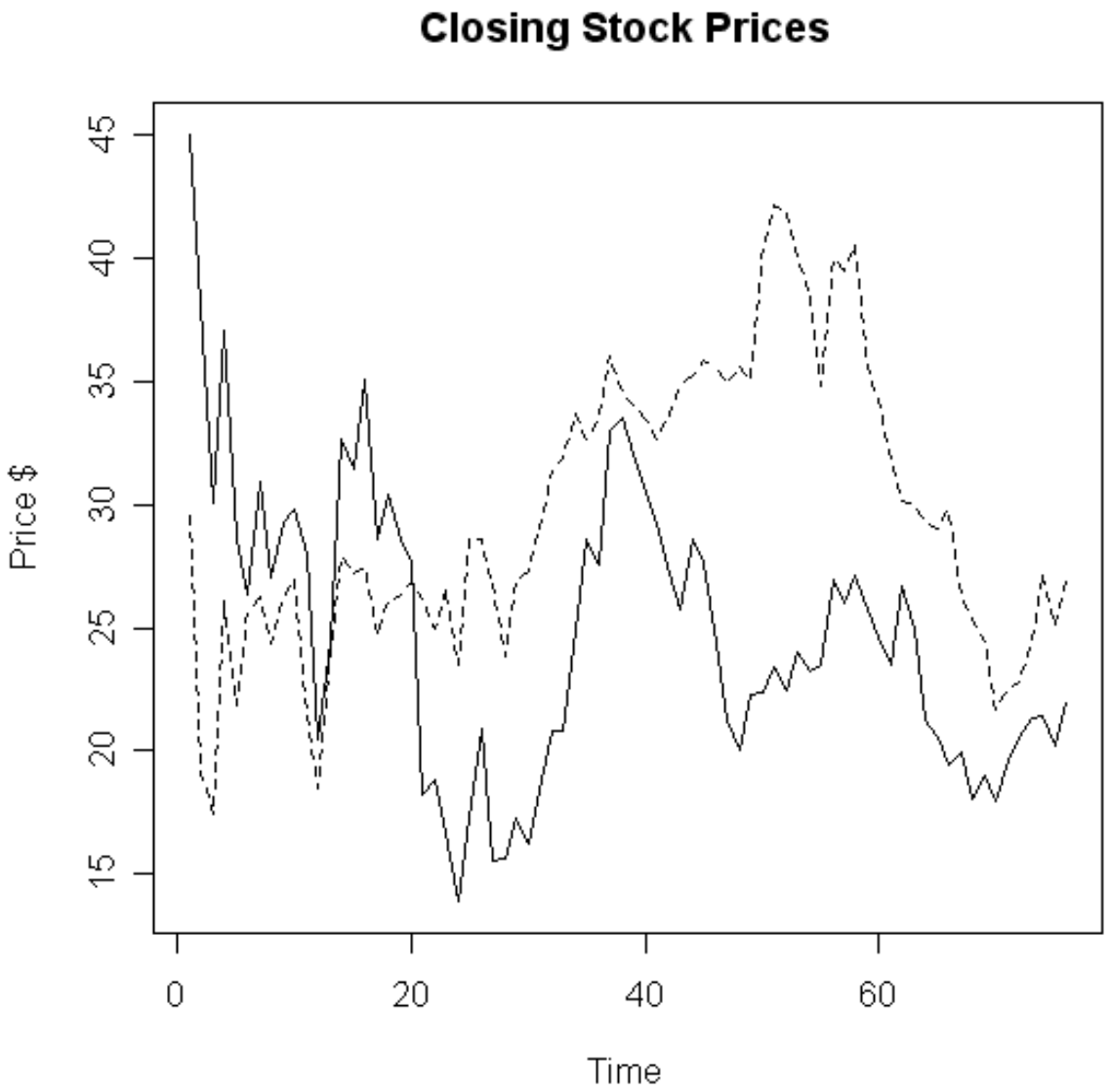
- Often we have more than one variable measured against the same predictor (X).
  - `plot(t1,D2$DELL,type="l",main='Dell Closing Stock Price',xlab='Time',ylab='Price $'))`
  - `lines(t1,D2$Intel)`

OVERLAYING GRAPHS

OVERLAYING GRAPHS

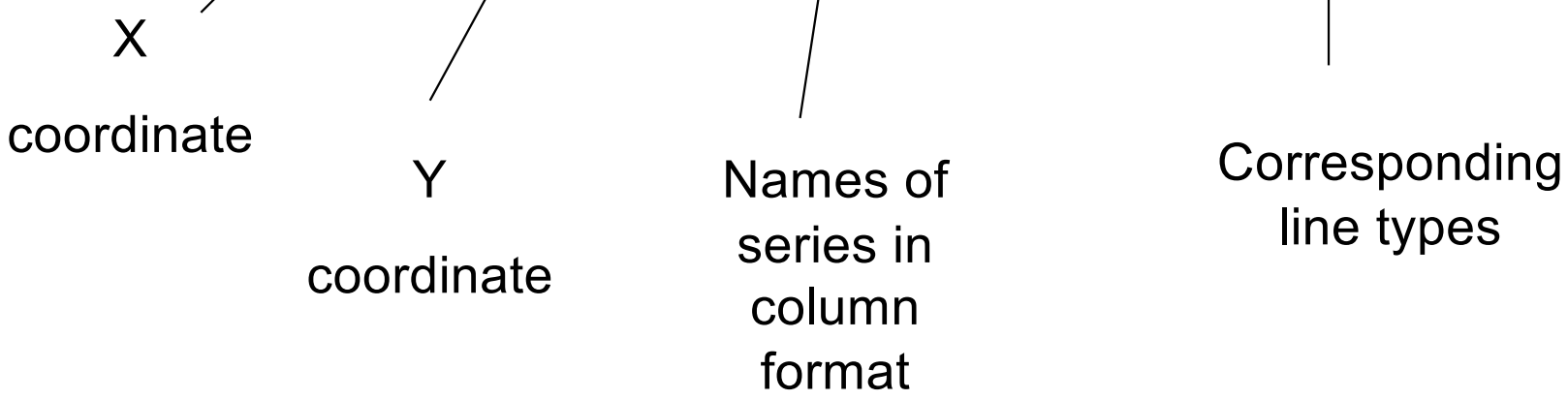
```
lines(t1,D2$Intel,lty=2)
```

OVERLAYING GRAPHS

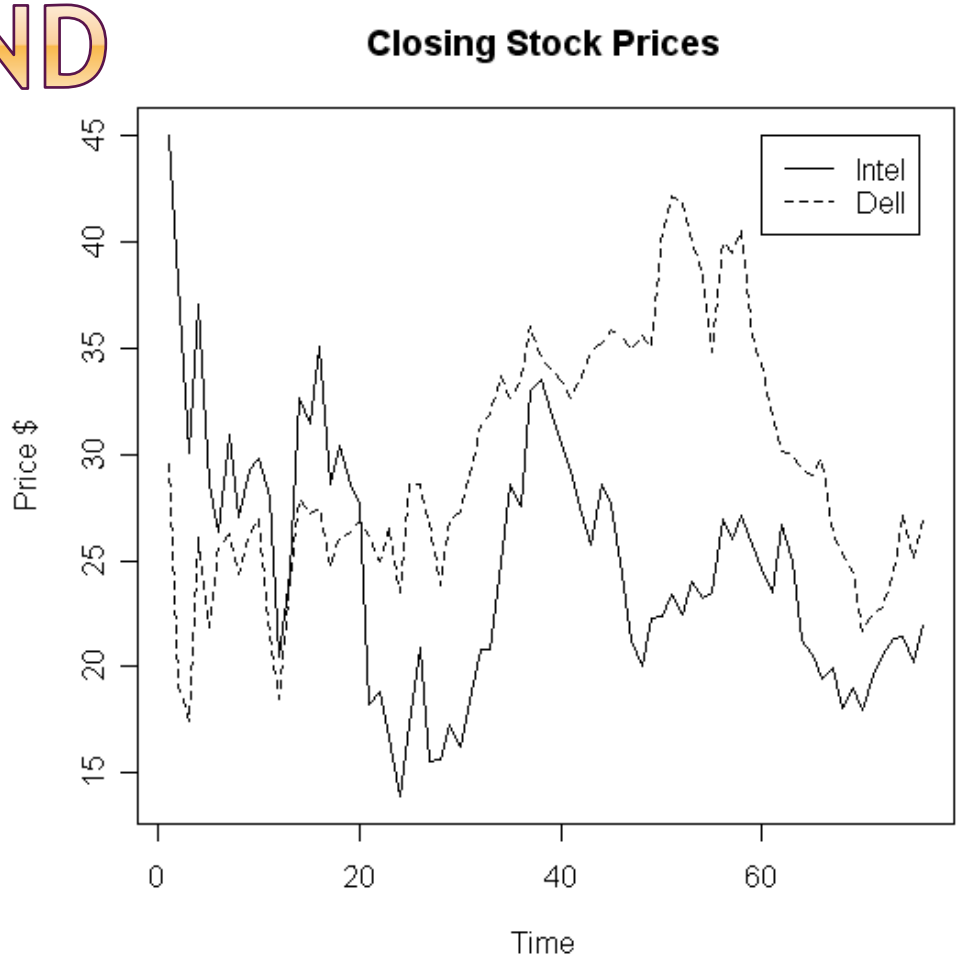


ADDING A LEGEND

- Adding a legend is a bit tricky in R.
- Syntax
- `legend( x, y, names, line types)`



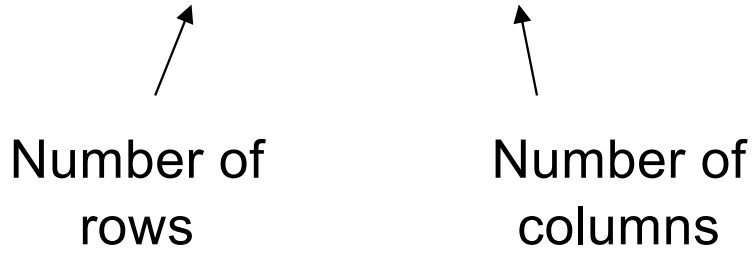
ADDING A LEGEND




```
legend(60,45,c('Intel','Dell'),lty=c(1,2))
```

PANELING GRAPHICS

- Suppose we want more than one graphic on a panel.
- We can partition the graphics panel to give us a framework in which to panel our plots.
- `par(mfrow = c( nrow, ncol))`

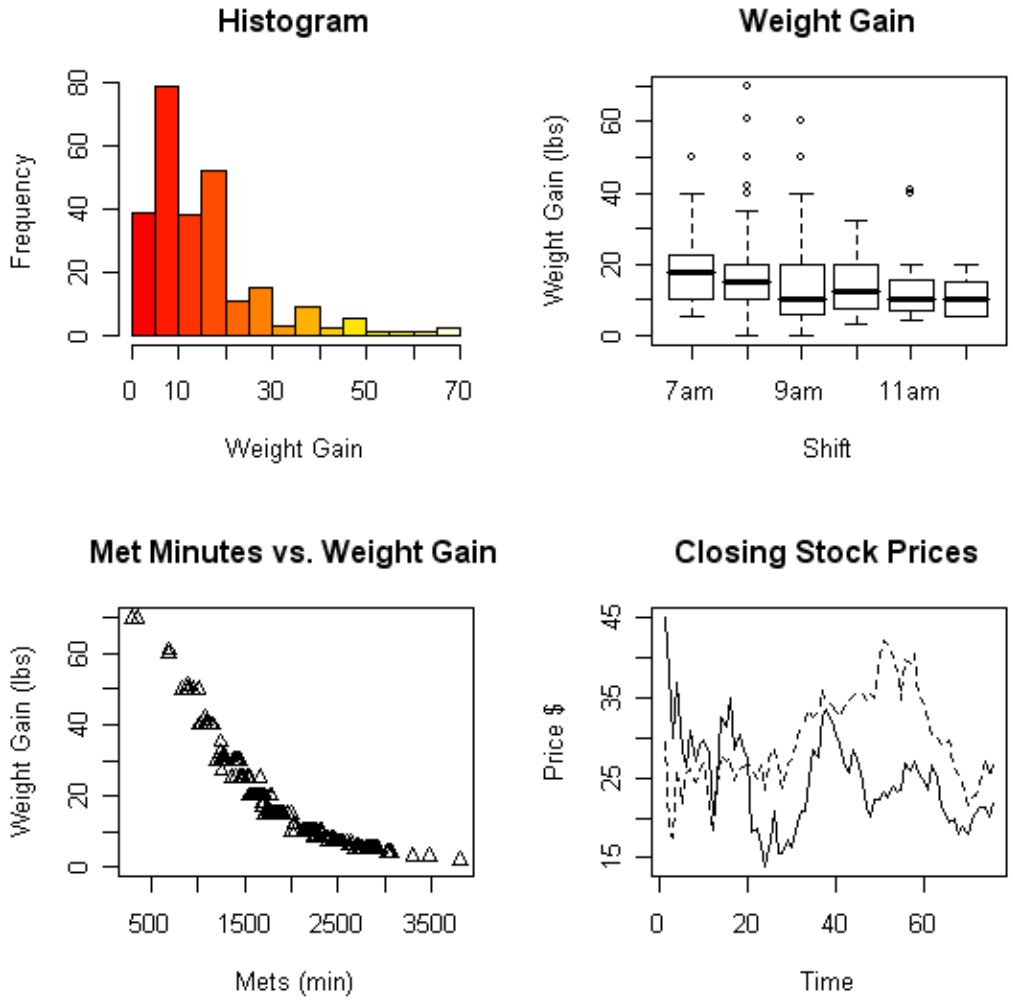


PANELING GRAPHICS

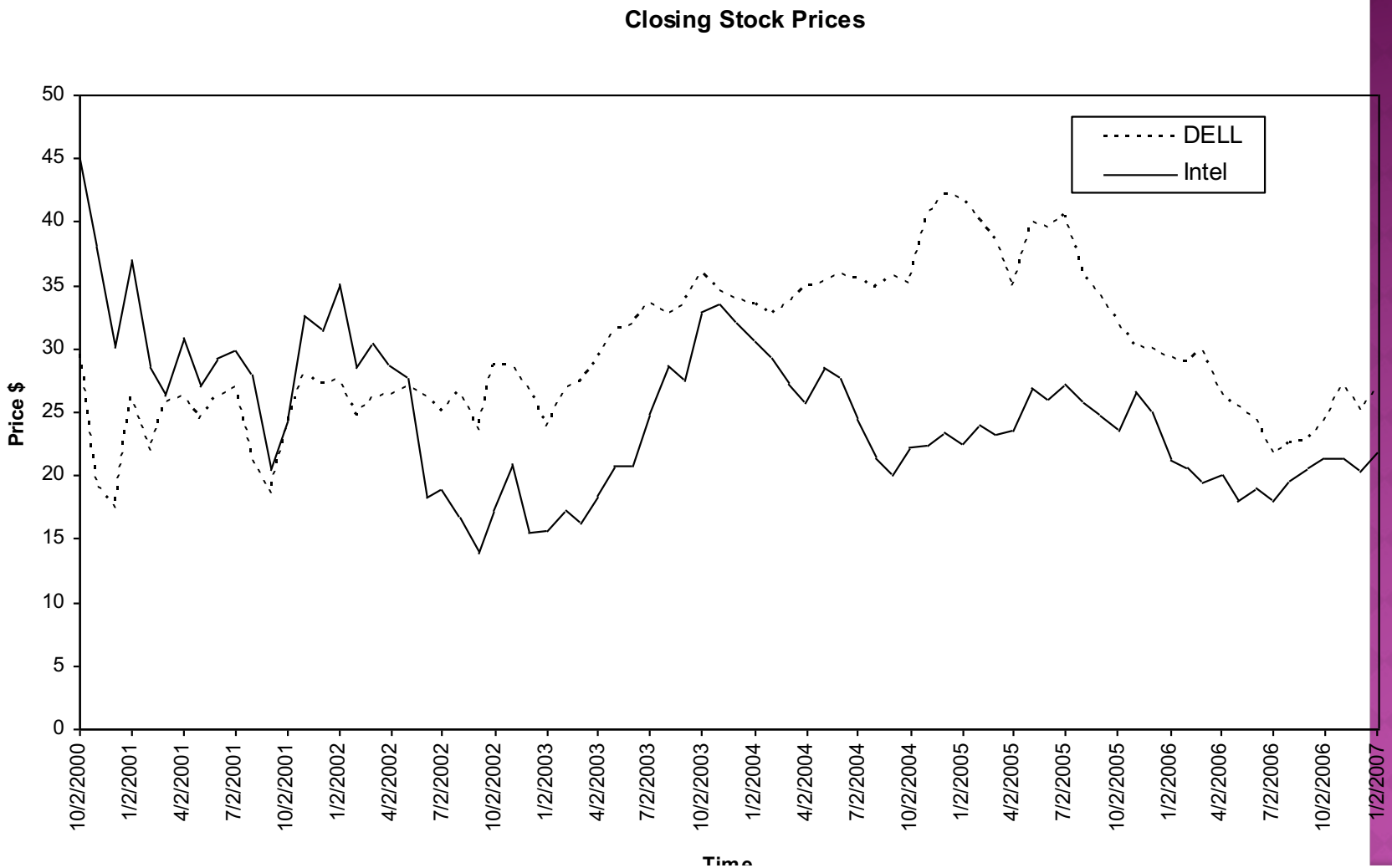
- Consider the following

```
par(mfrow=c(2,2))
hist(D$wg, main='Histogram',xlab='Weight Gain',
ylab ='Frequency', col=heat.colors(14))
boxplot(wg.7a$wg, wg.8a$wg, wg.9a$wg, wg.10a$wg,
wg.11a$wg, wg.12p$wg,  main='Weight Gain',
ylab='Weight Gain (lbs)',
xlab='Shift', names =
c('7am','8am','9am','10am','11am','12pm'))
plot(D$metmin,D$wg,main='Met Minutes vs. Weight
Gain', xlab='Mets (min)',ylab='Weight Gain
(lbs)',pch=2)
plot(t1,D2$Intel,type="l",main='Closing Stock
Prices',xlab='Time',ylab='Price $')
lines(t1,D2$DELL,lty=2)
```

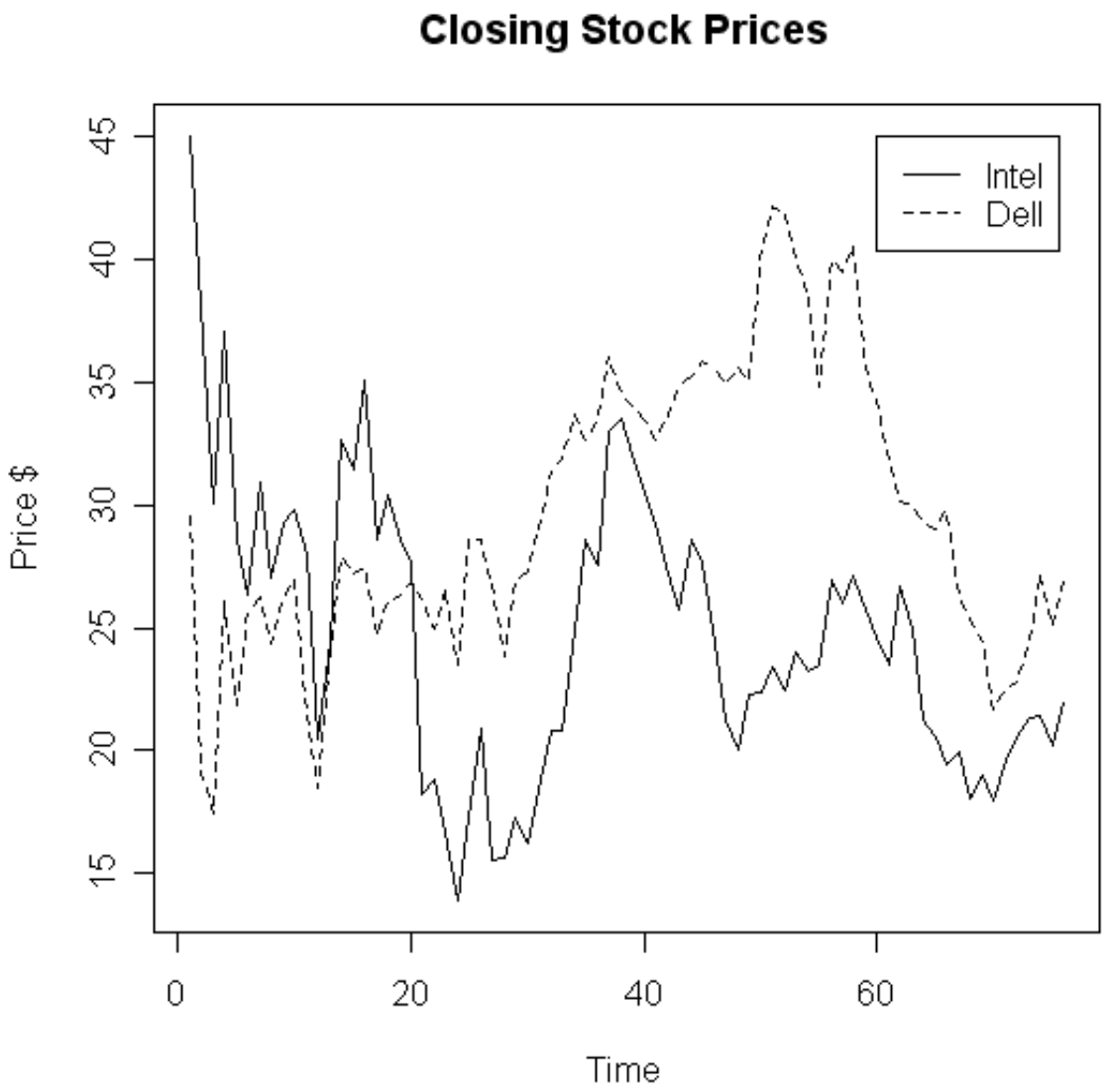
PANELING GRAPHICS



QUALITY - EXCEL



QUALITY - R


Closing Stock Prices
Intel
Dell
Price $
15
20
25
30
35
40
45
0
20
40
60
Time

SUMMARY

- R is an environment in which publication quality graphics can be generated.
- Similar statements control the graphics functions
  - Main
  - Xlab
  - Ylab