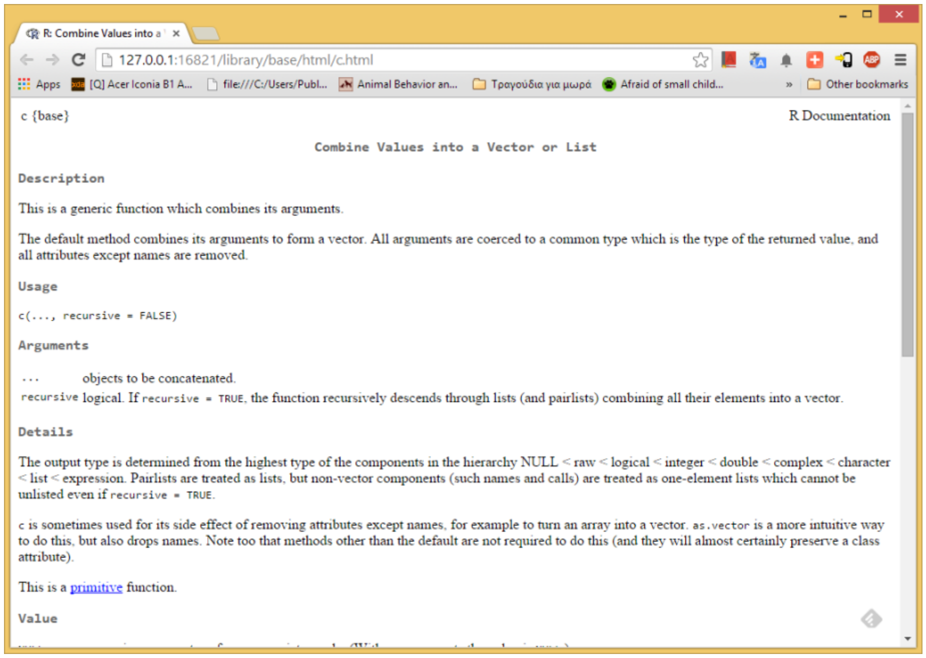
Περιεχόμενα

Γενικές εντολές

```
> 1+2*33-9^2/3
[1] 40
> sqrt(9)
[1] 3
> x<-1.57
> x
[1] 1.57
> x^2+15 -> y
> y
[1] 17.4649
> c(1,2,0.15,-sin(3))
```

```
[1]  1.00000  2.00000  0.15000 -0.14112
> help(c)
```



c {base} R Documentation

Combine Values into a Vector or List

Description

This is a generic function which combines its arguments.

The default method combines its arguments to form a vector. All arguments are coerced to a common type which is the type of the returned value, and all attributes except names are removed.

Usage

```
c(..., recursive = FALSE)
```

Arguments

`...` objects to be concatenated.

`recursive` logical. If `recursive = TRUE`, the function recursively descends through lists (and pairlists) combining all their elements into a vector.

Details

The output type is determined from the highest type of the components in the hierarchy NULL < raw < logical < integer < double < complex < character < list < expression. Pairlists are treated as lists, but non-vector components (such names and calls) are treated as one-element lists which cannot be unlisted even if `recursive = TRUE`.

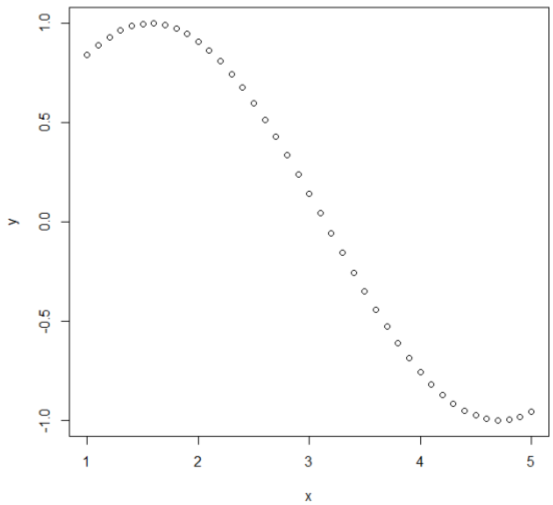
`c` is sometimes used for its side effect of removing attributes except names, for example to turn an array into a vector. `as.vector` is a more intuitive way to do this, but also drops names. Note too that methods other than the default are not required to do this (and they will almost certainly preserve a class attribute).

This is a primitive function.

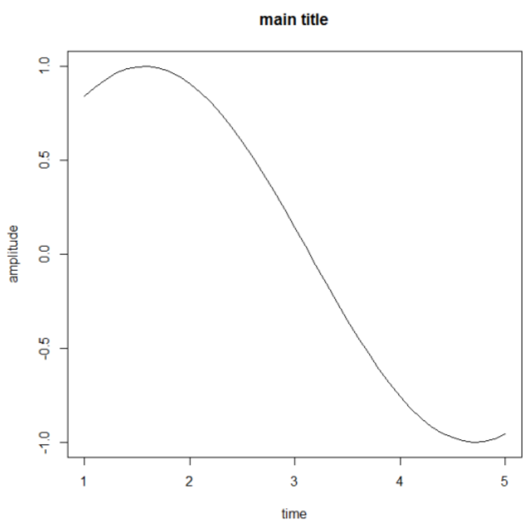
Value

```
> # auto einai sxolio (xrhsimo se R scripts)
>
> 1:5
[1] 1 2 3 4 5
> x <- seq(from=1, to=5, by=0.1)
> x
 [1] 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8
[20] 2.9 3.0 3.1 3.2 3.3 3.4 3.5 3.6 3.7 3.8 3.9 4.0 4.1 4.2 4.3 4.4 4.5 4.6 4.7
[39] 4.8 4.9 5.0
> y=sin(x)
> y
 [1]  0.84147098  0.89120736  0.93203909  0.96355819  0.98544973  0.99749499
 [7]  0.99957360  0.99166481  0.97384763  0.94630009  0.90929743  0.86320937
[13]  0.80849640  0.74570521  0.67546318  0.59847214  0.51550137  0.42737988
[19]  0.33498815  0.23924933  0.14112001  0.04158066 -0.05837414 -0.15774569
[25] -0.25554110 -0.35078323 -0.44252044 -0.52983614 -0.61185789 -0.68776616
[31] -0.75680250 -0.81827711 -0.87157577 -0.91616594 -0.95160207 -0.97753012
[37] -0.99369100 -0.99992326 -0.99616461 -0.98245261 -0.95892427
```

```
> plot(x,y)
```




```
> plot(x,y,type='l',xlab="time", ylab="amplitude", main="main title")
```



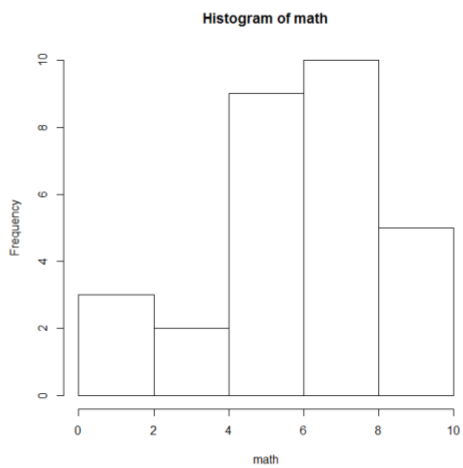

```
> country <- c("gr","gr","us","bg")
> am <- c(12, 58, 32, -17)
> d <- data.frame(country, am)
> d
  country  am
1      gr  12
2      gr  58
3      us  32
4      bg -17
> names(d)
[1] "country" "am"
> names(d)[2] = "amount"
> d
  country amount
1      gr     12
```

```
2      gr     58
3      us     32
4      bg    -17
> data <- read.table("survey_data.txt", header=TRUE, sep='\t')
> data
   prob math sex semester height weight birthmonth        color number
1   8.0  8.0   M        5   1.82     87        nov         blue   7777
2   5.5  4.0   M        5   1.73     63        oct        green   3865
3   6.0  9.0   F        5   1.68     52        jul    turquoise   2112
4   6.5  7.5   M        7   1.88     98        may        black    256
5    NA   NA   M       11   1.90    100        may        black   1618
6   5.0  6.5   M        8   1.74     65        apr         blue   2154
7   0.0  0.0   M        9   1.72     67        dec         blue      2
…
> dim(data)
[1] 38  9
> data[1,1]
[1] 8
> data[2,3]
[1] M
Levels: F M
> data[1,"math"]
[1] 8
> data[1,]
  prob math sex semester height weight birthmonth color number
1    8    8   M        5   1.82     87        nov  blue   7777
> data[2,]
  prob math sex semester height weight birthmonth color number
2  5.5    4   M        5   1.73     63        oct green   3865
> data[1,c("prob","math")]
  prob math
1    8    8
> data[1:3,c("prob","math")]
  prob math
1  8.0    8
2  5.5    4
3  6.0    9
> data[,1:2]
   prob math
1   8.0  8.0
2   5.5  4.0
3   6.0  9.0
4   6.5  7.5
5    NA   NA
6   5.0  6.5
…
> data$prob
 [1]  8.0  5.5  6.0  6.5   NA  5.0  0.0  0.0  7.5  7.5 10.0  5.0 10.0   NA  8.0
[16]  6.0  8.0  8.5  7.0  9.0  6.5   NA  6.0  1.0   NA   NA  5.5  8.5  6.5  6.0
[31] 10.0  8.0  9.0  6.0  7.0  6.5  7.5  5.5
> attach(data)
> prob
 [1]  8.0  5.5  6.0  6.5   NA  5.0  0.0  0.0  7.5  7.5 10.0  5.0 10.0   NA  8.0
```

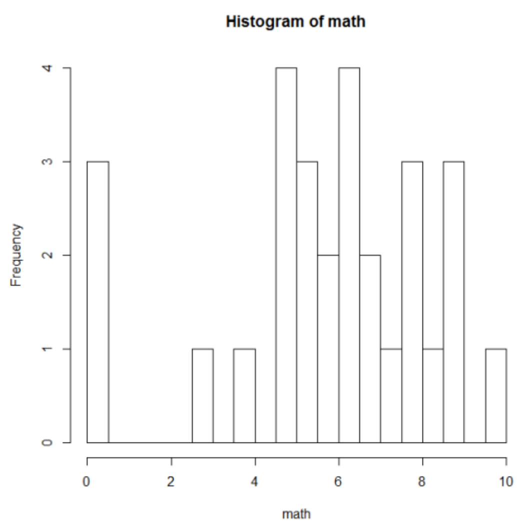
```
[16]  6.0  8.0  8.5  7.0  9.0  6.5   NA  6.0  1.0   NA   NA  5.5  8.5  6.5  6.0
[31] 10.0  8.0  9.0  6.0  7.0  6.5  7.5  5.5
> sex
 [1] M M F M M M M M F M F M M M M M M M F M M M M F M F F F M M F F M M M F M
Levels: F M
> math[sex=="F"]
 [1]  9.0  9.0 10.0   NA   NA  8.0  6.5  6.0  6.5  9.0  5.5
> math[sex=="F" & semester >= 7]
[1] 10.0   NA  8.0  6.5  6.0  6.5  9.0  5.5
> summary(data)
      prob             math         sex       semester          height     
 Min.   : 0.000   Min.   : 0.000   F:11   Min.   : 3.000   Min.   :1.590  
 1st Qu.: 6.000   1st Qu.: 5.000   M:27   1st Qu.: 5.000   1st Qu.:1.710  
 Median : 6.500   Median : 6.500          Median : 9.000   Median :1.780  
 Mean   : 6.576   Mean   : 5.948          Mean   : 8.605   Mean   :1.776  
 3rd Qu.: 8.000   3rd Qu.: 8.000          3rd Qu.:10.500   3rd Qu.:1.837  
 Max.   :10.000   Max.   :10.000          Max.   :18.000   Max.   :2.010  
 NA's   :5        NA's   :9                                                
     weight         birthmonth     color        number    
 Min.   : 45.00   nov    : 8   blue   :11   Min.   :   2  
 1st Qu.: 65.50   jul    : 7   black  : 6   1st Qu.: 269  
 Median : 75.50   apr    : 3   green  : 5   Median : 777  
 Mean   : 75.55   aug    : 3   red    : 4   Mean   :2342  
 3rd Qu.: 86.50   dec    : 3   orange : 3   3rd Qu.:3788  
 Max.   :100.00   jan    : 3   purple : 3   Max.   :9999  
                  (Other):11   (Other): 6   NA's   :1     
> fdata <- data[sex=="F",]
> summary(fdata)
      prob             math         sex       semester          height     
 Min.   : 5.500   Min.   : 5.500   F:11   Min.   : 3.000   Min.   :1.590  
 1st Qu.: 6.750   1st Qu.: 6.500   M: 0   1st Qu.: 6.000   1st Qu.:1.645  
 Median : 7.750   Median : 8.000          Median : 9.000   Median :1.670  
 Mean   : 7.750   Mean   : 7.722          Mean   : 8.545   Mean   :1.696  
 3rd Qu.: 8.875   3rd Qu.: 9.000          3rd Qu.: 9.000   3rd Qu.:1.710  
 Max.   :10.000   Max.   :10.000          Max.   :18.000   Max.   :2.010  
 NA's   :1        NA's   :2                                                
     weight        birthmonth          color        number    
 Min.   :45.00   nov    :3    purple      :3   Min.   :   3  
 1st Qu.:51.00   feb    :2    cyan        :2   1st Qu.: 583  
 Median :58.00   jan    :2    black       :1   Median : 777  
 Mean   :60.73   jul    :2    blue        :1   Mean   :2486  
 3rd Qu.:64.50   aug    :1    green       :1   3rd Qu.:3974  
 Max.   :98.00   sep    :1    oilish green:1   Max.   :7788  
                 (Other):0    (Other)     :2                  
```

Διερευνητική ανάλυση μιας ποσοτικής μεταβλητής

```
> hist(math)
```

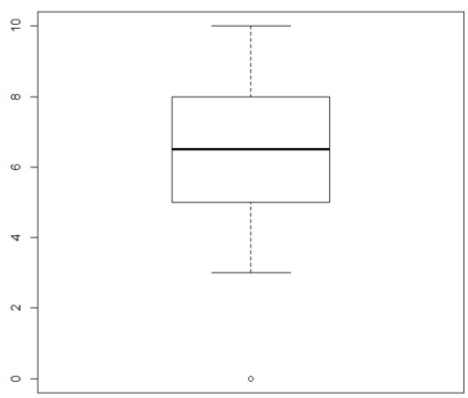



```
> hist(math, breaks=15)
```




```
> mean(math)
[1] NA
> math
 [1]  8.0  4.0  9.0  7.5   NA  6.5  0.0  0.0  9.0  5.0 10.0  5.5  7.0  6.5  5.5
[16]  7.0  8.0   NA  6.0   NA  0.0   NA   NA  3.0   NA   NA  8.0  6.5  6.0   NA
[31]  8.5  6.5  9.0  5.0  5.0   NA  5.5  5.0
> m <- math[!is.na(math)]
> m
 [1]  8.0  4.0  9.0  7.5  6.5  0.0  0.0  9.0  5.0 10.0  5.5  7.0  6.5  5.5  7.0
[16]  8.0  6.0  0.0  3.0  8.0  6.5  6.0  8.5  6.5  9.0  5.0  5.0  5.5  5.0
> mean(m)
[1] 5.948276
> sd(m)
[1] 2.61979
> var(m)
[1] 6.8633
> median(m)
```

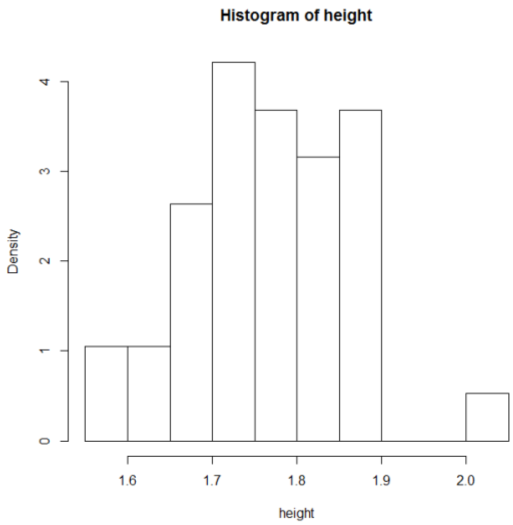
```
[1] 6.5
> range(m)
[1]  0 10
> quantile(m,0.77)
77%
  8
> fivenum(m)
[1]  0.0  5.0  6.5  8.0 10.0
> summary(m)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  0.000   5.000   6.500   5.948   8.000  10.000
> boxplot(m)
```



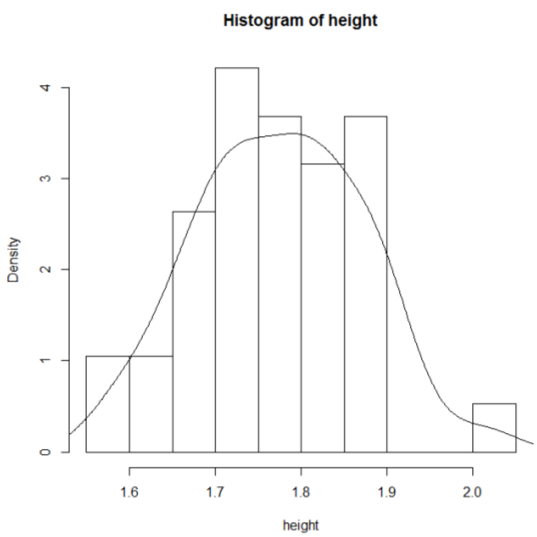

```
> F <- ecdf(height)
> F(1.70)
[1] 0.2368421
> F(1.70)-F(1.60)
[1] 0.1842105
> F(median(height))
[1] 0.5263158
```

Προσέγγιση με συναρτήσεις πυκνότητας

```
> hist(height, prob=TRUE)
```

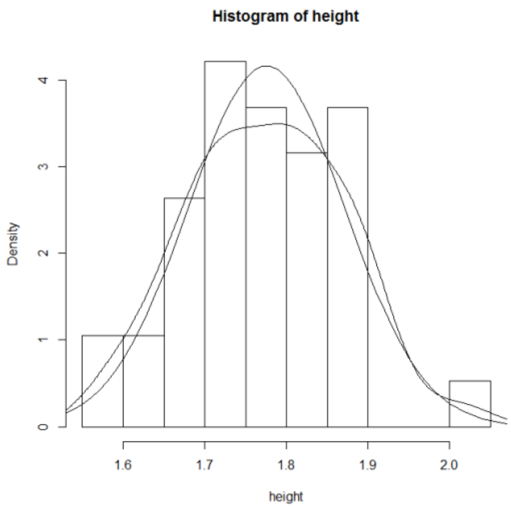



```
> pdf<- density(height)
> lines(pdf)
```

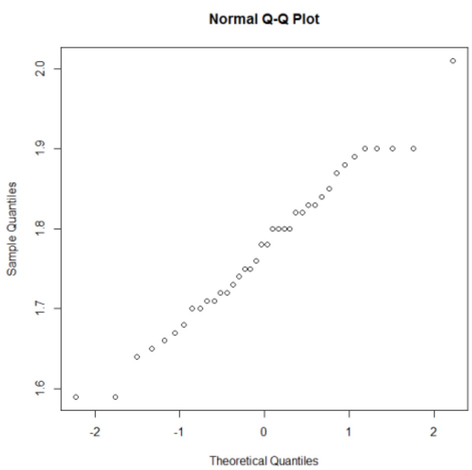


Προσέγγιση με Κανονική κατανομή

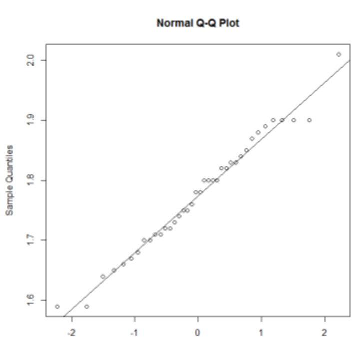
```
> m <- mean(height)
> s <- sd(height)
> x <- seq(from=1.5, to=2.1, by=0.01)
> y <- dnorm(x, mean=m, sd=s)
> lines(x,y)
```




```
> qqnorm(height)
```

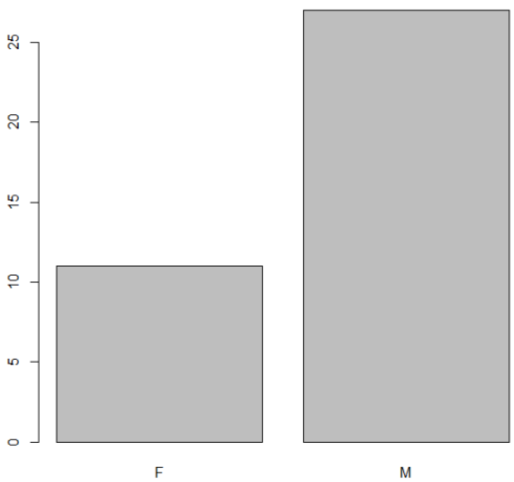



```
> qqline(height)
```

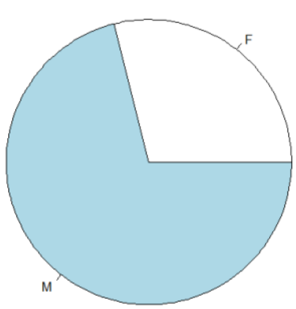


Διερευνητική ανάλυση μιας κατηγορικής μεταβλητής

```
> sex
 [1] M M F M M M M M F M F M M M M M M M M F M M M M F M F F F M M F F M M M F M
Levels: F M
> levels(sex)
[1] "F" "M"
> t <- table(sex)
> t
sex
 F  M
11 27
> prop.table(t)
sex
        F         M
0.2894737 0.7105263
> barplot(t)
```




```
> pie(t)
```

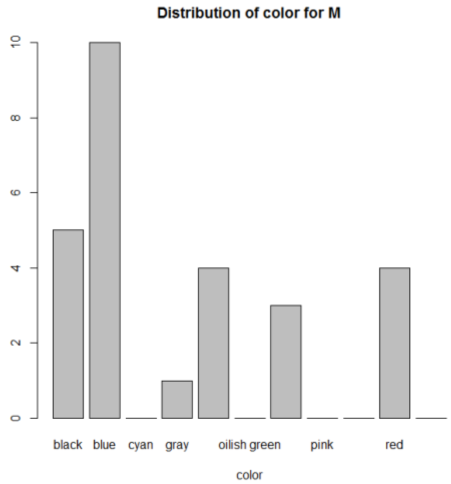


Διερευνητική ανάλυση σχέσης δύο κατηγορικών μεταβλητών

```
> t <- table(color, sex)
> t
              sex
color           F  M
  black         1  5
  blue          1 10
  cyan          2  0
  gray          0  1
  green         1  4
  oilish green  1  0
  orange        0  3
  pink          1  0
  purple        3  0
  red           0  4
  turquoise     1  0
> prop.table(t)
              sex
color                   F          M
  black        0.02631579 0.13157895
  blue         0.02631579 0.26315789
  cyan         0.05263158 0.00000000
  gray         0.00000000 0.02631579
  green        0.02631579 0.10526316
  oilish green 0.02631579 0.00000000
  orange       0.00000000 0.07894737
  pink         0.02631579 0.00000000
  purple       0.07894737 0.00000000
  red          0.00000000 0.10526316
  turquoise    0.02631579 0.00000000
> addmargins(t)
              sex
color           F  M Sum
  black         1  5   6
  blue          1 10  11
  cyan          2  0   2
  gray          0  1   1
  green         1  4   5
  oilish green  1  0   1
  orange        0  3   3
  pink          1  0   1
  purple        3  0   3
  red           0  4   4
  turquoise     1  0   1
  Sum          11 27  38
> addmargins(prop.table(t))
              sex
color                   F          M        Sum
  black        0.02631579 0.13157895 0.15789474
  blue         0.02631579 0.26315789 0.28947368
  cyan         0.05263158 0.00000000 0.05263158
  gray         0.00000000 0.02631579 0.02631579
```

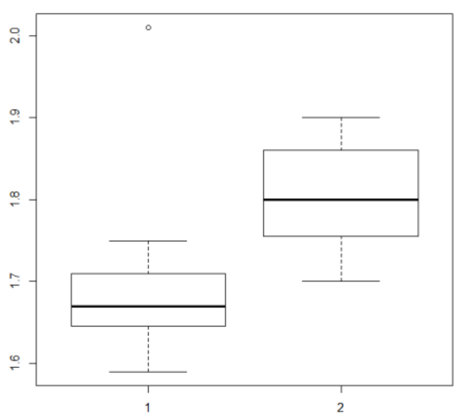
```
  green        0.02631579 0.10526316 0.13157895
  oilish green 0.02631579 0.00000000 0.02631579
  orange       0.00000000 0.07894737 0.07894737
  pink         0.02631579 0.00000000 0.02631579
  purple       0.07894737 0.00000000 0.07894737
  red          0.00000000 0.10526316 0.10526316
  turquoise    0.02631579 0.00000000 0.02631579
  Sum          0.28947368 0.71052632 1.00000000
> margin.table(t,1)
color
       black         blue         cyan         gray        green oilish green 
           6           11            2            1            5            1 
      orange         pink       purple          red    turquoise 
           3            1            3            4            1 
> margin.table(t,2)
sex
 F  M 
11 27 
> margin.table(prop.table(t),2)
sex
        F         M 
0.2894737 0.7105263 
> t[,"F"]
       black         blue         cyan         gray        green oilish green 
           1            1            2            0            1            1 
      orange         pink       purple          red    turquoise 
           0            1            3            0            1 
> prop.table(t[,"F"])
       black         blue         cyan         gray        green oilish green 
  0.09090909   0.09090909   0.18181818   0.00000000   0.09090909   0.09090909 
      orange         pink       purple          red    turquoise 
  0.00000000   0.09090909   0.27272727   0.00000000   0.09090909 
> prop.table(t[,"M"])
       black         blue         cyan         gray        green oilish green 
  0.18518519   0.37037037   0.00000000   0.03703704   0.14814815   0.00000000 
      orange         pink       purple          red    turquoise 
  0.11111111   0.00000000   0.00000000   0.14814815   0.00000000 
> 
```

```
barplot(t[,"M"], xlab="color", main="Distribution of color for M")
```

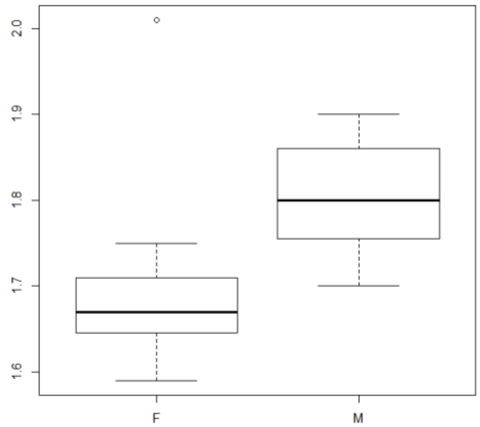


Διερευνητική ανάλυση σχέσης μεταξύ κατηγορικής και ποσοτικής μεταβλητής

```
> hm <- height[sex=="M"]
> hf <- height[sex=="F"]
> boxplot(hf, hm)
```

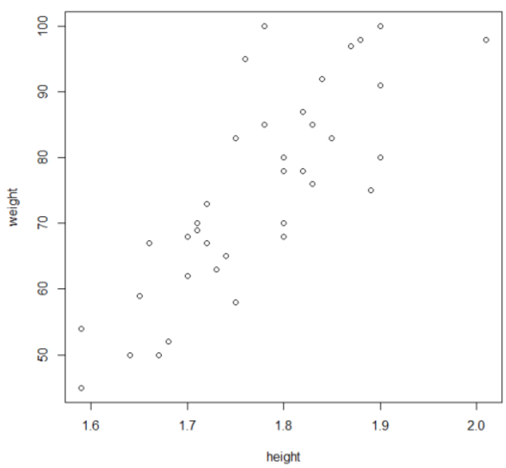



```
> boxplot(height~sex)
```

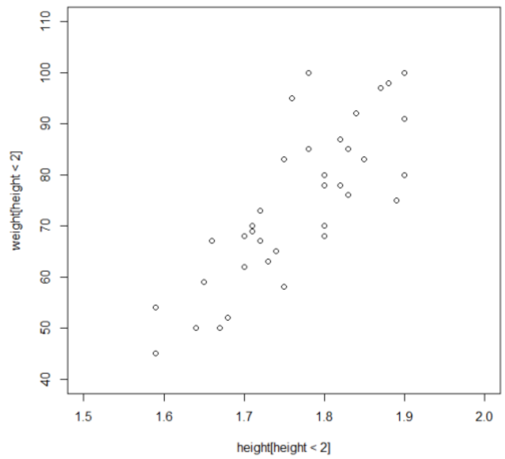


Διερευνητική ανάλυση σχέσης μεταξύ δύο ποσοτικών μεταβλητών

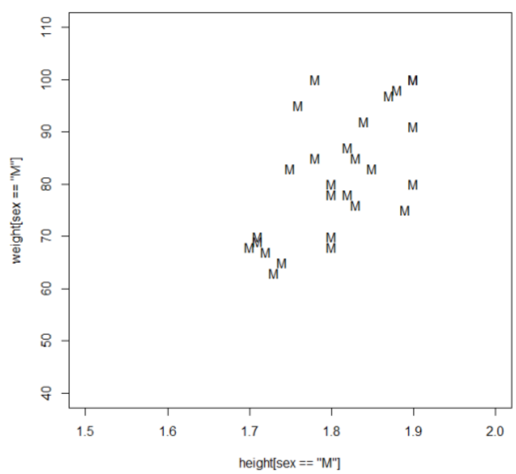
> plot(height, weight)   ή ισοδύναμα   plot(weight~height)



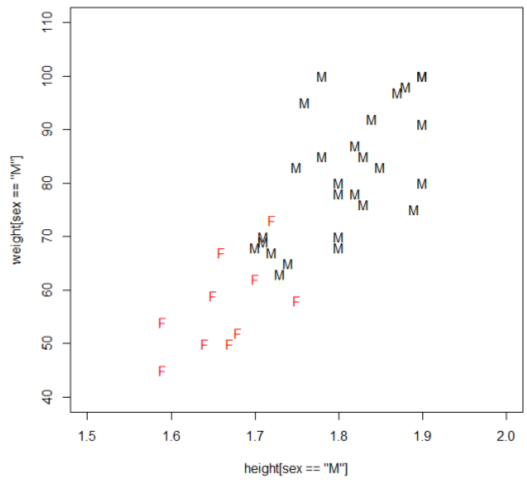
> plot(weight[height<2]~height[height<2], xlim=c(1.5,2), ylim=c(40,110))



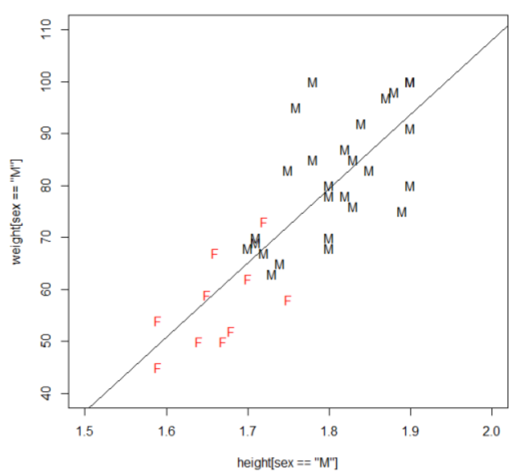

```
> plot(weight[sex=="M"]~height[sex=="M"], xlim=c(1.5,2), ylim=c(40,110),
pch='M')
```



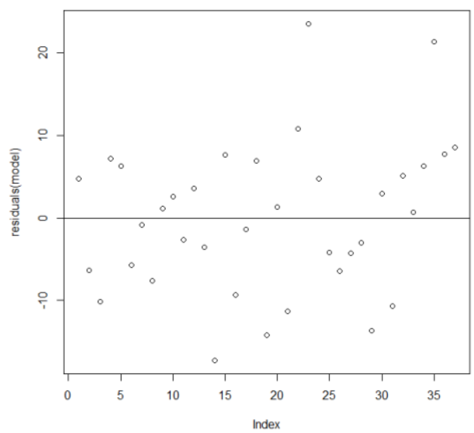

```
> points(weight[sex=="F" & height<2]~height[sex=="F" & height<2], pch='F',
col="red")
```




```
> cor(height[height<2], weight[height<2])
[1] 0.8139067
> model <- lm(weight[height<2]~height[height<2])
> model

Call:
lm(formula = weight[height < 2] ~ height[height < 2])

Coefficients:
       (Intercept)  height[height < 2]
            -178.5               143.2
> abline(model)
```

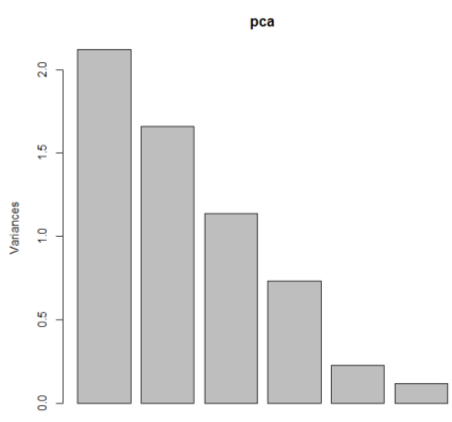



```
> plot(residuals(model))
> abline(0,0)
```

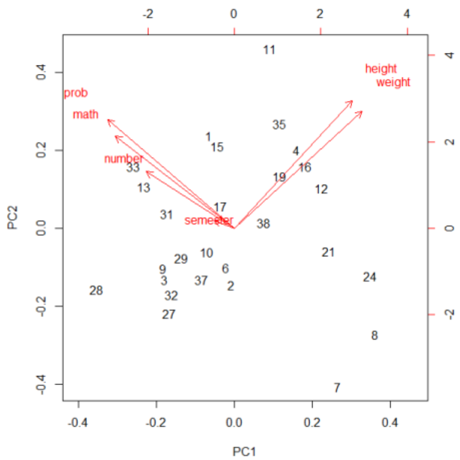


Principal Component Analysis

```
> d <- data[,c("prob","math","semester","height","weight","number")]
> dnona <- na.omit(d)
> pca <- prcomp(dnona, scale=TRUE)
> pca
Standard deviations:
[1] 1.4563600 1.2885937 1.0658495 0.8563442 0.4791612 0.3458118

Rotation:
                PC1        PC2         PC3         PC4          PC5         PC6
prob     -0.4843397 0.46922742 -0.10384904  0.06619455 -0.726108947  0.05329274
math     -0.4541983 0.39881885 -0.30504399  0.36820870  0.637106931 -0.01061848
semester -0.0761411 0.03363462  0.81823993  0.55130166 -0.004518525 -0.14000258
height    0.4494213 0.55185114 -0.03505507 -0.09604642  0.002087062 -0.69499836
weight    0.4873052 0.50464116  0.09480392  0.07418001  0.045708372  0.70092170
number   -0.3374547 0.24583630  0.46522072 -0.73578497  0.254453389  0.05596839
> plot(pca)
```

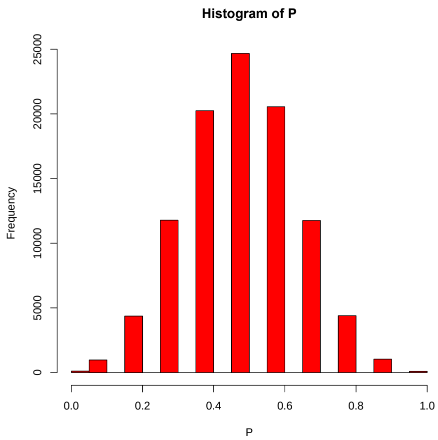



```
> biplot(pca)
```

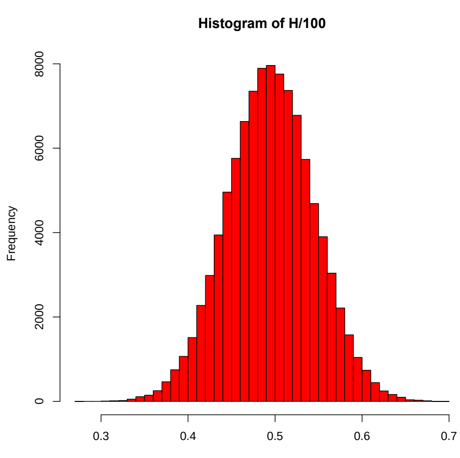


Δειγματοληψία

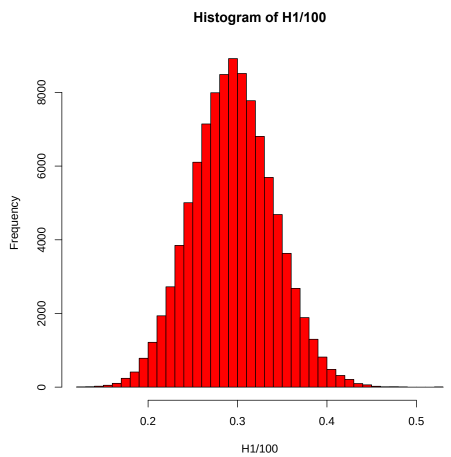
```
> rep(x=1,4)
[1] 1 1 1 1
> p <- 0.5 # proportion of 1's in a 0-1 population given below
> N <- 200000 # population size

> pop <- c(rep(x=1,p*N), rep(x=0,(1-p)*N))
> length(pop)
[1] 2000000
> mean(pop)
[1] 0.5
> sample(pop, 10) # simple random sample of size 10 from pop
 [1] 0 1 0 0 0 0 1 1 0 0
> sample(pop, 10)
 [1] 1 0 0 0 0 0 0 0 1 0
> sample(pop, 10)
 [1] 0 1 0 1 1 0 0 0 1 1
> mean(sample(pop,10)) # sample proportion
[1] 0.6
> mean(sample(pop,10))
[1] 0.7
> mean(sample(pop,10))
[1] 0.6
> # take 100000 SRSs, each of size 10, and record the sample proportion in
each one of them:
> replicate(n=100000,mean(sample(pop,10))) -> P
> hist(P) # make histogram of sample distribution
```




```
> mean(P-0.1<= p & p <= P+0.1) # proportion of SRSs for which [P-0.10,
P+0.10] includes p
[1] 0.65474
> mean(P) # mean of sample distribution. The result should be close to 0.5
because sample proportion is an unbiased estimator of p
[1] 0.50054
> sample(c("H","T"), replace=TRUE, size=100) -> X     # toss a "fair" coin
100 times
> X
[1] "T" "H" "H" "T" "H" "T" "H" "T" "H" "T" "T" "H" "T" "H" "T" "T" "T" "H" "T" "H" "H"
[22] "T" "H" "H" "H" "T" "H" "H" "T" "H" "T" "T" "H" "H" "H" "H" "H" "H" "T" "H" "H" "T"
[43] "H" "T" "H" "T" "H" "T" "H" "H" "H" "T" "T" "H" "T" "H" "H" "T" "H" "T" "H" "T" "T"
[64] "H" "H" "T" "H" "T" "H" "H" "T" "T" "H" "H" "H" "H" "T" "H" "T" "T" "H" "T" "H" "T"
[85] "T" "H" "H" "T" "T" "H" "T" "H" "T" "T" "H" "T" "T" "H" "H" "T"
> mean(X=="H")    # mean number of heads
[1] 0.54
> H <- sum(X=="H")   # count heads
> H
[1] 54
> abs(H-(100-H)) # difference between number of heads and tails
[1] 8
> replicate(n=100000,sum(sample(c("H","T"),replace=TRUE, size=100)=="H"))-> H
> hist(H/100, col="red", breaks=40)    # histogram of sample distribution of
proportion of heads in 100 tosses
```



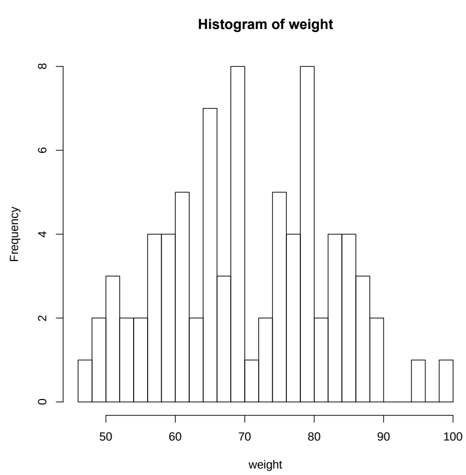
```
> mean(abs(H-(100-H))>=30)   # frequency of groups of 100 tosses in which
heads and tails differ by more than 30
[1] 0.00374
> # Now do the same with a biased coin where frequency of heads = 30%
> replicate(n=100000, sum(sample(c("H","T"), replace=TRUE, size=100,
prob=c(0.3,0.7))=="H")) -> H1
> hist(H1/100, col="red", breaks=40)
```




```
> mean(abs(H1-(100-H1))>=30)
[1] 0.88469
```

Διαστήματα εμπιστοσύνης & έλεγχοι σημαντικότητας για τη μέση τιμή

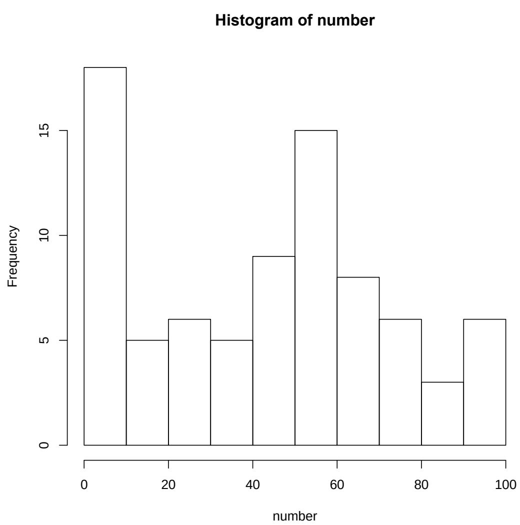
```
> d <- read.table("survey_data_2017.txt", header=TRUE, sep='\t')
> attach(d)
> # construct 95% confidence intervals for mean weight of computer science
students
> # for an assumed standard deviation sigma
> hist(weight, breaks=20)   # check data first!
```




```
> # No outliers, reasonably symmetric, sample size is large, so confidence
```

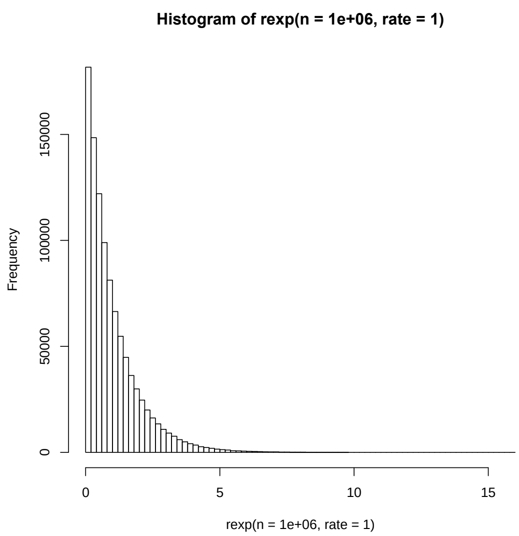
```
interval will be accurate
> sigma = 20 # assumed standard deviation of weight (in Kg)
> mean(weight, na.rm=TRUE) ->x
> x
[1] 70.7625
> m <- sigma*1.96/sqrt(80)  # margin of error
> x-m  # lower limit of confidence interval
[1] 66.37981
> x+m  # upper limit of confidence interval
[1] 75.14519
> abs(qnorm(0.1)) -> z   # Now construct an 80% confidence interval
> m <- sigma*z/sqrt(80)
> x-m
[1] 67.89686
> x+m
[1] 73.62814
> # Now construct a 95% confidence interval using the t distribution
> # i.e., no need to assume standard deviation is known
> # CHECK accurarcy first: weight distribution is reasonably symmetric as
shown above, also the sample size of 80 is large so the following
construction is accurate
> abs(qt(0.025, df=79)) -> t
> t
[1] 1.99045
> mt <- t*sd(weight, na.rm=TRUE)/sqrt(80)  # margin of error
> mt
[1] 2.652626
> x+mt   # upper limit
[1] 73.41513
> x-mt    # lower limit
[1] 68.10987
> # Now let's perform a z-test for:
> # H0: mean height of male students same as population's over the same ages
(known to be 1.7806)
> # Ha: not H0
> sigma <- 0.1   # assumed standard deviation of male height (in meters)
> hm <- height[sex=="M"]   # store male student sampled height
> z <- (mean(hm)-1.7806)/(sigma/sqrt(length(hm)))
[1] -0.2439789
> 2*pnorm(-abs(z))  # p value
[1] 0.8072472 # large, so H0 not rejected
> # Now let's do a t-test for the same thing, i.e., without assuming sigma is
known
> t <- (mean(hm)-1.7806)/(sd(hm)/sqrt(length(hm))) # t statistic
> t
[1] -0.3763833
```

```
> 2*pt(df=length(hm)-1, -abs(t)) # pvalue
[1] 0.7080542
> # large, so again H0 is not rejected
> t.test(hm,mu=1.7806) # perform the same thing using R's command t.test

    One Sample t-test

data:  hm
t = -0.37638, df = 56, p-value = 0.7081
alternative hypothesis: true mean is not equal to 1.7806
95 percent confidence interval:
1.760169 1.794568
sample estimates:
mean of x
1.777368

> # Let's check if numbers picked in survey from 1 to 100 are likely to have
mean 50
> # First look data to see if suitable to apply t test
> hist(number)
```



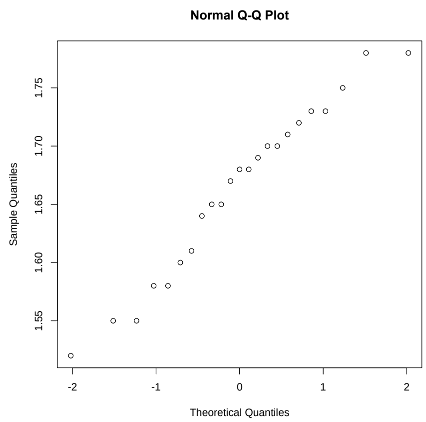

```
> # Asymmetric, but no outliers. Should be ok because sample size (80) is
large
> qqnorm(number)
> hist(number)
> t.test(number, mu=50)

    One Sample t-test

data:  number
t = -1.7296, df = 80, p-value = 0.08756
alternative hypothesis: true mean is not equal to 50
```

```
95 percent confidence interval:
37.86647 50.84958
sample estimates:
mean of x
44.35802

> # Now let’s see how accurate our confidence intervals really are:
> # First take 100000 samples of size 10 and compute 95% confidence intervals
> # For testing purposes we use a population distributed as N(1,1)
> t <- abs(qt(df=length(hm),0.025)) # 95% confidence level
> t
[1] 2.002465
> replicate(n=100000, c((mu<-mean(X<-rnorm(mean=1,n=10)))-(M<-
t*sd(X)/sqrt(10)), mu+M)) ->ci
> mean(ci[1,]<= 1 & ci[2,]>= 1)  # proportion of intervals containing the
true mean
[1] 0.95072
> # What if we sampled from an exponential distribution of mean 1?
> hist(rexp(n=1000000, rate=1), breaks=100) # see how it looks like…
```

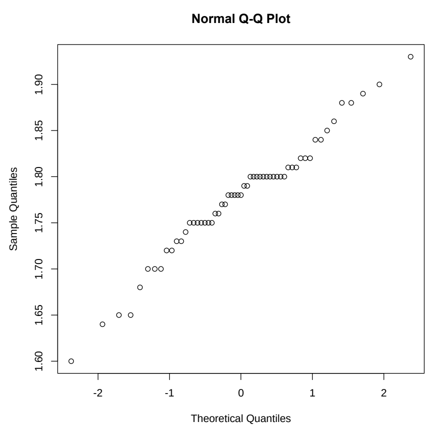



```
> replicate(n=100000, c((mu<-mean(X<-rexp(rate=1,n=10)))-(M<-
t*sd(X)/sqrt(10)), mu+M)) ->ci
> mean(ci[1,]<= 1 & ci[2,]>= 1)
[1] 0.90084
> # true confidence level is not 95%!!!
> # What if we use a larger sample size?
> replicate(n=100000, c((mu<-mean(X<-rexp(rate=1,n=80)))-(M<-
t*sd(X)/sqrt(80)), mu+M)) ->ci
> mean(ci[1,]<= 1 & ci[2,]>= 1)
[1] 0.94145
> # OK, that’s closer to 95%
>
```

```
> # Now let's check how accurate p values really are for the t-test:
> # H0: mean = 1, Ha: mean not 1
> # First use a N(1,1) population, sample size=10
> X <- rnorm(mean=1, n=10)
> sd(X)
[1] 0.7497415
> tt <- (mean(X)-1)/(sd(X)/sqrt(10)) # t statistic
> tt
[1] 0.3768549
> pt(df=9,-tt)*2 # p value as given by t distribution
[1] 0.7150132
> # lets confirm this experimentally:
> replicate(n=100000, (mean(X<-rnorm(mean=1,n=10))-1)/(sd(X)/sqrt(10))) -> T
> sum(abs(T)>=tt)
[1] 715689
> # this is accurate, as predicted by theory: (t statistic has the t
distribution when sampling from normal distribution)
> # How accurate the p value of 0.7150132 will be for the exponential
distribution?
> X <- rexp(rate=1, n=10) # take a n=10 sample from exponential distribution
with mean 1
> tt <- (mean(X)-1)/(sd(X)/sqrt(10)) # t statistic
> tt
[1] -1.511693
> pt(df=9,-abs(tt))*2   # p value using t distribution
[1] 0.1649003
> replicate(n=1000000, (mean(X<-rexp(rate=1,n=10))-1)/(sd(X)/sqrt(10))) -> T
> mean(abs(T)>=abs(tt))  # true p value
[1] 0.204814
> # Not accurate! Let's try larger samples
> X <- rexp(rate=1, n=100)
> tt <- (mean(X)-1)/(sd(X)/sqrt(100))
> pt(df=99,-abs(tt))*2
[1] 0.5207039
> tt
[1] -0.6445588
> replicate(n=1000000, (mean(X<-rexp(rate=1,n=100))-1)/(sd(X)/sqrt(100))) ->
T
> mean(abs(T)>=abs(tt))  # true p value
[1] 0.523602
> # OK, that's better!
> qqnorm(height[sex=="F"])
```




```
> qqnorm(height[sex=="M"])
```




```
> t.test(height[sex=="M"], height[sex=="F"])

      Welch Two Sample t-test

data:  height[sex == "M"] and height[sex == "F"]
t = 6.4659, df = 36.313, p-value = 1.602e-07
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.07847657 0.15017331
sample estimates:
mean of x mean of y
 1.777368  1.663043
```

Διαστήματα εμπιστοσύνης & έλεγχοι σημαντικότητας για ποσοστά

```
> # get a simple random sample of size 10 from a binary categorical
variable of a population with 30% ‘0’’s and 70% ‘1’’s:
> sample(c(0,1), 10, replace=TRUE, prob=c(0.3,0.7))
[1] 1 1 0 1 1 1 1 1 1 1
```

```
> # evaluate the accuracy of 90% confidence interval for the true proportion
of '1''s (=70%)
> replicate(n=100000, mean(sample(c(0,1), 10, replace=TRUE,
prob=c(0.3,0.7)))) -> P
> z <- abs(qnorm(0.05))
> z
[1] 1.644854
> m <- z*sqrt(P*(1-P)/10) # margin of error for sample size 10
> mean(P-m<= 0.7 & 0.7 <= P+m) # compute proportion of intervals containing
the true proportion 70%
[1] 0.80316
> # only 80.3%(!) instead of 90%; This is because the confidence level is
only approximately correct. It converges to correct value for higher sample
size. So let's try a sample size of 100:
> replicate(n=100000, mean(sample(c(0,1), 100, replace=TRUE,
prob=c(0.3,0.7)))) -> P
> m <- z*sqrt(P*(1-P)/100)
> mean(P-m<= 0.7 & 0.7 <= P+m)
[1] 0.87068
> # true confidence level is closer to 90% but not there yet. Let's try an
even bigger sample size
> replicate(n=100000, mean(sample(c(0,1), 1000, replace=TRUE, prob=c(0.3,0.7)))) -> P
> m <- z*sqrt(P*(1-P)/1000)
> mean(P-m<= 0.7 & 0.7 <= P+m)
[1] 0.9029
> # confidence level is close to what promised (90%) for large sample size;
here 1000 is sufficient
> # Now let's perform the test: H0: proportion of '1's=70%. (For sample
size=1000.)
> # Test is perform multiple times to see how often null hypothesis is
(uncorrectly) rejected (i.e., Type I error). We use significance level=10%
> z <- (P-0.7)/sqrt(0.7*(1-0.7)/1000) # value of z statistic
> mean(abs(z)>zstar)  # proportion of tests uncorrectly rejecting the null
hypothesis
[1] 0.1037
> # this is close to the significance level. This is expected, as Type I
error=significance level in theory
> # check the same thing but for sample size 10:
> replicate(n=100000, mean(sample(c(0,1), 10, replace=TRUE,
prob=c(0.3,0.7)))) -> P
> z <- (P-0.7)/sqrt(0.7*0.3/10)
> mean(abs(z)>zstar)
[1] 0.07579
> # Type I error and significance level diverge. This is because p value is
only approximate for small sample sizes
```

Διαστήματα εμπιστοσύνης & έλεγχοι σημαντικότητας για σχέση μεταξύ δύο κατηγορικών μεταβλητών

```
> # We first test if the proportion of students which have passed
"Probability" is related to sex, for the population of students enrolled in
"Statistics in Probability" in the past 10 years
> # Will treat the responses to 2017 survey as a random sample from the above
population
> d <- read.table("survey_data_2017.txt", header=TRUE, sep='\t')
> attach(d)
> prob >= 5 & !is.na(prob) -> pass
> table(pass, sex) ->tps
> tps
       sex
pass     F  M
  FALSE  8 11
  TRUE  16 46
> prop.table(tps, margin=2)
       sex
pass            F         M
  FALSE 0.3333333 0.1929825
  TRUE  0.6666667 0.8070175
> addmargins(tps)
       sex
pass     F  M Sum
  FALSE  8 11  19
  TRUE  16 46  62
  Sum   24 57  81
> n1 <- 57 # number of male students sampled
> n2 <- 24 # number of female students sampled
> prop.table(tps, margin=2)["TRUE","M"]->p1   # sample proportion of passed
male students
> p1
[1] 0.8070175
> prop.table(tps, margin=2)["TRUE","F"]->p2 # same for female students
> p2
[1] 0.6666667
> addmargins(prop.table(tps))["TRUE","Sum"]->p # sample proportion of passed
students regardless of their sex
> p
[1] 0.7654321
> z <- (p1-p2)/sqrt(p*(1-p)*(1/n1+1/n2))  # z statistic
> z
[1] 1.361219
> 2*pnorm(-abs(z))  # p value
[1] 0.1734445
> # Not very small as to reject null hypothesis.
> # Now construct a 90% confidence interval for estimating the difference in
pass rates between male and females:
> zstar <- abs(qnorm(0.05))
```

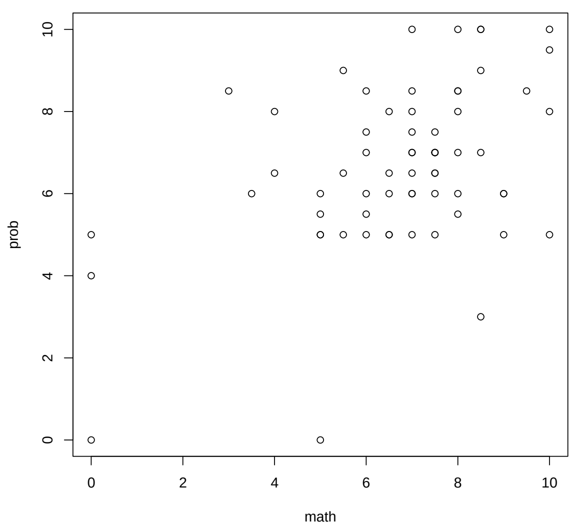
```
> zstar
[1] 1.644854
> m <- zstar*sqrt(p1*(1-p1)/n1+p2*(1-p2)/n2) # margin of error
> m
[1] 0.1801212
> p1-p2+m*c(-1,1)  # the confidence interval
[1] -0.03977031  0.32047207
> chisq.test(tps) # chi-squared test for the same thing as the z-test above

      Pearson's Chi-squared test with Yates' continuity correction

data:  tps
X-squared = 1.1537, df = 1, p-value = 0.2828

> # Yates continuity applied. (Happens automatically in small samples.) Try
without it:
> chisq.test(tps, correct=FALSE)

      Pearson's Chi-squared test

data:  tps
X-squared = 1.8529, df = 1, p-value = 0.1734
> # Notice p value is same as in the z-test. This is always true in 2x2
tables
> z^2  # and X-squared statistic same as z squared.
[1] 1.852918
> # Now test if selection of color (dark/light) is related to the season one
was born in
> dark <- color=="black" | color=="blue" | color=="purple" | color=="green"
> spring <- month == "mar" | month == "apr" | month == "may"
> summer <- month == "jun" | month == "jul" | month == "aug"
> fall <- month == "sep" | month == "oct" | month == "nov"
> winter <- month == "jan" | month == "feb" | month == "dec"
> seasoni <- 1*spring+2*summer+3*fall+4*winter
> seasonnames <- c("spring", "summer", "fall", "winter")
> as.factor(seasonnames[seasoni[seasoni>0]]) ->seasons
> dark[seasoni>0]-> dark
> table(dark,seasons)
       seasons
dark    fall spring summer winter
  FALSE    6      5      3      4
  TRUE    13     19     16     13
> chisq.test(table(dark, seasons))

      Pearson's Chi-squared test

data:  table(dark, seasons)
X-squared = 1.421, df = 3, p-value = 0.7006

Warning message:
In chisq.test(table(dark, seasons)) :
  Chi-squared approximation may be incorrect
```

```
> # Warning message due to small numbers in table entries. P-value may be
incorrect - compute the true p-value:
> chisq.test(table(dark, seasons), simulate.p.value=TRUE, B=1000000)

	Pearson's Chi-squared test with simulated p-value (based on 1e+06
	replicates)

data:  table(dark, seasons)
X-squared = 1.421, df = NA, p-value = 0.7093

> # p-value is high so no relation is detected
>
> # Now test if numbers is survey were selected uniformly from 1 to 100,
using a chi-squared goodness of fit test:
> # (Numbers are split into 5 groups: 1-20 (group 1), 21-40 (group 2), etc.)
> as.factor(floor((number-1)/20)+1) -> group

> table(group)

group
 1  2  3  4  5
23 11 24 14  9
> prop.table(table(group))
group
        1         2         3         4         5
0.2839506 0.1358025 0.2962963 0.1728395 0.1111111
> chisq.test(table(group), p=c(1/5, 1/5, 1/5, 1/5, 1/5))

	Chi-squared test for given probabilities

data:  table(group)
X-squared = 11.778, df = 4, p-value = 0.01908

> chisq.test(table(group)) # same as above; no need to specify the
distribution in a parameter, if checking against a uniform

	Chi-squared test for given probabilities

data:  table(group)
X-squared = 11.778, df = 4, p-value = 0.01908
> table(group[sex == "F"])

1 2 3 4 5
9 4 8 1 2
> chisq.test(table(group[sex=="F"]))  # test if women choose numbers
uniformly

	Chi-squared test for given probabilities

data:  table(group[sex == "F"])
X-squared = 10.583, df = 4, p-value = 0.03167

Warning message:
In chisq.test(table(group[sex == "F"])) :
  Chi-squared approximation may be incorrect

> chisq.test(table(group[sex=="F"]), simulate.p.value=TRUE, B=10000)
```

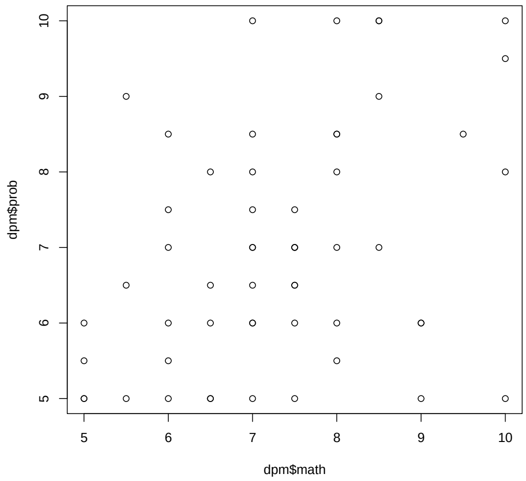
```
	Chi-squared test for given probabilities with simulated p-value (based
on
	10000 replicates)

data:  table(group[sex == "F"])
X-squared = 10.583, df = NA, p-value = 0.0349
```

Συμπερασματολογία για γραμμική παλινδρόμηση

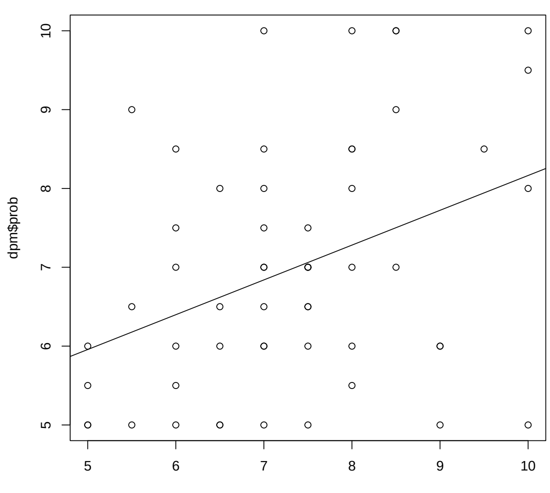
```
> # Make inferences about the relation between grade in mathematics
> # (math) and grade in probability (prob), in the population consisting
> # of students who would have taken the Statistics in Informatics class.
> # Load sample data (survey)
> d <- read.table("survey_data_2017.txt", header=TRUE, sep="\t")
> attach(d)
> plot(prob~math)
```




```
> #create a dataframe consisting of students who have passed both prob & math
> dpm <- d[prob>=5 & math >=5 & !is.na(prob) & !is.na(math),]
> plot(dpm$prob~dpm$math)
```

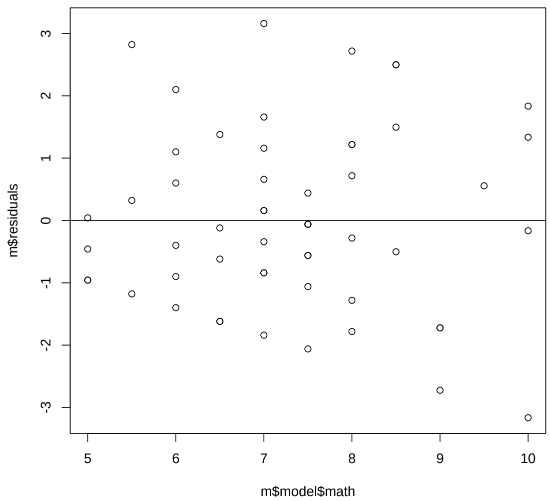



```
> m <- lm(prob~math, data=dpm) # linear least-squares regression
> m

Call:
lm(formula = prob ~ math, data = dpm)

Coefficients:
(Intercept)         math
     3.7504       0.4414
> coefficients(m) # intercept (b0) and slope (b1) coefficients
(Intercept)        math
  3.7503737   0.4414183
> coef(m) # abbreviation of the above
(Intercept)        math
  3.7503737   0.4414183
> abline(m)
```

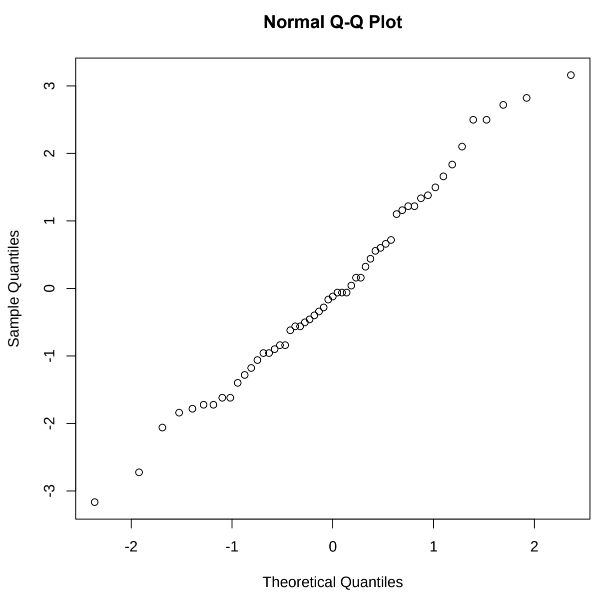



```
> # A linear relation between math and prob (in the population) is not ruled
out
```

```
> # Check homoscedasticity
> plot(m$model$math,m$residuals)
> abline(0,0)
```



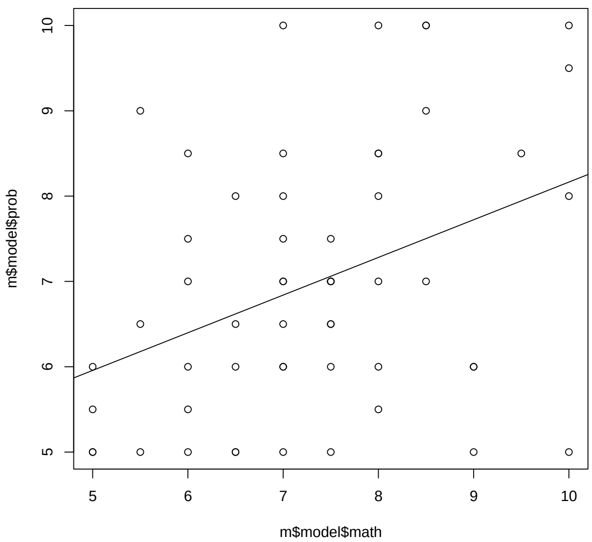

```
> # Not severely violated. Now check normality:
> qqnorm(m$residuals)
```



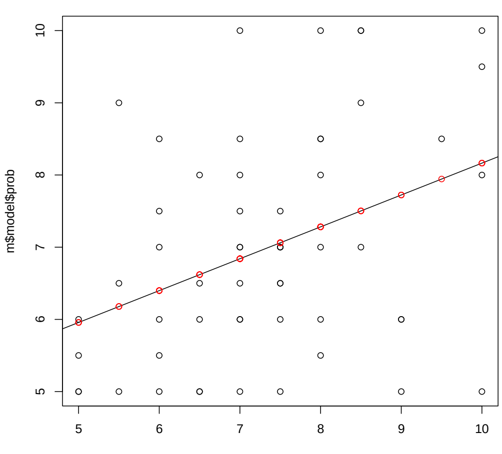

```
> # Close enough.
> summary(m)

Call:
lm(formula = prob ~ math, data = dpm)

Residuals:
    Min      1Q  Median      3Q     Max 
-3.1646 -0.9575 -0.1196  1.1304  3.1597 

Coefficients:
```

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.7504     1.0996   3.411  0.00125 **
math           0.4414     0.1482   2.979  0.00436 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.462 on 53 degrees of freedom
Multiple R-squared:  0.1434,  Adjusted R-squared:  0.1272
F-statistic: 8.873 on 1 and 53 DF,  p-value: 0.00436

> summary(m)$coefficients # table of estimate data
             Estimate Std. Error  t value    Pr(>|t|)
(Intercept) 3.7503737  1.0996123 3.410633 0.001246998
math        0.4414183  0.1481921 2.978690 0.004359645
> summary(m)$coefficients["math","Estimate"]->b1
> b1 # estimate of slope
[1] 0.4414183
> # Produce a 95% confidence interval for slope parameter:
> summary(m)$coefficients["math","Std. Error"]->SEb1
> SEb1 # standard error for slope estimator
[1] 0.1481921
> n <- length(m$residuals)
> n
[1] 55
> t <- abs(qt(df=n-2, 0.025))
> t
[1] 2.005746
> b1+t*SEb1*c(-1,1) # the confidence interval
[1] 0.1441826 0.7386540
> confint(m) # same thing (1st line: intercept, 2nd line: slope)
                2.5 %   97.5 %
(Intercept) 1.5448307 5.955917
math        0.1441826 0.738654
> confint(m, level=0.9) # 90% confidence intervals
                  5 %      95 %
(Intercept) 1.9094949 5.5912526
math        0.1933275 0.6895091
> predict(m, newdata=data.frame(math=5)) # estimate mean prob grade between students with math==5 (in the population)
       1
5.957465
> predict(m, newdata=data.frame(math=5), interval="confidence") # add 95% confidence interval
       fit      lwr      upr
1 5.957465 5.167741 6.747189
> predict(m, newdata=data.frame(math=5), interval="prediction")# 95% prediction interval
       fit      lwr      upr
1 5.957465 2.921015 8.993915
> predict(m, newdata=data.frame(math=c(5,5.5,6)), interval="prediction")
       fit      lwr      upr
1 5.957465 2.921015 8.993915
2 6.178174 3.171696 9.184653
```

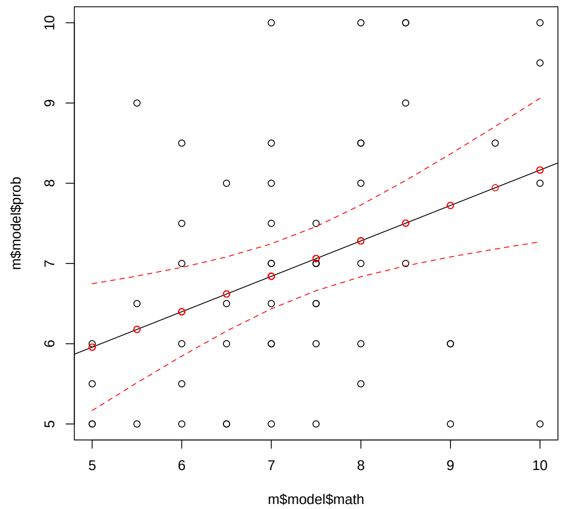
```
3 6.398883 3.415266 9.382501
> plot(m$model$prob~m$model$math)
> abline(m)
```



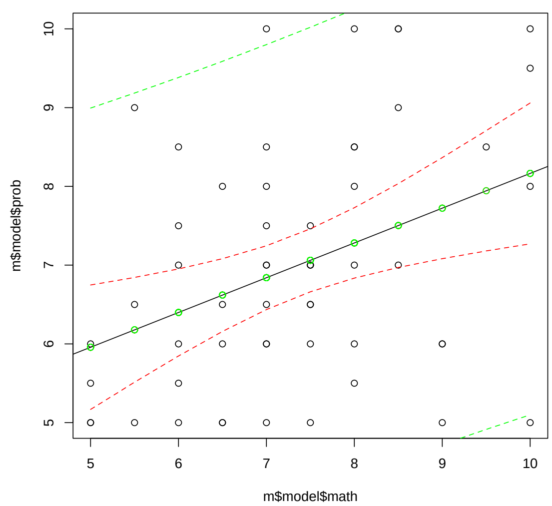

```
> x <- sort(m$model$math)
> x
> y <- predict(m, newdata=data.frame(math=x), interval="confidence")
> y[1:5,] % fitted values and confidence interval for mean response
       fit      lwr      upr
1 5.957465 5.167741 6.747189
2 5.957465 5.167741 6.747189
3 5.957465 5.167741 6.747189
4 5.957465 5.167741 6.747189
5 6.178174 5.512931 6.843418
> points(x,y[,"fit"], col="red") # plot fitted values
```




```
> lines(x,y[,"lwr"], col="red", lty="dashed") # plot lower confidence intvls.
> lines(x,y[,"upr"], col="red", lty="dashed") # plot upper confidence intvls.
```



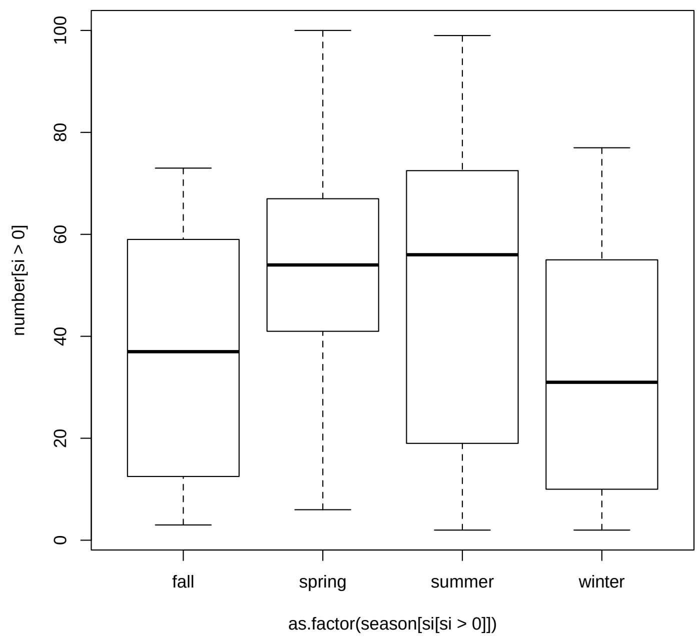

```
> yp <- predict(m, newdata=data.frame(math=x), interval="prediction")
> points(x,yp[,"fit"], col="green") # plot predicted values
> lines(x,yp[,"upr"], col="green", lty="dashed") #plot upper prediction ivl.
> lines(x,yp[,"lwr"], col="green", lty="dashed") #plot lower prediction ivl.
```



Ανάλυση Διακύμανσης (ANOVA)

```
> # Make inferences about the relation between season of birth and chosen
number between 1 and 100 for the population of student that have or will ever
take Statistics in Informatics class
> # use birth month (month variable) to calculate season
> winter = month == "jan" | month == "feb" | month == "dec"
> fall = month == "sep" | month == "oct" | month == "nov"
> summer = month == "jun" | month == "jul" | month == "aug"
> spring = month == "mar" | month == "may" | month == "apr"
> si <- spring+2*summer+3*fall+4*winter
> si
```

```
 [1] 3 3 3 1 3 4 1 1 1 3 1 4 3 4 1 1 2 1 2 3 4 4 2 1 2 4 4 2 3 1 2 3 4 3 3 2
1 2 1 2
[41] 1 2 2 2 1 3 4 4 2 3 0 3 4 1 4 2 3 3 2 4 1 1 1 2 2 4 2 3 1 4 1 4 3 1 1 1
1 0 4 3
[81] 2
> # Right now si is the season index: 1 for spring, 2 for summer etc.
> season <- c("spring", "summer", "fall", "winter")
> # plot relation using side-by-side boxplots
> plot(number[si>0]~as.factor(season[si[si>0]]))
```

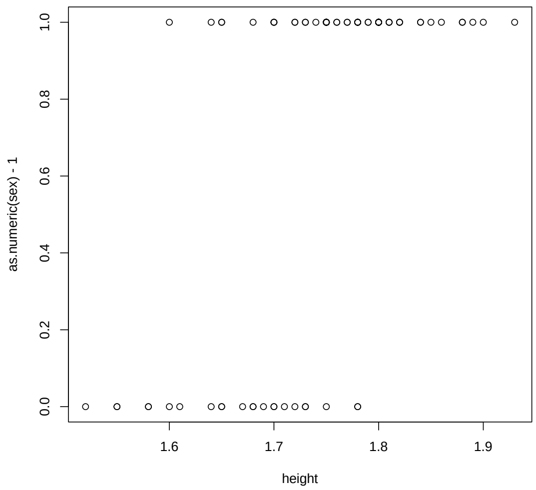



```
> tapply(number[si>0], as.factor(season[si[si>0]]), mean) # group means
    fall   spring   summer   winter
35.52632 51.83333 50.26316 36.35294
> tapply(number[si>0], as.factor(season[si[si>0]]), sd) # group standard dev.
    fall   spring   summer   winter
25.45447 26.60936 33.32640 26.76271
> # Maximum standard dev. does not exceed twice the minimum standard dev.
> # so homoscedasticity is assumed to hold
> aov(number[si>0]~as.factor(season[si[si>0]])) -> ares # perform ANOVA
> anova(ares) # display ANOVA table
Analysis of Variance Table

Response: number[si > 0]
                              Df Sum Sq Mean Sq F value Pr(>F)
as.factor(season[si[si > 0]])  3   4573  1524.4  1.9247 0.1328
Residuals                     75  59400   792.0
> SSG <- anova(ares)[1,2]
> SSG
[1] 4573.123
> SSE <- anova(ares)[2,2]
> DFG <- anova(ares)[1,1]
```

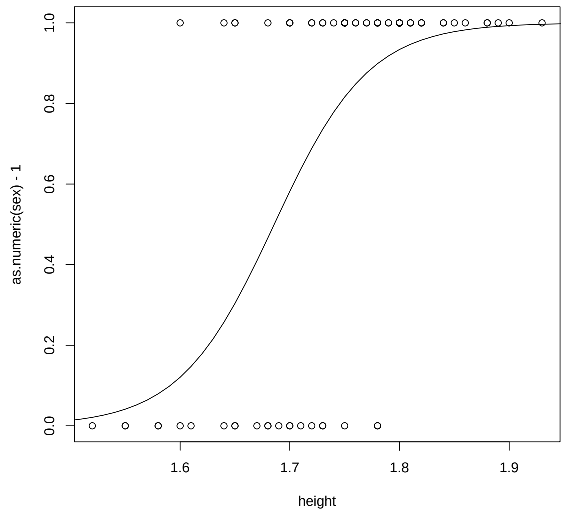
```
> DFG
[1] 3
> DFE <- anova(ares)[2,1]
> MSE <- SSE/DFE
> SST <- SSG+SSE
> DFT <- DFG+DFE
> MSG <- SSG/DFG
> MSG
[1] 1524.374
> anova(ares)[1,3] # MSG is also contained in the ANOVA table above
[1] 1524.374
> anova(ares)[1,"F value"] # F-value is also contained in the ANOVA table
[1] 1.924727
> F <- MSG/MSE # Just check if it agrees with calculation by hand
> F
[1] 1.924727
> 1-pf(F, df1=DFG, df2=DFE) # p-value for ANOVA F-test
[1] 0.132816
> anova(ares)[1,"Pr(>F)"] # p-value is already calculated in the ANOVA table
[1] 0.132816
> # p-value is not too small, so the null hypothesis, i.e., season means are
the same, cannot be rejected
> # Notice that pairwise t-tests may (erroneously) reject the null
hypothesis!
> t.test(number[si==1], number[si==3])

      Welch Two Sample t-test

data:  number[si == 1] and number[si == 3]
t = 2.0447, df = 39.487, p-value = 0.04759
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
  0.1819946 32.4320404
sample estimates:
mean of x mean of y
 51.83333  35.52632
> # This is the reason ANOVA test is useful!
```

Λογιστική παλινδρόμηση

```
> # We will make inferences about the relation of height as the
> # explanatory variable, and sex as the response variable.
> # Population: students of informatics
> # First let's plot the data (from survey)
> plot(height, as.numeric(sex)-1)
```

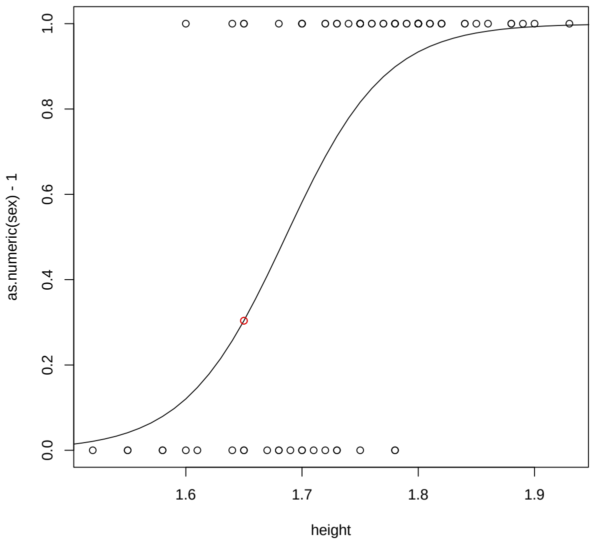



```
> m <- glm(sex~height, family=binomial("logit")) # apply logistic regression
> m

Call:  glm(formula = sex ~ height, family = binomial("logit"))

Coefficients:
(Intercept)       height  
     -39.09        23.19  

Degrees of Freedom: 79 Total (i.e. Null);  78 Residual
  (1 observation deleted due to missingness)
Null Deviance:        95.98 
Residual Deviance: 61.46      AIC: 65.46
> coefficients(m) # returns coefficients of the logistic regression
(Intercept)      height 
  -39.09015    23.18809 
> # Now let’s plot the fitted model against the data
> x <- seq(from=1.5, to=2, by=0.01) # range of height values
> # get fitted values for each height in x
> y <- predict(m, newdata=data.frame(height=x), type="response")
> lines(x,y)
```

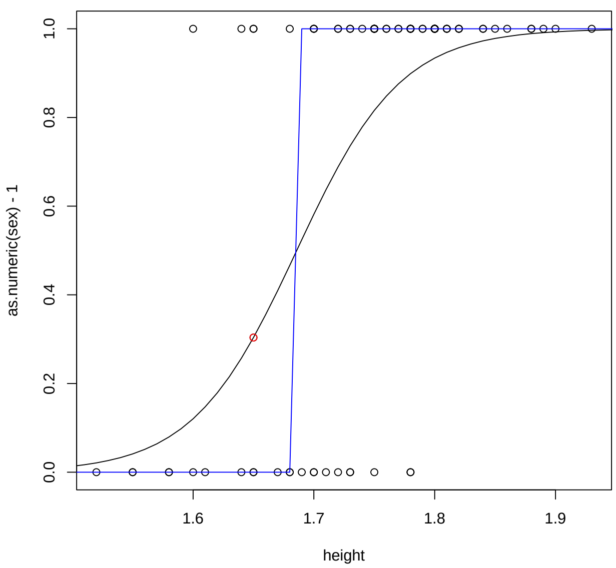



```
> summary(m) # the coefficients are also given by the summary function

Call:
glm(formula = sex ~ height, family = binomial("logit"))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1407  -0.3190   0.3698   0.5898   2.0579

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -39.090      9.290  -4.208 2.58e-05 ***
height        23.188      5.407   4.289 1.80e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 95.984  on 79  degrees of freedom
Residual deviance: 61.457  on 78  degrees of freedom
  (1 observation deleted due to missingness)
AIC: 65.457

Number of Fisher Scoring iterations: 5
> b1 <- summary(m)$coefficients[2,"Estimate"] # slope estimate
> b1
[1] 23.18809
> SEb1 <- summary(m)$coefficients[2,"Std. Error"] # standard error of slope
estimate
> SEb1
[1] 5.406553
> # Let's construct 95% confidence intervals for the slope (in population)
> z <- abs(qnorm(0.025))
```

```
> z
[1] 1.959964
> b1+z*SEb1*c(-1,1) # the confidence interval
[1] 12.59144 33.78474
> confint.default(m) # same as above; note that confint(m) yields a different
interval!
                2.5 %    97.5 %
(Intercept) -57.29730 -20.88299
height       12.59144  33.78474
Waiting for profiling to be done...
> exp(confint.default(m))[2,] # confidence interval for odds ratio
       2.5 %       97.5 %
2.940320e+05 4.704648e+14
> # Perform significance test for existence of relation
> z <- b1/SEb1 # z test statistic
> z
[1] 4.288887
> summary(m)$coefficients[2,"z value"] # same as above
[1] 4.288887
> summary(m)

Call:
glm(formula = sex ~ height, family = binomial("logit"))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1407  -0.3190   0.3698   0.5898   2.0579

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  -39.090      9.290  -4.208 2.58e-05 ***
height        23.188      5.407   4.289 1.80e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 95.984  on 79  degrees of freedom
Residual deviance: 61.457  on 78  degrees of freedom
  (1 observation deleted due to missingness)
AIC: 65.457

Number of Fisher Scoring iterations: 5
> # p-value is 1.80e-05
> # Estimate the proportion of males in the population with height 1.65m
> predict(m, data.frame(height=1.65), type="response")-> y
> y
        1
0.3036888
> points(1.65, y, col="red")
```




```
> round(y) # Predict the sex of an individual with height 1.65m (1=male,
0=female)
1
0
> # Make predictions of sex for all heights in x
> yp <- round(predict(m, data.frame(height=x), type="response"))
> lines(x,yp, col="blue")
```




```
> # Predict male if and only if > 1.69
```

Τερματισμός και έξοδος από το πρόγραμμα

```
> q()
```