LAB 03

TOPIC:

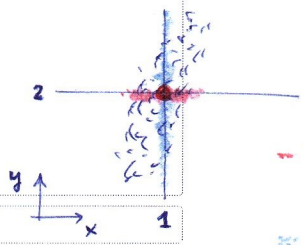
- Principal Component Analysis

```
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### PCA
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### Principal component analysis of a simulated dataset
# Generate the data
library(mvtnorm)
library(ellipse)
mu  <- c(1,2)
sig <- cbind(c(1,1), c(1,4))
n   <- 100

set.seed(27032020)
X <- rmvnorm(n, mu, sig)
```

← library that allows to generate multivariate gaussian data

n = how many vectors we want

```
# we plot the data
x11()
plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=19, xlim=c(-6,6), ylim=c(-5,10))

# we plot the sample mean
points(colMeans(X)[1], colMeans(X)[2], col='red', pch=19,lwd=3)

# we plot the projection on the x- an y-axis and compute their variance
abline(h=colMeans(X)[2], lty=2, col='grey')
points(X[,1], rep(colMeans(X)[2], n), col='red')
var(X[,1])
```



= projection of the data along the x direction

= projection of the data along the y direction

```
## [1] 1.008267
```

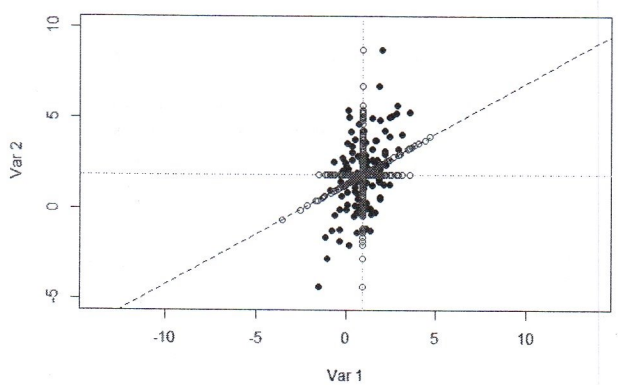
```
abline(v=colMeans(X)[1], lty=2, col='grey')
points(rep(colMeans(X)[1], n), X[,2], col='red')
var(X[,2])
```

```
## [1] 4.782997
```

```
# let's compute the variance along all the directions
theta   <- seq(0, pi, by = 2*pi/360)
Var     <- NULL

# Example: along the direction with angle theta:
theta[30]
```

```
## [1] 0.5061455
```

```
abline(a = colMeans(X)[2] - tan(theta[30])*colMeans(X)[1], b = tan(theta[30]), lty=2)
a <- c(cos(theta[30]), sin(theta[30]))
proj30=(a)%*%(t(X)-colMeans(X))
points(colMeans(X)[1]+cos(theta[30])*proj30,
       colMeans(X)[2]+sin(theta[30])*proj30, col='red')
```




```
var(X %*% a)
```

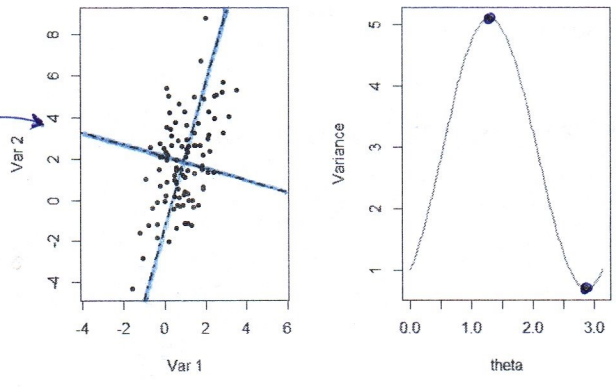
```
##          [,1]
## [1,] 2.861341
```

```
# For cycle
for(i in 1:length(theta))  # for i between 1 and length(theta) ripeat:
{
  a   <- c(cos(theta[i]), sin(theta[i])) # unit vector in direction theta[i]
  v   <- var(X %*% a)                    # sample variance of the projection of X along the direction identified by vector a
  Var <- c(Var, v)
}

# we can compute the min and max of the variance
max.var   <- max(Var)
max.theta <- theta[which.max(Var)]
max.a     <- c(cos(max.theta), sin(max.theta))

min.var   <- min(Var)
min.theta <- theta[which.min(Var)]
min.a     <- c(cos(min.theta), sin(min.theta))
```

```
# Graphically:
x11(width=10, height=7)
par(mfrow=c(1,2))
plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=20)
abline(a = colMeans(X)[2] - tan(max.theta)*colMeans(X)[1], b = tan(max.theta), lty = 4, col = 'navyblue', lwd = 2)
abline(a = colMeans(X)[2] - tan(min.theta)*colMeans(X)[1], b = tan(min.theta), lty = 4, col = 'blue', lwd = 2)

plot(theta, Var, type = 'l', col='dark grey', lwd = 2,ylab='Variance')
points(max.theta, max.var, pch=16, col='navyblue')
points(min.theta, min.var, pch=16, col='blue')
```

direction for the maximum variance

direction for the minimum variance




```
### ------------------------------------------------------------------------
### Let's verify the theory
# We compute the sample covariance matrix

M <- colMeans(X)
S <- cov(X)

# We compute the eigenvectors and eigenvalues
eigen(S)
```

covariance matrix

```
## eigen() decomposition
## $values
## [1] 5.1000111 0.6912528
##
## $vectors
##           [,1]       [,2]
## [1,] 0.2681522 -0.9633766
## [2,] 0.9633766  0.2681522
```

```
# Note. eigen(S)$vectors returns a matrix whose columns are the eigenvectors of S

# We compare with the values and directions of max/min variability
# We found empirically
max.var
```

```
## [1] 5.099744
```

```
min.var
```

```
## [1] 0.6915196
```

very similar! Not the same since we did a discretization but very good (very similar to the values obtained with the eigenvalues and eigenvectors of S)

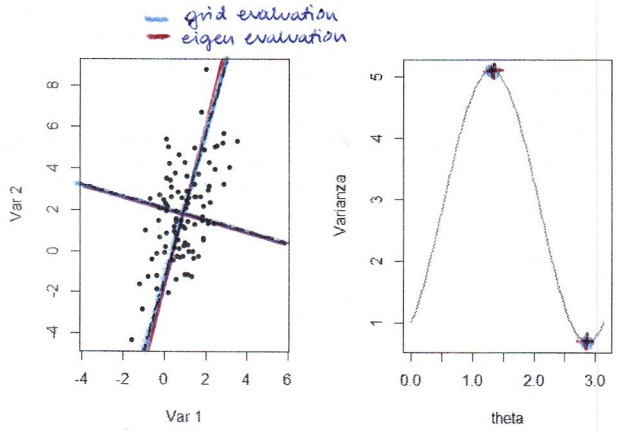
```
max.a
```

```
## [1] 0.2756374 0.9612617
```

```
min.a
```

```
## [1] -0.9612617  0.2756374
```

```
# Let's plot the directions of max/min variability
par(mfrow=c(1,2))
plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=20)
ellipse(M, S, 1, add=T,lwd=3)

abline(a = M[2] - eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1]*M[1], b = eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1], lty = 2
, col = 'dark red', lwd = 2)
abline(a = M[2] - eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2]*M[1], b = eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2], lty = 2
, col = 'red', lwd = 2)
abline(a = M[2] - tan(max.theta)*M[1], b = tan(max.theta), lty = 4, col = 'navyblue', lwd = 2)
abline(a = M[2] - tan(min.theta)*M[1], b = tan(min.theta), lty = 4, col = 'blue', lwd = 2)

plot(theta, Var, type = 'l', col='dark grey', lwd = 2,ylab='Varianza')
points(max.theta, max.var, pch=20, col='navyblue')
points(min.theta, min.var, pch=20, col='blue')
points(atan(eigen(S)$vector[2,1]/eigen(S)$vector[1,1]), max.var, pch=3, col='dark red')
points(atan(eigen(S)$vector[2,2]/eigen(S)$vector[1,2])+pi, min.var, pch=3, col='red')
```



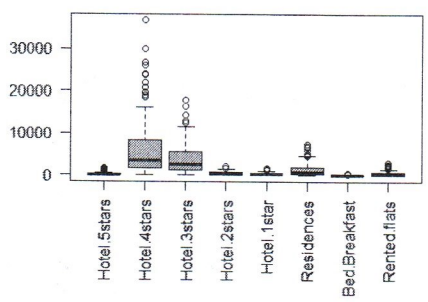

```
graphics.off()
```

```
### ------------------------------------------------------------------------
### Principal component analysis of the dataset 'tourists'

# The dataset "tourists.txt" collects the data on the flow of Italian tourism
# from outside Lombardy to Milan for the year 2015. Each statistical unit
# corresponds to a Region of origin and a month of observation. For each unit,
# the tourists' flow is quantified through the number of nights spent by clients
# in: `5 stars hotels', `4 stars hotels', `3 stars hotels', `2 stars hotels',
# `1 star hotels', `residences', `B&B' and `rented flats'.

tourists <- read.table('tourists.txt', header=T)
head(tourists)
```

```
##   Month Region.of.origin Hotel.5stars Hotel.4stars Hotel.3stars Hotel.2stars
## 1   Jan         PIEMONTE          255         5733         3878         351
## 2   Feb         PIEMONTE          277         6613         3816         352
## 3   Mar         PIEMONTE          272         7278         4518         523
## 4   Apr         PIEMONTE          181         5311         3550         383
## 5   May         PIEMONTE          301         5885         3231         396
## 6   Jun         PIEMONTE          259         5436         3115         456
##   Hotel.1star Residences Bed.Breakfast Rented.flats
## 1         504        1489           41         551
## 2         380        1617           31         415
## 3         475        1949           42         520
## 4         330        1853           42         542
## 5         421        2050           85         561
## 6         301        2273           64         587
```

→ we'll focus the PCA on the columns

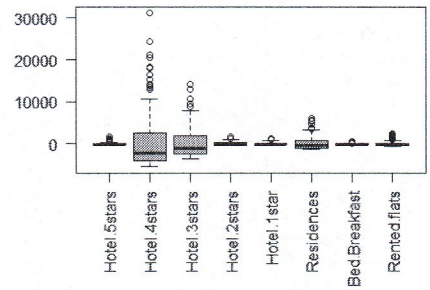
```
tourists.label <- tourists[,1:2]
tourists <- tourists[,-(1:2)]

n <- dim(tourists)[1]
p <- dim(tourists)[2]

# Boxplot
x11()
par(mar=rep(8,4))
boxplot(tourists, las=2, col='gold')
```



← we can see that some variables have a wilder variability

```
# We observe that the variability of the number of nights in 3,4 stars hotels and residences is higher than that of the othe
rs. This may influence the PCA

# Note: PCA is not about the mean, it is about the variability
x11()
par(mar=rep(8,4))
boxplot(scale(x=tourists,center = T, scale=F), las=2, col='gold')
```

→ we put the mean = 0 for all of them (basically we translate the boxplot) to see if there is visually difference in the variability



we can clearly see that the variability is different in some of them

```
# We perform the PCA on original data
pc.tourists <- princomp(tourists, scores=T)
pc.tourists
```

```
## Call:
## princomp(x = tourists, scores = T)
##
## Standard deviations:
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 6573.01508  824.72074  364.23534  162.54342  117.61268   64.14162   61.48464
##     Comp.8
##   34.06981
##
##  8  variables and  240 observations.
```

```
summary(pc.tourists)
```

```
## Importance of components:
##                             Comp.1       Comp.2       Comp.3       Comp.4
## Standard deviation     6573.0150832 824.72074094 3.642353e+02 1.625434e+02
## Proportion of Variance    0.9804356   0.01543489 3.010606e-03 5.995544e-04
## Cumulative Proportion     0.9804356   0.99587044 9.988811e-01 9.994806e-01
##                             Comp.5       Comp.6       Comp.7       Comp.8
## Standard deviation     1.176127e+02 6.414162e+01 6.148464e+01 3.406981e+01
## Proportion of Variance 3.139049e-04 9.336190e-05 8.578730e-05 2.634082e-05
## Cumulative Proportion  9.997945e-01 9.998879e-01 9.999737e-01 1.000000e+00
```

- standard deviation of each component
- proportion of variance = variance explained by each component divided by the total variance
- cumulative proportion

```
# To obtain the rows of the summary: standard deviation of the components
pc.tourists$sd
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 6573.01508  824.72074  364.23534  162.54342  117.61268   64.14162   61.48464
##     Comp.8
##   34.06981
```

```
# proportion of variance explained by each PC
pc.tourists$sd^2/sum(pc.tourists$sd^2)
```

```
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
## 9.804356e-01 1.543489e-02 3.010606e-03 5.995544e-04 3.139049e-04 9.336190e-05
##       Comp.7       Comp.8
## 8.578730e-05 2.634082e-05
```

```
# cumulative proportion of explained variance
cumsum(pc.tourists$sd^2)/sum(pc.tourists$sd^2)
```

```
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 0.9804356 0.9958704 0.9988811 0.9994806 0.9997945 0.9998879 0.9999737 1.0000000
```

```
# loadings (recall: coefficients of the linear combination of the original variables that defines each principal component)

load.tour <- pc.tourists$loadings
load.tour
```

```
##
## Loadings:
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## Hotel.5stars                                    0.179  0.879  0.423
## Hotel.4stars    0.863  0.484 -0.130
## Hotel.3stars    0.454 -0.837 -0.171  0.152  0.203
## Hotel.2stars          -0.149 -0.146 -0.157 -0.792 -0.515  0.158  0.110
## Hotel.1star           -0.145        -0.249 -0.457  0.814 -0.160  0.127
## Residences      0.196 -0.103  0.954  0.126 -0.155
## Bed.Breakfast                              -0.102  0.132  0.415 -0.890
## Rented.flats                0.140 -0.931  0.279 -0.146
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
## Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

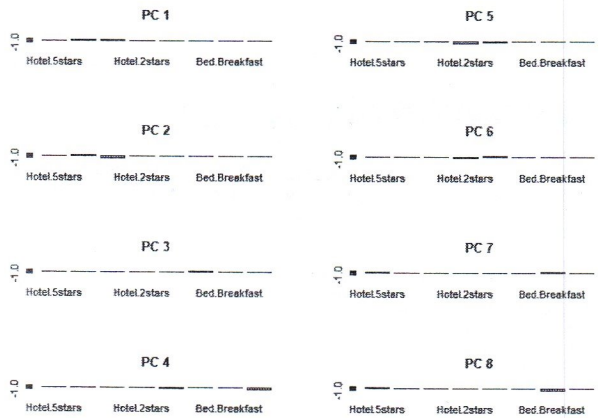
the empty are because they're so close to zero, if we want to see the full matrix

```
load.tour[,1:8]
```

```
##                  Comp.1      Comp.2      Comp.3      Comp.4      Comp.5
## Hotel.5stars   0.03698800  0.06875905  0.03032049  0.015383216  0.09843182
## Hotel.4stars   0.86250762  0.48404327 -0.12963676 -0.003314527 -0.03025384
## Hotel.3stars   0.45374825 -0.83652698 -0.17078613  0.151654002  0.20265148
## Hotel.2stars   0.05252810 -0.14868959 -0.14612870 -0.157100320 -0.79181454
## Hotel.1star    0.03921975 -0.14477781 -0.04127983 -0.249443398 -0.45701223
## Residences     0.19594019 -0.10300007  0.95393089  0.125697570 -0.15539773
## Bed.Breakfast 0.01153809 -0.01380745 -0.02285702 -0.086555157 -0.10166414
## Rented.flats   0.07747214 -0.08560395  0.13979581 -0.930891638  0.27930944
##                     Comp.6       Comp.7      Comp.8
## Hotel.5stars    0.17926970  0.878775217  0.42270057
## Hotel.4stars    0.01048520 -0.060705523 -0.01598857
## Hotel.3stars   -0.01267825  0.030573859 -0.00227378
## Hotel.2stars   -0.51485280  0.158149219  0.10974173
## Hotel.1star     0.81423507 -0.160340573  0.12659831
## Residences     -0.02558117 -0.003028534 -0.02005940
## Bed.Breakfast  0.13233387  0.415205521 -0.88961706
## Rented.flats  -0.14619468 -0.003020767  0.03423665
```

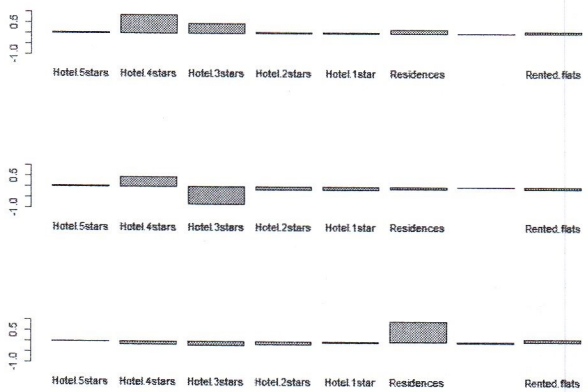
the true principal components are calculated using the full matrix

```
# graphical representation of the loadings of the first six principal components
x11()
par(mfcol = c(4,2))
for(i in 1:8) barplot(load.tour[,i], ylim = c(-1, 1), main=paste("PC",i))
```



Probably we'll take just the first PC; ⇒ let's take a closer look to the first 3 principal components:

```
x11()
par(mfrow = c(3,1))
for(i in 1:3) barplot(load.tour[,i], ylim = c(-1, 1))
```



Interpretation of the loadings:

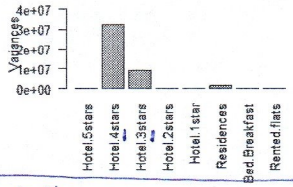
- First PCs: weighted average of the number of nights in 3,4 stars hotel and residences
- Second PCs: contrast between the number of nights in 3 and 4 stars hotel
- Third PC: residences

The loadings reflect the previous observation: the first 3 PCs are driven by the variables displaying the highest variability.

```
# Explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
plot(pc.tourists, las=2, main='Principal components', ylim=c(0,4.5e7))
barplot(sapply(tourists,sd)^2, las=2, main='Original Variables', ylim=c(0,4.5e7), ylab='Variances')
plot(cumsum(pc.tourists$sd^2)/sum(pc.tourists$sd^2), type='b', axes=F, xlab='number of components',
     ylab='contribution to the total variance', ylim=c(0,1))
abline(h=1, col='blue')
abline(h=0.8, lty=2, col='blue')
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(tourists),labels=1:ncol(tourists),las=2)
```

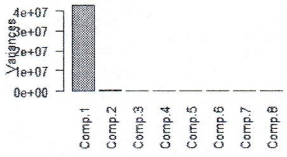
← to see how many principal components are really needed

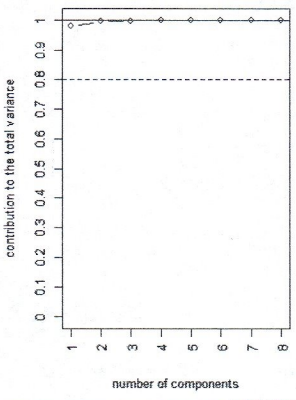
Variances of the Original Variables :



these are the plots we need to look at in order to decide how many principal components take into consideration

Variances of the Principal components :






```
# The first PC explains more than 98% of the total variability.
# This is due to the masking effect of those 3 variables over the others
```

```
###
### Principal component analysis of the dataset 'tourists', but on the standardized variables

# We compute the standardized variables
tourists.sd <- scale(tourists)
tourists.sd <- data.frame(tourists.sd)

head(tourists.sd)
```

usually done when we have different units of measure for the features

```
##   Hotel.5stars Hotel.4stars Hotel.3stars Hotel.2stars Hotel.1star Residences
## 1   0.20868970   0.01397475   0.09114066  -0.29293603   0.5201322  0.1247660
## 2   0.29424616   0.16848570   0.07093516  -0.29033860   0.1037466  0.2203068
## 3   0.27480151   0.28524681   0.29971358   0.15382199   0.4227517  0.4681159
## 4  -0.07909111  -0.06012027  -0.01575296  -0.20981826  -0.0641508  0.3964602
## 5   0.38758048   0.04066301  -0.11971352  -0.17605167   0.2414225  0.5435035
## 6   0.22424542  -0.03817269  -0.15751736  -0.02020584  -0.1615313  0.7099536
##   Bed.Breakfast Rented.flats
## 1   -0.43655633 -0.011801778
## 2   -0.54858961 -0.263408587
## 3   -0.42535300 -0.069153330
## 4   -0.42535300 -0.028452228
## 5    0.05639008  0.006698723
## 6   -0.17887980  0.054800025
```

```
pc.tourists <- princomp(tourists.sd, scores=T)
pc.tourists
```

```
## Call:
## princomp(x = tourists.sd, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 2.6718997 0.6343339 0.4643065 0.3037082 0.2224482 0.1910011 0.1430582 0.1048307
##
##  8  variables and  240 observations.
```

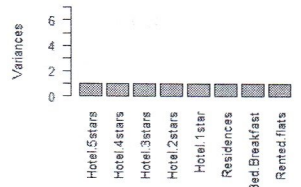
```
summary(pc.tourists)
```

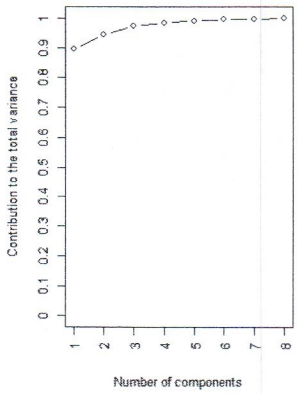
```
## Importance of components:
##                           Comp.1     Comp.2     Comp.3     Comp.4      Comp.5
## Standard deviation     2.6718997 0.63433394 0.46430646 0.30370821 0.222448208
## Proportion of Variance 0.8961148 0.05050789 0.02706031 0.01157808 0.006211281
## Cumulative Proportion  0.8961148 0.94662273 0.97368304 0.98526112 0.991472400
##                            Comp.6      Comp.7      Comp.8
## Standard deviation     0.191001055 0.143058239 0.104830741
## Proportion of Variance 0.004579256 0.002568911 0.001379433
## Cumulative Proportion  0.996051656 0.998620567 1.000000000
```

```
# Explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
plot(pc.tourists, las=2, main='Principal Components', ylim=c(0,7))
abline(h=1, col='blue')
barplot(sapply(tourists.sd,sd)^2, las=2, main='Original Variables', ylim=c(0,7), ylab='Variances')
plot(cumsum(pc.tourists$sde^2)/sum(pc.tourists$sde^2), type='b', axes=F, xlab='Number of components', ylab='Contribution to
 the total variance', ylim=c(0,1))
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(tourists.sd),labels=1:ncol(tourists.sd),las=2)
```

Variance of the Original Variables :

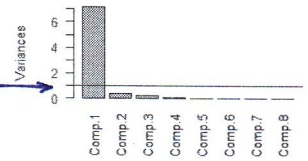
by construction they're scaled →





Principal Components

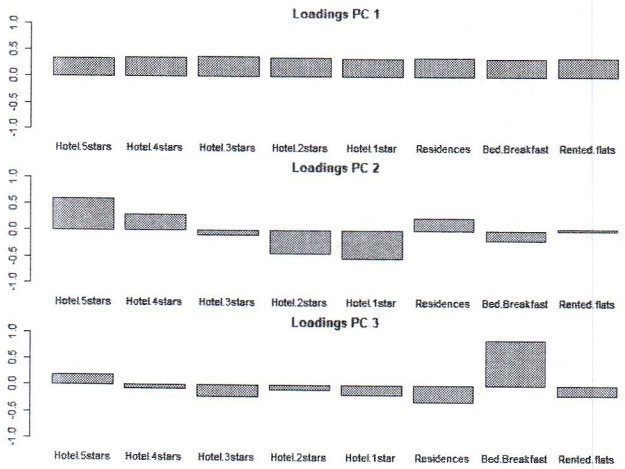
mean value of the variances →




```
# If we wanted to perform dimensionality reduction, we could keep 1 or 2 PCs

# loadings
load.tour <- pc.tourists$loadings
load.tour
```

```
##
## Loadings:
##               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## Hotel.5stars   0.341  0.591  0.193  0.269  0.147  0.402  0.324  0.369
## Hotel.4stars   0.363  0.305         0.271  0.163        -0.345 -0.743
## Hotel.3stars   0.368        -0.207  0.159        -0.105 -0.699  0.533
## Hotel.2stars   0.352 -0.428         0.502  0.227 -0.437  0.437
## Hotel.1star    0.347 -0.523 -0.173         -0.152  0.724        -0.134
## Residences     0.358  0.254 -0.308 -0.237 -0.708 -0.276  0.274
## Bed.Breakfast  0.338 -0.171  0.869 -0.194 -0.185 -0.135
## Rented.flats   0.360        -0.173 -0.691  0.579 -0.135
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
## Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

```
x11()
par(mar = c(2,2,2,1), mfrow=c(3,1))
for(i in 1:3)barplot(load.tour[,i], ylim = c(-1, 1), main=paste('Loadings PC ',i,sep=''))
```



PC1: average of all the original variables ←

PC2: contrast between the expensive solution again the cheap solution ←

! Interpretation of the loadings: In this case, the first PC represents an average of the number of nights spent in all the types of hotels and residences, taken with very similar weights. The sencond PC contrasts the more expensive solutions (4,5 stars hotels and residences) against the cheap solutions (1,2 stars hotels and B&B).

- High PC1: general high flow of tourists
- Low PC1: general low flow of tourists
- High PC2: high flow for expensive solutions, low flow for cheap solutions
- Low PC2: low flow for expensive solutions, high flow for cheap solutions

```
# scores
scores.tourists <- pc.tourists$scores
head(scores.tourists)
```

coordinates of the original data in the new reference system obtained with the rotation that lead us to the PC's space ←

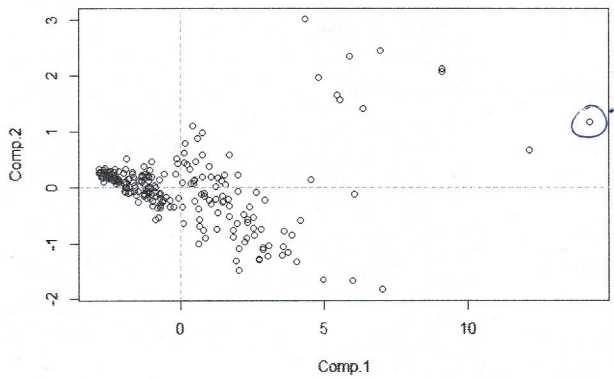
```
##        Comp.1    Comp.2     Comp.3     Comp.4     Comp.5      Comp.6
## 1  0.07995640 0.0803750 -0.4623249 -0.04905726 -0.1338738  0.60459972
## 2 -0.08002789 0.4295531 -0.4629376  0.21841150 -0.2233720  0.36028568
## 3  0.50727996 0.1290731 -0.6140708  0.26341977 -0.2587780  0.25312341
## 4 -0.16271958 0.2317603 -0.4675140 -0.13246620 -0.2764895 -0.03288108
## 5  0.34064659 0.3283221 -0.0504921 -0.15477189 -0.3946383  0.26087543
## 6  0.15495852 0.4388325 -0.2756186 -0.14435742 -0.3784939 -0.18082750
##        Comp.7       Comp.8
## 1 -0.01125450  0.045780105
## 2 -0.03555737 -0.005413831
## 3  0.04699448 -0.014769479
## 4  0.05775706  0.001361058
## 5  0.28166951 -0.025116051
## 6  0.39051071  0.015650718
```

```
summary(scores.tourists)
```

```
##      Comp.1             Comp.2             Comp.3             Comp.4
##  Min.   :-2.835   Min.   :-1.83090   Min.   :-1.68826   Min.   :-1.234657
##  1st Qu.:-1.927   1st Qu.:-0.23460   1st Qu.:-0.19552   1st Qu.:-0.122926
##  Median :-0.859   Median : 0.08915   Median : 0.03929   Median : 0.004307
##  Mean   : 0.000   Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.000000
##  3rd Qu.: 1.345   3rd Qu.: 0.23031   3rd Qu.: 0.18204   3rd Qu.: 0.179998
##  Max.   :14.217   Max.   : 3.00347   Max.   : 2.12223   Max.   : 0.860293
##      Comp.5             Comp.6             Comp.7             Comp.8
##  Min.   :-0.79962   Min.   :-0.593645   Min.   :-0.407385   Min.   :-0.367028
##  1st Qu.:-0.11141   1st Qu.:-0.096587   1st Qu.:-0.080357   1st Qu.:-0.043799
##  Median : 0.01562   Median :-0.007289   Median : 0.003714   Median :-0.002511
##  Mean   : 0.00000   Mean   : 0.000000   Mean   : 0.000000   Mean   : 0.000000
##  3rd Qu.: 0.11411   3rd Qu.: 0.096923   3rd Qu.: 0.076118   3rd Qu.: 0.049184
##  Max.   : 0.65384   Max.   : 0.734674   Max.   : 0.465267   Max.   : 0.370298
```

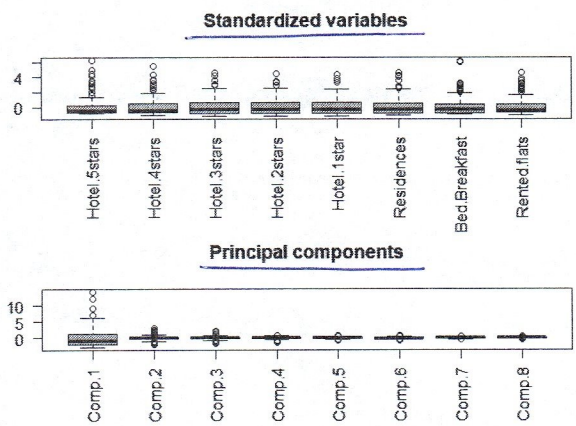
```
x11()
plot(scores.tourists[,1:2])
abline(h=0, v=0, lty=2, col='grey')
```

Graphical representation of the scores: (we're already making the dimension reduction (we go to 2 dimensions))



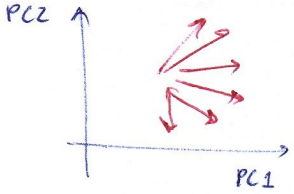
each dot represent one of the original datum projected in this 2-dimensional space of the 2 principal components (2 firsts principal components)

```
x11()
layout(matrix(c(1,2),2))
boxplot(tourists.sd, las=2, col='gold', main='Standardized variables')
scores.tourists <- data.frame(scores.tourists)
boxplot(scores.tourists, las=2, col='gold', main='Principal components')
```

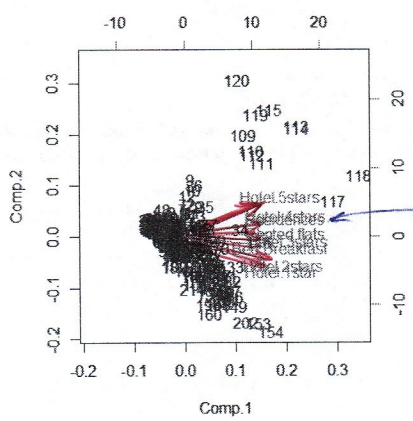



```
x11()
biplot(pc.tourists)
```

the arrows are just the projection of the old reference system on the new one



the coordinates of the arrows of PC1 are all positive (since PC1 was an average of all the variables), the coordinates of PC2 underline the contrast between expensive/cheap solutions



(same scatterplot as before (↑) but with numbers instead of dots (number = number of the line))

the arrows are basically the loadings: the first arrow (Hotel 5 stars) has as coordinates the loadings of the first and second principal component corresponding to the original variable "Hotel 5 stars"

```
# Let's use the categorical variables to further interpret the results
head(tourists.label)
```

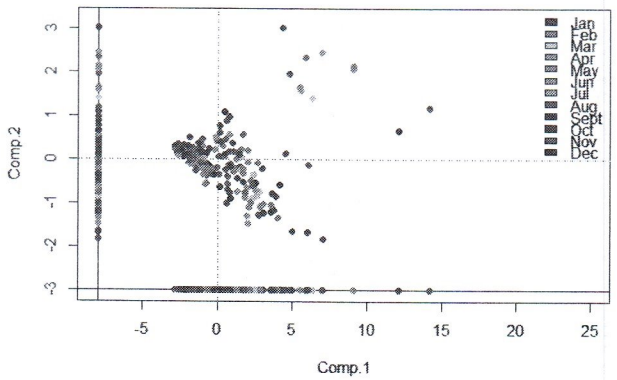
```
##   Month Region.of.origin
## 1   Jan         PIEMONTE
## 2   Feb         PIEMONTE
## 3   Mar         PIEMONTE
## 4   Apr         PIEMONTE
## 5   May         PIEMONTE
## 6   Jun         PIEMONTE
```

```
# Color according to Month
head(tourists.label[,1])
```

```
## [1] "Jan" "Feb" "Mar" "Apr" "May" "Jun"
```

```
# We order the labels according to time order
tourists.label[,1] <- factor(tourists.label[,1], levels=c("Jan","Feb","Mar","Apr","May","Jun","Jul","Aug","Sept","Oct","Nov"
,"Dec"))
col.ramp <- rainbow(12)
col.lab1 <- rep(NA, n)
for(i in 1:n)
  col.lab1[i] = col.ramp[which(tourists.label[i,1] == levels(tourists.label[,1]))]
```

```
x11()
plot(scores.tourists[,1:2], col=col.lab1, pch=19, xlim=c(-8,25), ylim=c(-3,3.2))
abline(h=-3, v=-8, col=1)
points(scores.tourists[,1], rep(-3, n), col=col.lab1, pch=19)
points(rep(-8, n),scores.tourists[,2], col=col.lab1, pch=19)
abline(h=0, v=0, lty=2, col='grey')
legend('topright',levels(tourists.label[,1]),fill=rainbow(12),bty='n')
```



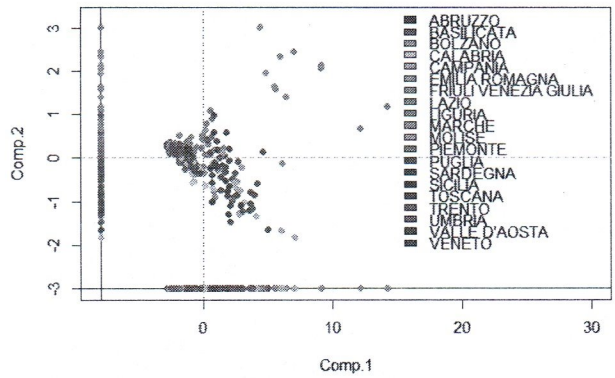

```
# Months of Expo 2015: May to Oct
expo.label=factor(ifelse(tourists.label[,1] %in% c("May","Jun","Jul","Aug","Sept","Oct"), 'Expo', 'Non Expo'))
col.expo = ifelse(tourists.label[,1] %in% c("May","Jun","Jul","Aug","Sept","Oct"), 'red', 'blue')
```

```
x11(width=10, height = 10)
layout(cbind(c(2,4), c(1,3)), widths = c(1,4), heights = c(4, 1))
par(mar=rep(3,4))
plot(scores.tourists[,1:2], col=col.expo, pch=19, xlim=c(-8,20), ylim=c(-3,3.2), las=2)
abline(h=-3, v=-8, col=1)
points(scores.tourists[,1], rep(-3, n), col=col.expo, pch=19)
points(rep(-8, n),scores.tourists[,2], col=col.expo, pch=19)
abline(h=0, v=0, lty=2, col='grey')
boxplot(scores.tourists[,2]~expo.label, col=c('red', 'blue'), ylim=c(-3,3.2), las=2)
boxplot(scores.tourists[,1]~expo.label, col=c('red', 'blue'), ylim=c(-8,20), horizontal=T, las=2)

# Color according to Region of Origin
head(tourists.label[,2])
```

cool : [

```
## [1] "PIEMONTE" "PIEMONTE" "PIEMONTE" "PIEMONTE" "PIEMONTE" "PIEMONTE"
```

```
col.ramp <- rainbow(20)
col.lab2 <- rep(NA, n)
for(i in 1:n)
  col.lab2[i] = col.ramp[which(tourists.label[i,2] == levels(as.factor(tourists.label[,2])))]

x11()
plot(scores.tourists[,1:2], col=col.lab2, pch=19, xlim=c(-8,30), ylim=c(-3,3.2))
abline(h=-3, v=-8, col=1)
points(scores.tourists[,1], rep(-3, n), col=col.lab2, pch=19)
points(rep(-8, n),scores.tourists[,2], col=col.lab2, pch=19)
abline(h=0, v=0, lty=2, col='grey')
legend('topright',levels(as.factor(tourists.label[,2])),fill=rainbow(20),bty='n')
```




```
graphics.off()
```

```
### ------------------------------------------------------------------------
### Homework: try to perform the PCA on the log-transformed data
tourists.mod = tourists
for(i in 1:8) tourists.mod[which(tourists[,i]==0),i] = 1
tourists.log = log(tourists.mod)
head(tourists.log)
```

```
##   Hotel.5stars Hotel.4stars Hotel.3stars Hotel.2stars Hotel.1star Residences
## 1     5.541264     8.653994     8.263075     5.860786    6.222576   7.305860
## 2     5.624018     8.796793     8.246958     5.863631    5.940171   7.388328
## 3     5.605802     8.892611     8.415825     6.259581    6.163315   7.575072
## 4     5.198497     8.577535     8.174703     5.948035    5.799093   7.524561
## 5     5.707110     8.680162     8.080547     5.981414    6.042633   7.625595
## 6     5.556828     8.600799     8.043984     6.122493    5.707110   7.728856
##   Bed.Breakfast Rented.flats
## 1      3.713572     6.311735
## 2      3.433987     6.028279
## 3      3.737670     6.253829
## 4      3.737670     6.295266
## 5      4.442651     6.329721
## 6      4.158883     6.375025
```

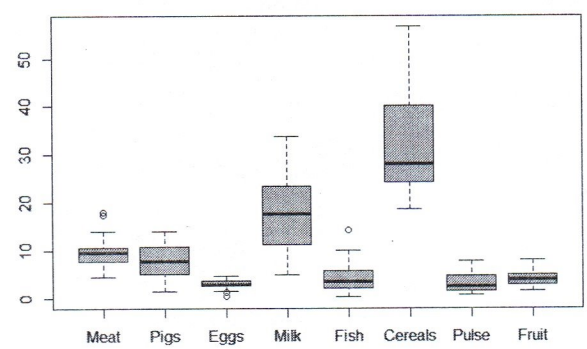
```
### ------------------------------------------------------------------------
### Principal component analysis of the dataset 'food'

# upload the data
food <- read.table('Food.txt', header=T)
head(food)
```

```
##                Meat Pigs Eggs Milk Fish Cereals Pulse Fruit
## Albania        10.1  1.4  0.5  8.9  0.2    42.3   5.5   1.7
## Austria         8.9 14.0  4.3 19.9  2.1    28.0   1.3   4.3
## Belg.Lux.      13.5  9.3  4.1 17.5  4.5    26.6   2.1   4.0
## Bulgaria        7.8  6.0  1.6  8.3  1.2    56.7   3.7   4.2
## Czechoslovakia  9.7 11.4  2.8 12.5  2.0    34.3   1.1   4.0
## Denmark        10.6 10.8  3.7 25.0  9.9    21.9   0.7   2.4
```

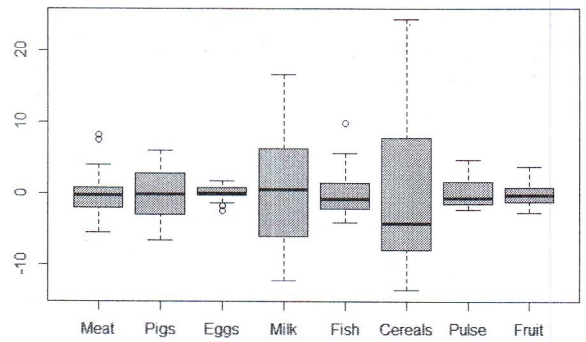
```
dim(food)
```

```
## [1] 25  8
```

```
n <- dim(food)[1]
p <- dim(food)[2]

# exploration
x11()
boxplot(food,col='gold')
```




```
x11()
boxplot(scale(x=food, center = T, scale=F), col='gold')
```




```
S <- cov(food)
round(S,digits = 2)
```

```
##           Meat   Pigs  Eggs   Milk   Fish Cereals Pulse Fruit
## Meat     11.20   1.50  2.13  11.96   0.69  -18.36 -2.32 -0.45
## Pigs      1.50  13.01  2.33   6.05  -1.87  -17.34 -4.48  0.00
## Eggs      2.13   2.33  1.19   4.37   0.41   -8.83 -1.22 -0.03
## Milk     11.96   6.05  4.37  50.54   3.36  -46.26 -8.77 -5.24
## Fish      0.69  -1.87  0.41   3.36  11.58  -19.58 -0.99  1.63
## Cereals -18.36 -17.34 -8.83 -46.26 -19.58  120.45 14.19  0.92
## Pulse    -2.32  -4.48 -1.22  -8.77  -0.99   14.19  3.94  1.34
## Fruit    -0.45   0.00 -0.03  -5.24   1.63    0.92  1.34  3.25
```

```
R <- cor(food)
round(R,digits = 2)
```

```
##          Meat  Pigs  Eggs  Milk  Fish Cereals Pulse Fruit
## Meat     1.00  0.12  0.58  0.50  0.06   -0.50 -0.35 -0.07
## Pigs     0.12  1.00  0.59  0.24 -0.15   -0.44 -0.63  0.00
## Eggs     0.58  0.59  1.00  0.56  0.11   -0.74 -0.56 -0.01
## Milk     0.50  0.24  0.56  1.00  0.14   -0.59 -0.62 -0.41
## Fish     0.06 -0.15  0.11  0.14  1.00   -0.52 -0.15  0.27
## Cereals -0.50 -0.44 -0.74 -0.59 -0.52    1.00  0.65  0.05
## Pulse   -0.35 -0.63 -0.56 -0.62 -0.15    0.65  1.00  0.37
## Fruit   -0.07  0.00 -0.01 -0.41  0.27    0.05  0.37  1.00
```

```
var.gen <- det(S)
var.gen
```

```
## [1] 1073887
```

```
var.tot <- sum( diag(S) )
var.tot
```

```
## [1] 215.1638
```

```
### ------------------------------------------------------------------------
### Principal component analysis of the dataset 'food', based on the correlation matrix
food.sd <- scale(food)
food.sd <- data.frame(food.sd)

head(food.sd)
```

```
##                      Meat       Pigs       Eggs        Milk        Fish
## Albania         0.08126490 -1.8299828 -2.2437259 -1.15570645 -1.20028213
## Austria        -0.27725673  1.6636208  1.2335002  0.39161231 -0.64187467
## Belg.Lux.       1.09707621  0.3604512  1.0504883  0.05401549  0.06348211
## Bulgaria       -0.60590157 -0.5545403 -1.2371605 -1.24010566 -0.90638347
## Czechoslovakia -0.03824231  0.9427184 -0.1390890 -0.64931122 -0.67126454
## Denmark         0.23064892  0.7763564  0.6844645  1.10900556  1.65053488
##                     Cereals      Pulse       Fruit
## Albania         0.9159176  1.2227536 -1.35040507
## Austria        -0.3870690 -0.8923886  0.09091397
## Belg.Lux.      -0.5146342 -0.4895043 -0.07539207
## Bulgaria        2.2280161  0.3162641  0.03547862
## Czechoslovakia  0.1869740 -0.9931096 -0.07539207
## Denmark        -0.9428885 -1.1945517 -0.96235764
```

```
sapply(food.sd,mean)
```

```
##          Meat          Pigs          Eggs          Milk          Fish
##  1.796913e-16  5.724587e-19  4.775694e-17  5.744537e-17  6.304419e-17
##       Cereals         Pulse         Fruit
##  2.642418e-16 -1.112825e-17 -6.606911e-17
```

```
sapply(food.sd,sd)
```

```
##    Meat    Pigs    Eggs    Milk    Fish Cereals   Pulse   Fruit
##       1       1       1       1       1       1       1       1
```

```
cov(food.sd)
```

```
##               Meat         Pigs        Eggs        Milk        Fish     Cereals
## Meat     1.00000000  0.124155626  0.58236284  0.5025975  0.06095745 -0.49987746
## Pigs     0.12415563  1.000000000  0.59222166  0.2360289 -0.15214735 -0.43819850
## Eggs     0.58236284  0.592221664  1.00000000  0.5619534  0.11150484 -0.73589172
## Milk     0.50259751  0.236028903  0.56195338  1.0000000  0.13874572 -0.59292569
## Fish     0.06095745 -0.152147350  0.11150484  0.1387457  1.00000000 -0.52423080
## Cereals -0.49987746 -0.438198497 -0.73589172 -0.5929257 -0.52423080  1.00000000
## Pulse   -0.34944855 -0.625738013 -0.55997614 -0.6212026 -0.14715294  0.65099727
## Fruit   -0.07422123 -0.000128088 -0.01472764 -0.4086207  0.26613865  0.04654808
##              Pulse        Fruit
## Meat    -0.3494486 -0.074221231
## Pigs    -0.6257380 -0.000128088
## Eggs    -0.5599761 -0.014727635
## Milk    -0.6212026 -0.408620702
## Fish    -0.1471529  0.266138652
## Cereals  0.6509973  0.046548078
## Pulse    1.0000000  0.374969710
## Fruit    0.3749697  1.000000000
```

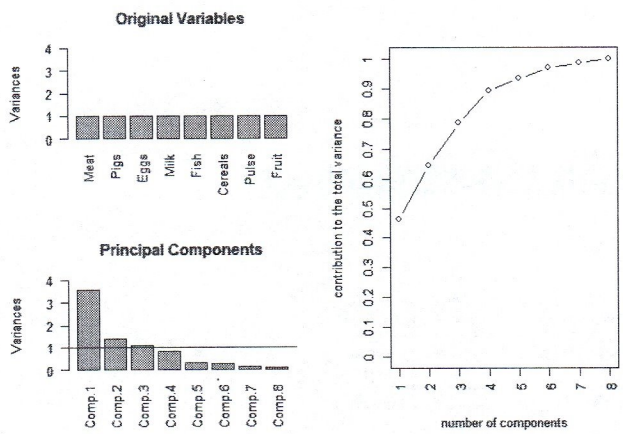
```
# PC on correlation matrix
pc.food <- princomp(food.sd, scores=T)
pc.food
```

```
## Call:
## princomp(x = food.sd, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7    Comp.8
## 1.8861623 1.1828748 1.0381046 0.9126749 0.5616426 0.5182049 0.3489782 0.3267995
##
##  8  variables and  25 observations.
```

```
summary(pc.food)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
## Standard deviation     1.8861623 1.1828748 1.0381046 0.9126749 0.56164261
## Proportion of Variance 0.4632302 0.1821866 0.1403205 0.1084604 0.04107323
## Cumulative Proportion  0.4632302 0.6454168 0.7857372 0.8941976 0.93527083
##                           Comp.6     Comp.7     Comp.8
## Standard deviation     0.51820489 0.34897823 0.32679953
## Proportion of Variance 0.03496567 0.01585753 0.01390598
## Cumulative Proportion  0.97023649 0.98609402 1.00000000
```

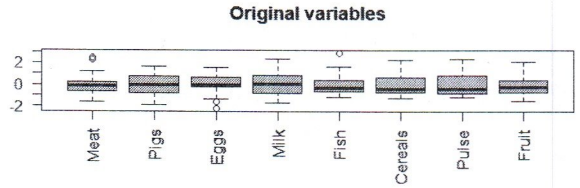
```
# explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
barplot(pc.food$sdev^2, las=2, main='Principal Components', ylim=c(0,4), ylab='Variances')
abline(h=1, col='blue')
barplot(sapply(food.sd,sd)^2, las=2, main='Original Variables', ylim=c(0,4), ylab='Variances')
plot(cumsum(pc.food$sdev^2)/sum(pc.food$sde^2), type='b', axes=F, xlab='number of components',
     ylab='contribution to the total variance', ylim=c(0,1))
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(food.sd),labels=1:ncol(food.sd),las=2)
```

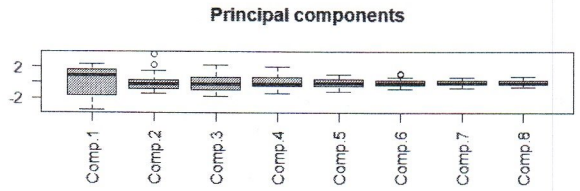



```
# scores
scores.food <- pc.food$scores
head(scores.food)
```

```
##                   Comp.1     Comp.2     Comp.3     Comp.4     Comp.5
## Albania       -2.9433087 -1.3978458  1.5957916  0.5186505  1.0231236
## Austria        1.6088967 -0.6530390 -1.6017337 -0.2957612 -0.4101784
## Belg.Lux.      1.4305750  0.1301247 -0.1845748  0.6988219  0.4609494
## Bulgaria      -2.6987861 -1.0152768 -0.3210675  0.1000601  0.5042264
## Czechoslovakia  0.2291677 -0.7790596 -1.0926580 -0.3573908  0.7954215
## Denmark         2.3581438  0.3359382  0.7215596 -1.2504550  0.1074689
##                    Comp.6      Comp.7      Comp.8
## Albania       -0.21591341  0.72564517 -0.52072792
## Austria       -0.10508561 -0.19401720  0.02950793
## Belg.Lux.     -0.27201570 -0.27537046  0.07699587
## Bulgaria       0.63305623 -0.69101212  0.26475668
## Czechoslovakia  0.37907340 -0.14838924 -0.17071810
## Denmark       -0.07236453 -0.08420645  0.73405976
```

```
# variability of the original variables / scores
x11()
layout(matrix(c(1,2),2))
boxplot(food.sd, las=2, col='gold', main='Original variables')
scores.food <- data.frame(scores.food)
boxplot(scores.food, las=2, col='gold', main='Principal components')
```

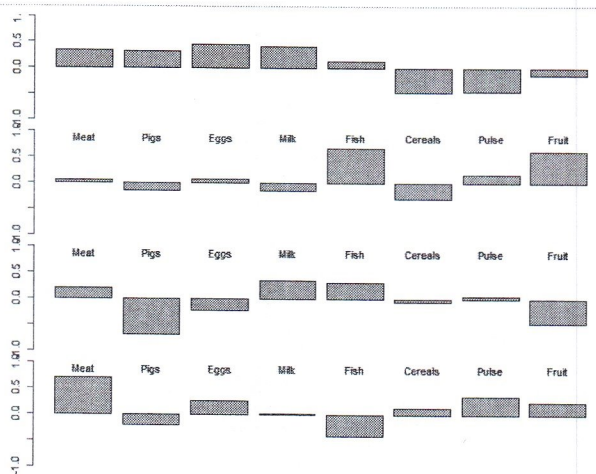




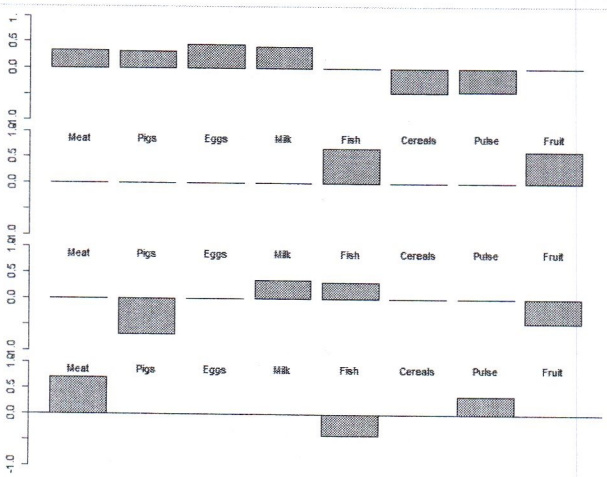

```
# loadings
load.food    <- pc.food$loadings
load.food
```

```
##
## Loadings:
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## Meat     0.332         0.204  0.717  0.485  0.197  0.123  0.206
## Pigs     0.315 -0.153 -0.679 -0.202                0.372  0.479
## Eggs     0.444        -0.229  0.260 -0.268 -0.530 -0.566
## Milk     0.407 -0.151  0.359        -0.703  0.385  0.154  0.127
## Fish     0.128  0.666  0.324 -0.404  0.130        -0.114  0.492
## Cereals -0.454 -0.286         0.129         0.322 -0.524  0.558
## Pulse   -0.435  0.174         0.364 -0.345 -0.464  0.460  0.317
## Fruit   -0.128  0.625 -0.454  0.254 -0.235  0.458        -0.234
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7 Comp.8
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.125  0.125  0.125  0.125  0.125  0.125  0.125  0.125
## Cumulative Var  0.125  0.250  0.375  0.500  0.625  0.750  0.875  1.000
```

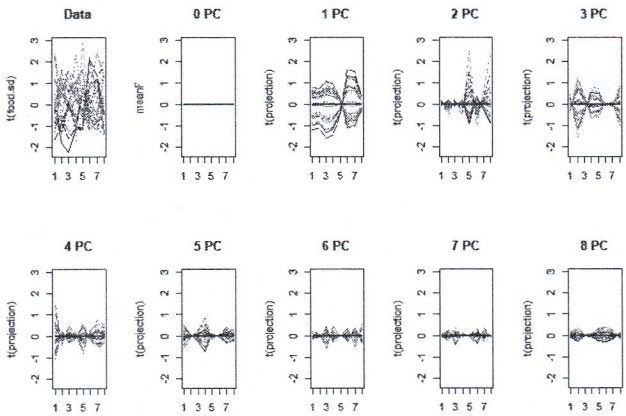
```
x11()
par(mar = c(1,4,0,2), mfrow = c(4,1))
for(i in 1:4)barplot(load.food[,i], ylim = c(-1, 1))
```



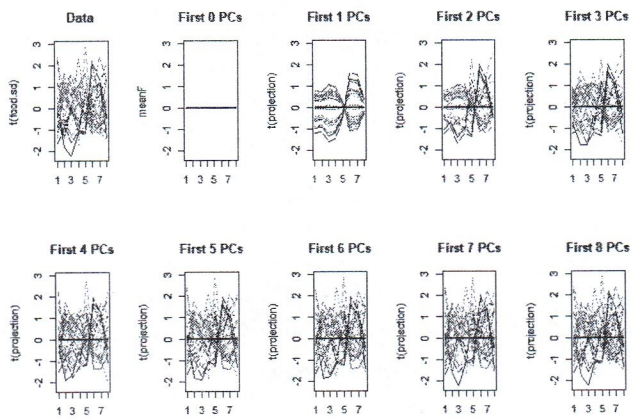

```
# let's plot only the most significant loadings
x11()
par(mar = c(1,4,0,2), mfrow = c(4,1))
for(i in 1:4) barplot(ifelse(abs(load.food[,i]) < 0.3, 0, load.food[,i]) , ylim = c(-1, 1));abline(h=0)
```



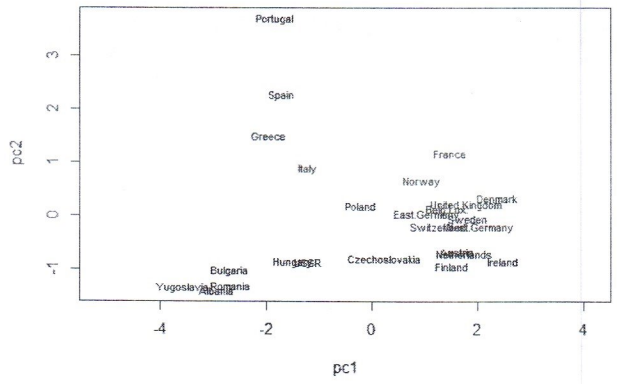
```r
# Projection on the space generated by the k-th principal component
x11(width=21, height=7)
par(mfrow=c(2,5))
matplot(t(food.sd), type='l', main = 'Data', ylim=range(food.sd))

meanF <- colMeans(food.sd)
matplot(meanF, type='l', main = '0 PC', lwd=2, ylim=range(food.sd))
for(i in 1:8)
{
  projection <- matrix(meanF, dim(food.sd)[[1]], dim(food.sd)[[2]], byrow=T) + scores.food[,i] %*% t(load.food[,i])
  matplot(t(projection), type='l', main = paste(i, 'PC'), ylim=range(food.sd))
  matplot(meanF, type='l', lwd=2, add=T)
}
```



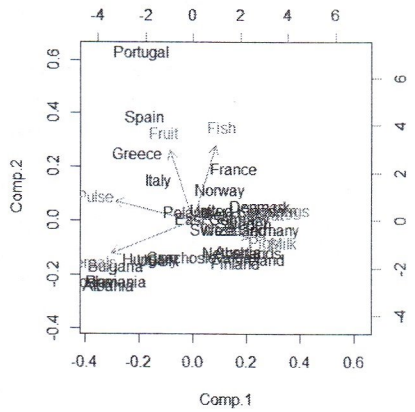

```r
# Projection on the space generated by the first k principal components
x11(width=21, height=7)
par(mfrow=c(2,5))
matplot(t(food.sd), type='l', main = 'Data', ylim=range(food.sd))
meanF <- colMeans(food.sd)
matplot(meanF, type='l', main = 'First 0 PCs', lwd=2, ylim=range(food.sd))
projection <- matrix(meanF, dim(food.sd)[[1]], dim(food.sd)[[2]], byrow=T)
for(i in 1:8){
  projection <- projection + scores.food[,i] %*% t(load.food[,i])
  matplot(t(projection), type='l', main = paste('First', i, 'PCs'), ylim=range(food.sd))
  matplot(meanF, type='l', lwd=2, add=T)
}
```



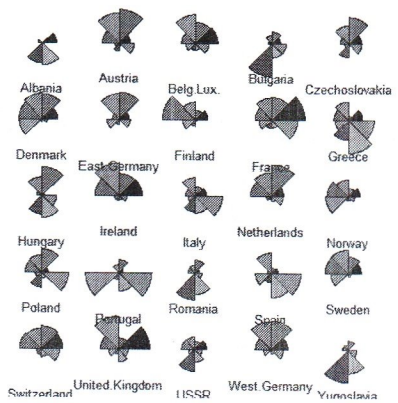

```r
# Scores
x11()
par(mfrow=c(1,1))
plot(scores.food[,1],scores.food[,2],type="n",xlab="pc1",ylab="pc2", asp=1, xlim=c(-4,3))
text(scores.food[,1],scores.food[,2],dimnames(food)[[1]], cex=0.7)
```




```
x11()
biplot(pc.food)
```




```
x11()
stars(food.sd, draw.segments=T)
```




```
graphics.off()
```

```
###
### p-dimensional geometrical interpretation of the principal components
library(rgl)

# theoretical mean and covariance matrix of the model
mu  <- c(0, 2, 3)
mu
```

```
## [1] 0 2 3
```

```
sig <- rbind(c(9, 1, 1), c(1, 4, 1), c(1, 1, 1))
sig
```

```
##      [,1] [,2] [,3]
## [1,]    9    1    1
## [2,]    1    4    1
## [3,]    1    1    1
```

```
nobs <- 100

X <- rmvnorm(nobs, mu, sig)

# sample mean and covariance matrix
M <- colMeans(X)
M
```

```
## [1] 0.3882809 2.2340284 3.1678238
```

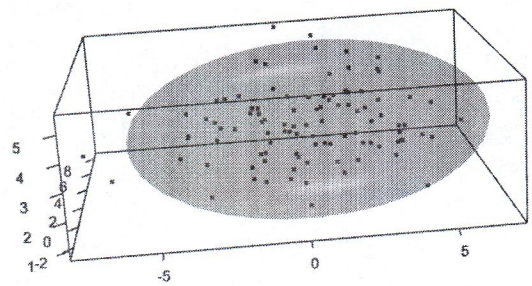
```
S <- cov(X)
S
```

```
##            [,1]       [,2]      [,3]
## [1,]  7.2295884 -0.6320721 0.4135501
## [2,] -0.6320721  4.4114988 0.8989690
## [3,]  0.4135501  0.8989690 0.9575370
```

```
open3d()                    # open a new device
```

```
points3d(X, asp=1, size=4)  # plot the points
axes3d()                    # add the axes
plot3d(ellipse3d(S, centre=M, level= 9/10), alpha=0.15, add = TRUE) # add the ellipsoid
a = scene3d()
rgl.close()
x11()
rglwidget(a)
```

We consider a 3D dataset. The gray ellipse is the graphical representation of the sample variance/covariance matrix



We can interprete the principal components as the directions that best approximate the data

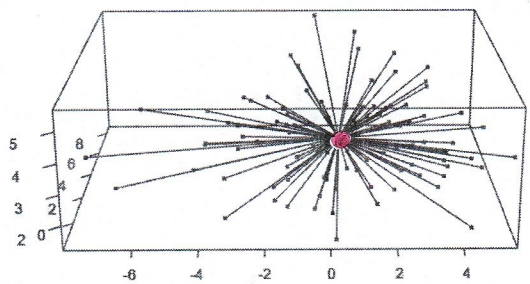
```
# principal components
PC <- princomp(X)
summary(PC)
```

```
## Importance of components:
##                           Comp.1    Comp.2     Comp.3
## Standard deviation     2.7016268 2.1190227 0.82679716
## Proportion of Variance 0.5851839 0.3600086 0.05480746
## Cumulative Proportion  0.5851839 0.9451925 1.00000000
```

```
# "0" principal component: the best approximation of dimension 0 (a point)
open3d()
```

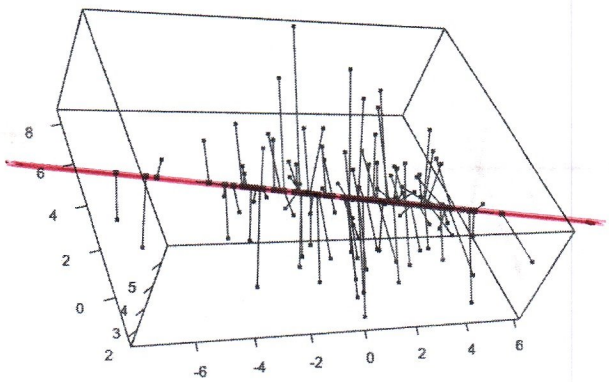
```
points3d(X, asp=1, size=4)
axes3d()
points3d(t(M), col='red', size=6)

for(i in 1:100)
  lines3d(rbind(X[i,], M))

a = scene3d()
rgl.close()
x11()
rglwidget(a)
```

linear space of dimension 0 that approximates the data (a point)
⇒ sample mean



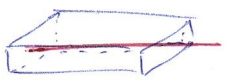
```
# I principal component: the best approximation of dimension 1 (a line)
open3d()
```

```
points3d(X, asp=1, size=4)
axes3d()

PC1 <- NULL
for(i in 1:nobs) PC1 <- rbind(PC1, PC$loadings[,1]*PC$scores[i,1] + M)
points3d(PC1, col='red', size=6)

for(i in 1:nobs)lines3d(rbind(X[i,], PC1[i,]),col='blue')

lines3d(rbind(M + 2*PC$sdev[1] * PC$loadings[,1], M - 2*PC$sdev[1] * PC$loadings[,1]), col='forestgreen',lwd=2)

a = scene3d()
rgl.close()
x11()
rglwidget(a)
```



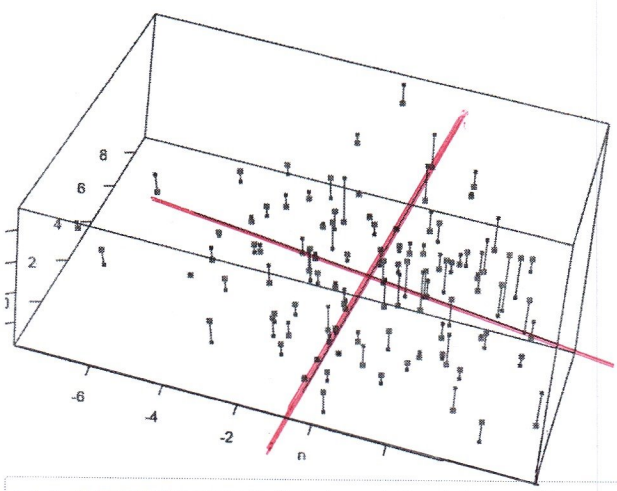
linear space of dimension **1** that approximate best the data

direction that minimizes the sum of all the black segments



```
# I and II principal components: the best approximation of dimension 2 (a plane)
open3d()
```

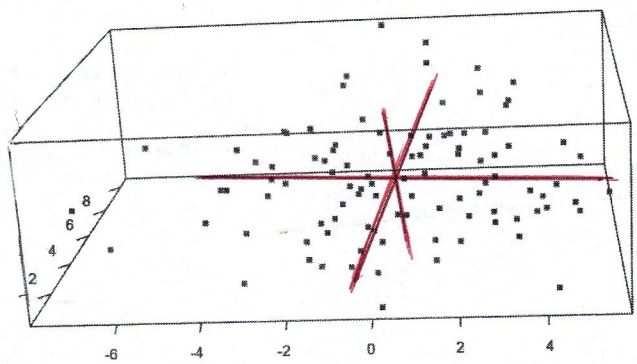
```
points3d(X, asp=1, size=4)
axes3d()

PC12 <- NULL
for(i in 1:nobs)PC12 <- rbind(PC12, PC$loadings[,1]*PC$scores[i,1] + PC$loadings[,2]*PC$scores[i,2] + M)
points3d(PC12, col='red', size=6)

for(i in 1:nobs)lines3d(rbind(X[i,], PC12[i,]),col='blue')

lines3d(rbind(M + 2*PC$sdev[1] * PC$loadings[,1], M - 2*PC$sdev[1] * PC$loadings[,1]), col='forestgreen',lwd=2)
lines3d(rbind(M + 2*PC$sdev[2] * PC$loadings[,2], M - 2*PC$sdev[2] * PC$loadings[,2]), col='forestgreen',lwd=2)

a = scene3d()
rgl.close()
x11()
rglwidget(a)
```



linear space of dimension **2** that approximate best the data

```
# I, II and III principal components: the best approximation of dimension 3 (the entire space)
open3d()
```

```
points3d(X, asp=1, size=4)
axes3d()

PC123 <- NULL
for(i in 1:nobs)PC123 <- rbind(PC123, PC$loadings[,1]*PC$scores[i,1] + PC$loadings[,2]*PC$scores[i,2] + PC$loadings[,3]*PC$s
cores[i,3] + M)
points3d(PC123, col='red', size=6)

for(i in 1:nobs)lines3d(rbind(X[i,], PC123[i,]),col='blue')

lines3d(rbind(M + 2*PC$sdev[1] * PC$loadings[,1], M - 2*PC$sdev[1] * PC$loadings[,1]), col='forestgreen',lwd=2)
lines3d(rbind(M + 2*PC$sdev[2] * PC$loadings[,2], M - 2*PC$sdev[2] * PC$loadings[,2]), col='forestgreen',lwd=2)
lines3d(rbind(M + 2*PC$sdev[3] * PC$loadings[,3], M - 2*PC$sdev[3] * PC$loadings[,3]), col='forestgreen',lwd=2)

a = scene3d()
rgl.close()
x11()
rglwidget(a)
```



```
###------------------------------------------------------------------------
### Example:
### Question (c) of Problem 3 of the 29/06/2010 exam
## The file scotland.txt collects the number of residents in Scotland, according to the last census of 2001, divided by age
 and county. Assume the data associated with different counties to be independent and identically distributed, and assume th
e data corresponding to different age ranges to be dependent. Perform a dimensionality reduction of the dataset through a pr
incipal component analysis and interpret the obtained components.

age <- read.table('scotland.txt', header=T)
head(age)
```

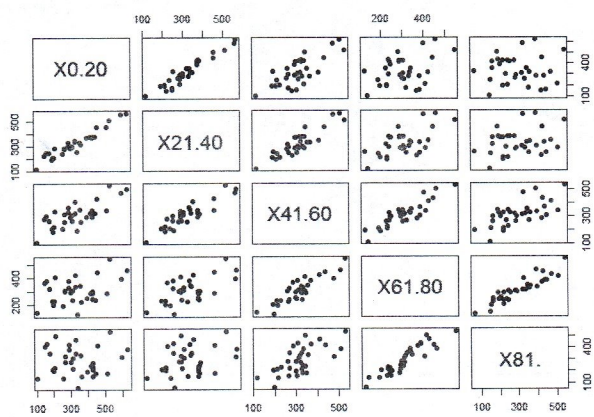
```
##                    X0.20 X21.40 X41.60 X61.80 X81.
## Caithness            193    192    209    227  217
## Sutherland           628    561    507    456  389
## Ross and Cromarty    424    369    301    243  161
## Inverness-shire      307    288    268    236  231
## Nairnshire           176    248    268    319  354
## County of Moray      334    252    195    125   64
```

residents are divided by age and location (we don't scale data:
- the unit of measure is always the same
- there is no big difference in variability)

```
dim(age)
```

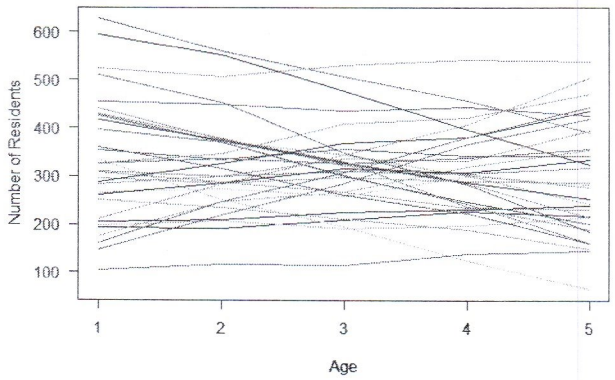
```
## [1] 33  5
```

```
x11()
pairs(age, pch=19)
```



← variables are visibly correlated

```
matplot(t(age), type='l', xlab='Age', ylab='Number of Residents', lty=1, col=rainbow(33), las=1)
```



← every line is a row of the matrix

We can see that each statistical unit is more or less a straight line (could be well approximated by a straight line)

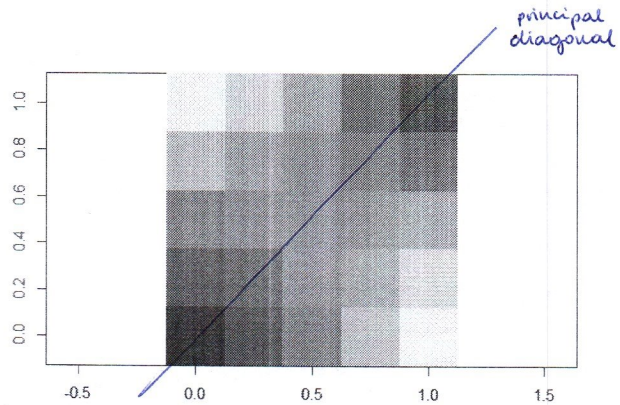
⟹ for describing a line we need only 2 parameters

⟹ dimensional reduction

(dim(space): 5 ⟶ 2)

```
S <- cov(age)
image(S, asp=1)
```

variance/covariance matrix



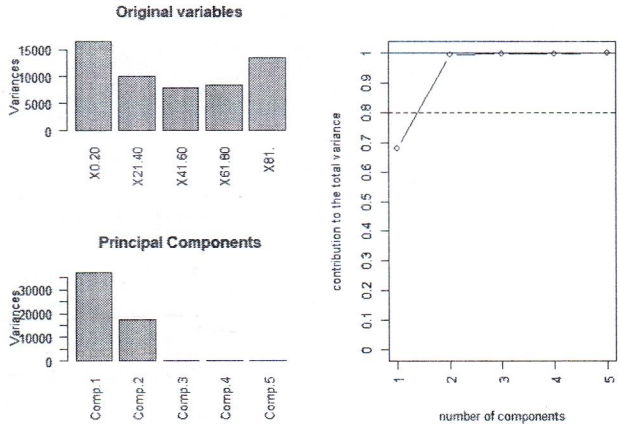

```
var.gen <- det(S)
var.tot <- sum( diag(S) )

# PCA (on the covariance matrix)
pc.age <- princomp(age, scores=T)
pc.age
```

```
## Call:
## princomp(x = age, scores = T)
##
## Standard deviations:
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5
## 192.869739 131.655545  10.978951   9.277125   8.226957
##
##  5  variables and  33 observations.
```

```
summary(pc.age)
```

```
## Importance of components:
##                             Comp.1      Comp.2       Comp.3      Comp.4
## Standard deviation     192.8697390 131.6555445 10.978950563 9.277125047
## Proportion of Variance   0.6787322   0.3162631  0.002199338 0.001570352
## Cumulative Proportion    0.6787322   0.9949954  0.997194699 0.998765052
##                             Comp.5
## Standard deviation     8.226957182
## Proportion of Variance 0.001234948
## Cumulative Proportion  1.000000000
```

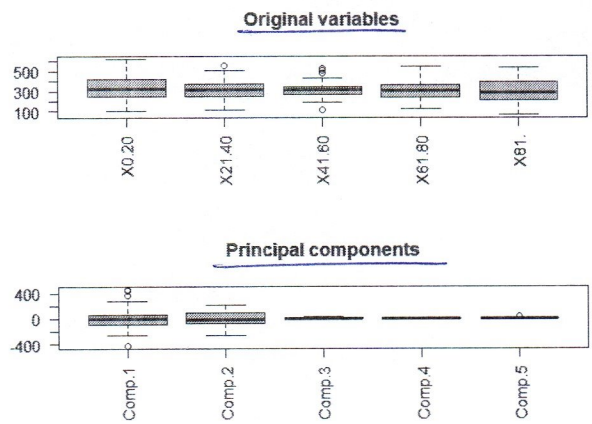
```
# Explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
barplot(pc.age$sdev^2, las=2, main='Principal Components', ylab='Variances')
barplot(sapply(age,sd)^2, las=2, main='Original variables', ylab='Variances')
plot(cumsum(pc.age$sdev^2)/sum(pc.age$sde^2), type='b', axes=F, xlab='number of components', ylab='contribution to the total
variance', ylim=c(0,1))
abline(h=1, col='blue')
abline(h=0.8, lty=2, col='blue')
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(age),labels=1:ncol(age),las=2)
```




```
# Scores
scores.age <- pc.age$scores
head(scores.age)
```

```
##                        Comp.1     Comp.2     Comp.3     Comp.4     Comp.5
## Caithness         -246.159343  -20.59577  -4.203095  16.470275 -2.2983798
## Sutherland         449.508331   89.33650  -3.391313   5.517131  5.1684126
## Ross and Cromarty   -8.760094  179.28694  -7.246958   2.132629 -6.7331927
## Inverness-shire   -103.769656   49.62311  10.419909  -6.790403  0.7991642
## Nairnshire        -117.634818 -149.50143 -18.345831 -11.155232 -8.6693862
## County of Moray   -241.551600  218.02333  14.510531   6.345341  6.4755616
```

```
layout(matrix(c(1,2),2))
boxplot(age, las=2, col='gold', main='Original variables')
scores.age <- data.frame(scores.age)
boxplot(scores.age, las=2, col='gold', main='Principal components')
```

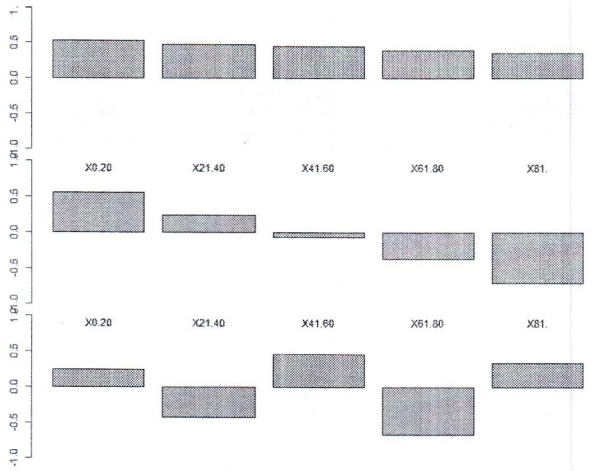


← good
we don't need
to scale the data


```
load.age    <- pc.age$loadings
load.age
```

```
##
## Loadings:
##         Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## X0.20    0.536  0.554  0.249  0.280  0.515
## X21.40   0.482  0.252 -0.417 -0.703 -0.190
## X41.60   0.450         0.465  0.205 -0.731
## X61.80   0.395 -0.361 -0.658  0.530
## X81.     0.349 -0.703  0.339 -0.324  0.405
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5
## SS loadings       1.0    1.0    1.0    1.0    1.0
## Proportion Var    0.2    0.2    0.2    0.2    0.2
## Cumulative Var    0.2    0.4    0.6    0.8    1.0
```

```
x11()
par(mar = c(1,4,0,2), mfrow = c(3,1))
for(i in 1:3)barplot(load.age[,i], ylim = c(-1, 1))
```



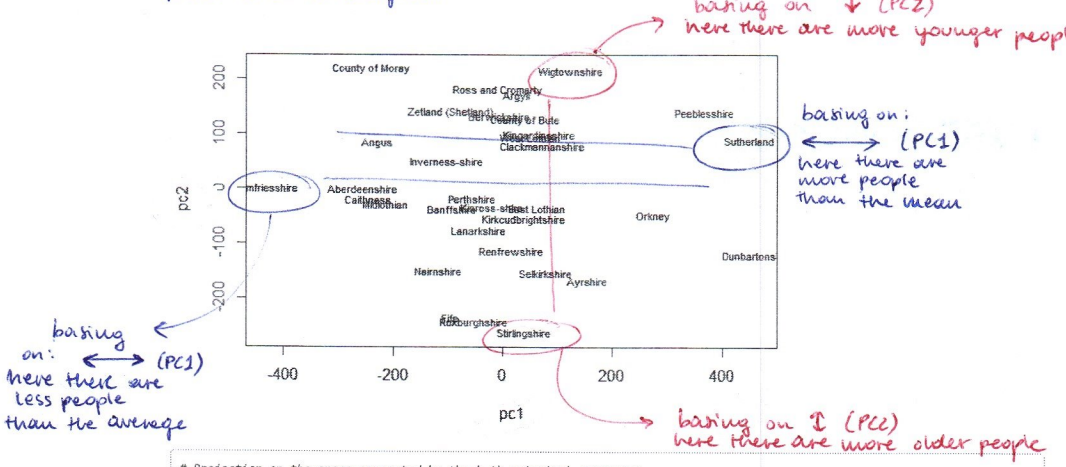
Interpretation:

← average (overall mean of the original variables)

← contrast between younger and older people

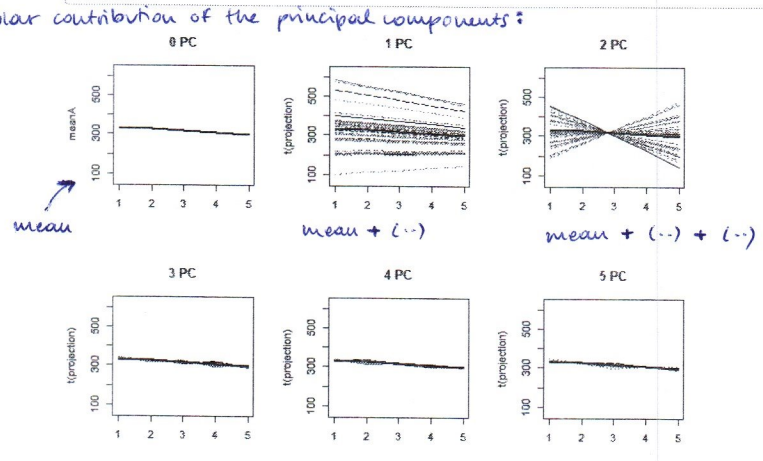
```
x11()
plot(scores.age[,1],scores.age[,2],type="n",xlab="pc1",ylab="pc2", asp=1)
text(scores.age[,1],scores.age[,2],dimnames(age)[[1]], cex=0.7)
```

More interpretation of the weights:




```
# Projection on the space generated by the k-th principal component
x11(width=18, height=7)
par(mfrow=c(2,3))
#matplot(t(age), type='l', main = 'Data', ylim=range(age))
meanA <- colMeans(age)
matplot(meanA, type='l', main = '0 PC', lwd=2, ylim=range(age))
for(i in 1:5){
  projection <- matrix(meanA, dim(age)[[1]], dim(age)[[2]], byrow=T) + scores.age[,i] %*% t(load.age[,i])
  matplot(t(projection), type='l', main = paste(i, 'PC'), ylim=range(age))
  matplot(meanA, type='l', lwd=2, add=T)
}
```

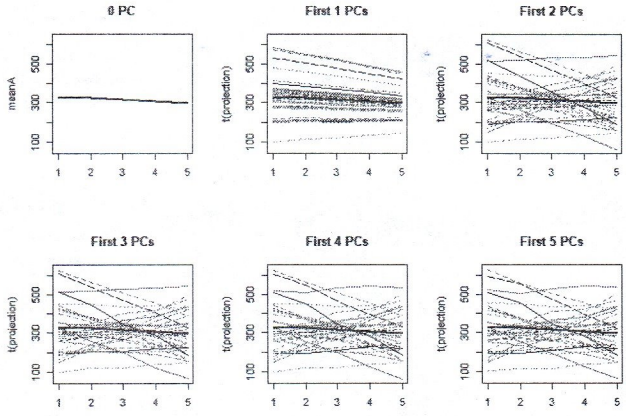
Singular contribution of the principal components:



0PC → gives the mean
1PC → translate vertically
2PC → add a slope
3PC, 4PC, 5PC → almost 0 contributes

```
# Projection on the space generated by the first k principal components
x11(width=18, height=7)
par(mfrow=c(2,3))
#matplot(t(age), type='l', main = 'Data', ylim=range(age))
meanA <- colMeans(age)
matplot(meanA, type='l', main = '0 PC', lwd=2, ylim=range(age))
projection <- matrix(meanA, dim(age)[[1]], dim(age)[[2]], byrow=T)
for(i in 1:5)
{
  projection <- projection + scores.age[,i] %*% t(load.age[,i])
  matplot(t(projection), type='l', main = paste('First', i, 'PCs'), ylim=range(age))
  matplot(meanA, type='l', lwd=2, add=T)
}
```

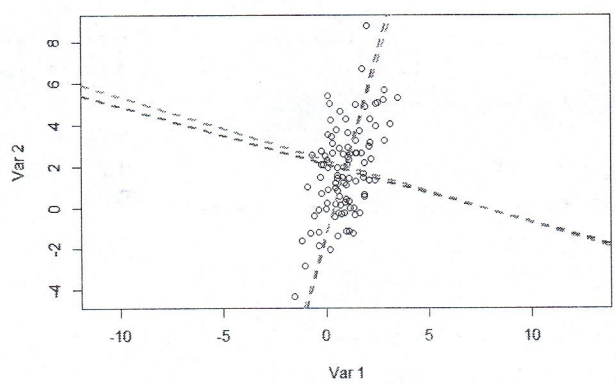
What data can we reconstruct with k principal components?



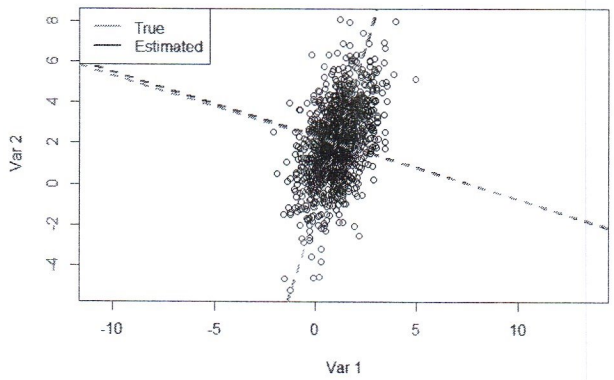
As we saw, 3PC, 4PC and 5PC give almost nothing ⟹ 2 principal components are enough

```
### ------------------------------------------------------------------------
### Additional execises:
# Problem 1.
# Along the ringroads of Milan four control units measure the concentration of the pollutant NO in the air. The measures col
lected during the last year are reported in the file NO.txt. Perform a principal component analysis of the available data. I
n particular:
# (a) Compute the loadings
# (b) Compute the variances along the PCs
# (c) Comment and interpret the results at points (a) and (b)
# (d) On the 3rd July the control unites registered the values (13, 10, 11, 13). Compute the corresponding scores
```

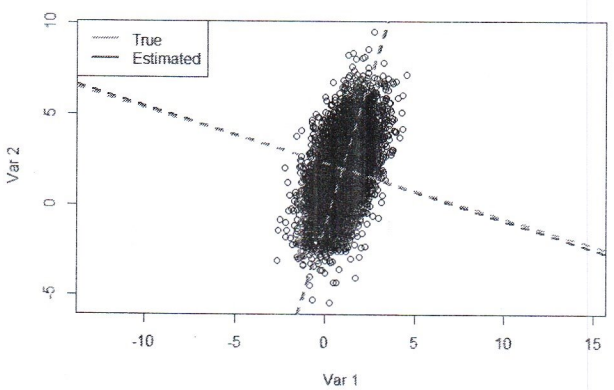
```
### ------------------------------------------------------------------------
### Additional material: Theoretical & sample PCA

# Let's generate a sample of size 100 from a Gaussian
set.seed(27032020)
mu  <-  c(1,2)
sig <-  cbind(c(1,1), c(1,4))
n   <-  100

X   <- rmvnorm(n, mu, sig)
M   <- colMeans(X)
S   <- cov(X)

x11(width=14, height=7)
# par(mfrow=c(1,3))
plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=1)
ellipse(M, S, 1, add=T,lwd=3, col = 'red')
ellipse(mu, sig, 1, add=T,lwd=3, col='springgreen')

abline(a = M[2] - eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1]*M[1], b = eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1], lty = 2
, col = 'red', lwd = 2)
abline(a = M[2] - eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2]*M[1], b = eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2], lty = 2
, col = 'red', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1]*mu[1], b = eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1
], lty = 2, col = 'springgreen', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2]*mu[1], b = eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2
], lty = 2, col = 'springgreen', lwd = 2)

legend('topleft',c('True','Estimated'),col=c('springgreen','red'),lty=c(1,1),lwd=2)
```



```
### let's now increase n...
n <- 1000

X <- rmvnorm(n, mu, sig)
M <- colMeans(X)
S <- cov(X)

plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=1)
ellipse(M, S, 1, add=T,lwd=3, col = 'red')
ellipse(mu, sig, 1, add=T,lwd=3, col='springgreen')

abline(a = M[2] - eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1]*M[1], b = eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1], lty = 2
, col = 'red', lwd = 2)
abline(a = M[2] - eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2]*M[1], b = eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2], lty = 2
, col = 'red', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1]*mu[1], b = eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1
], lty = 2, col = 'springgreen', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2]*mu[1], b = eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2
], lty = 2, col = 'springgreen', lwd = 2)

legend('topleft',c('True','Estimated'),col=c('springgreen','red'),lty=c(1,1),lwd=2)
```




```
###
n <- 5000

X <- rmvnorm(n, mu, sig)
M <- colMeans(X)
S <- cov(X)

plot(X, asp=1, xlab='Var 1', ylab='Var 2',pch=1)
ellipse(M, S, 1, add=T,lwd=3, col = 'red')
ellipse(mu, sig, 1, add=T,lwd=3, col='springgreen')

abline(a = M[2] - eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1]*M[1], b = eigen(S)$vectors[2,1]/eigen(S)$vectors[1,1], lty = 2
, col = 'red', lwd = 2)
abline(a = M[2] - eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2]*M[1], b = eigen(S)$vectors[2,2]/eigen(S)$vectors[1,2], lty = 2
, col = 'red', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1]*mu[1], b = eigen(sig)$vectors[2,1]/eigen(sig)$vectors[1,1
], lty = 2, col = 'springgreen', lwd = 2)
abline(a = mu[2] - eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2]*mu[1], b = eigen(sig)$vectors[2,2]/eigen(sig)$vectors[1,2
], lty = 2, col = 'springgreen', lwd = 2)

legend('topleft',c('True','Estimated'),col=c('springgreen','red'),lty=c(1,1),lwd=2)
```



LAB 03 - Ex. 1

```
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### Example 1. PCA: NO
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### Along the ringroads of Milan four control units measure the concentration of
### the pollutant NO in the air. The measures collected during the last year are
### reported in the file NO.txt. Perform a principal component analysis of the
### available data. In particular:
###   (a) Compute the loadings
###   (b) Compute the variances along the PCs
###   (c) Comment and interpret the results at points (a) and (b)
###   (d) On the 3rd July the control unites registered the values (13, 10, 11, 13).
###       Compute the corresponding scores

# Import the data
NO <- read.table('NO.txt', header=T)
head(NO)
```

```
##      V1    V2    V3    V4
## 1 11.84 10.59  9.47 12.00
## 2 15.79 11.62  7.96 13.40
## 3 11.41  7.00 10.73 17.27
## 4 13.28 10.95  7.60 12.67
## 5  6.89  4.20  3.78 12.03
## 6 14.38 13.25  9.73  9.98
```

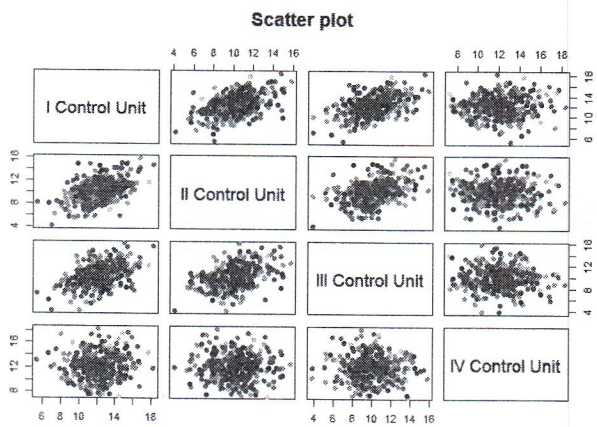
```
dim(NO)
```

```
## [1] 365   4
```

```
head(dimnames(NO))
```

```
## [[1]]
##   [1] "1"   "2"   "3"   "4"   "5"   "6"   "7"   "8"   "9"   "10"  "11"  "12"
##  [13] "13"  "14"  "15"  "16"  "17"  "18"  "19"  "20"  "21"  "22"  "23"  "24"
##  [25] "25"  "26"  "27"  "28"  "29"  "30"  "31"  "32"  "33"  "34"  "35"  "36"
##  [37] "37"  "38"  "39"  "40"  "41"  "42"  "43"  "44"  "45"  "46"  "47"  "48"
##  [49] "49"  "50"  "51"  "52"  "53"  "54"  "55"  "56"  "57"  "58"  "59"  "60"
##  [61] "61"  "62"  "63"  "64"  "65"  "66"  "67"  "68"  "69"  "70"  "71"  "72"
##  ..
## [349] "349" "350" "351" "352" "353" "354" "355" "356" "357" "358" "359" "360"
## [361] "361" "362" "363" "364" "365"
##
## [[2]]
## [1] "V1" "V2" "V3" "V4"
```

```
NO <- data.frame(NO)
var.names <- c("I Control Unit","II Control Unit","III Control Unit","IV Control Unit")
dimnames(NO)[[2]] <- var.names

## ------------------------------------------------------------------------
## DATA EXPLORATION
## ------------------------------------------------------------------------
# Scatter plot
pairs(NO, col=rainbow(dim(NO)[1]), pch=16, main='Scatter plot')
```




```
## Comment
# Positive correlation between I-II, II-III, I-III, low correlation between I-IV,
# II-IV, very low and negative correlation between III-IV.
# This is confirmed by quantitative analyses.
```

```
# we compute the sample mean, sample covariance matrix and sample correlation matrix
M <- sapply(NO,mean)
M
```

```
##   I Control Unit  II Control Unit III Control Unit  IV Control Unit
##        12.173205        10.204110         9.974329        12.043178
```

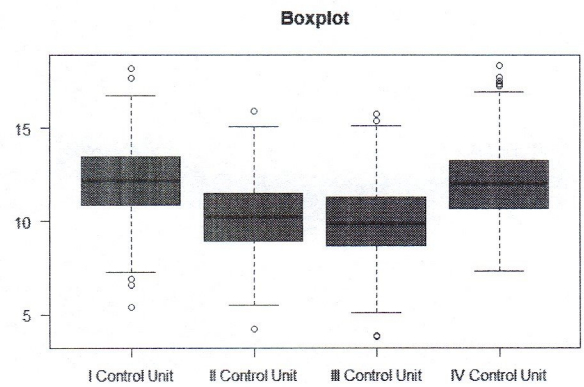
```
S <- cov(NO)
S
```

```
##                  I Control Unit II Control Unit III Control Unit
## I Control Unit        3.7946999      1.74995849       1.79825175
## II Control Unit       1.7499585      3.66873526       1.62541623
## III Control Unit      1.7982517      1.62541623       3.91388341
## IV Control Unit       0.1131931     -0.08197216      -0.02001242
##                  IV Control Unit
## I Control Unit        0.11319308
## II Control Unit      -0.08197216
## III Control Unit     -0.02001242
## IV Control Unit       4.23889317
```

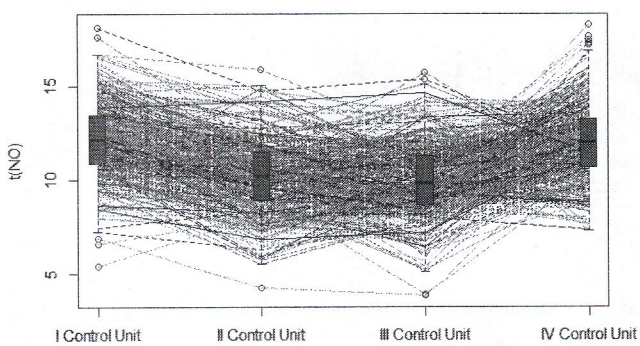
```
R <- cor(NO)
R
```

```
##                  I Control Unit II Control Unit III Control Unit
## I Control Unit       1.00000000      0.46900864       0.466614073
## II Control Unit      0.46900864      1.00000000       0.428945842
## III Control Unit     0.46661407      0.42894584       1.000000000
## IV Control Unit      0.02822311     -0.02078653      -0.004913257
##                  IV Control Unit
## I Control Unit       0.028223114
## II Control Unit     -0.020786529
## III Control Unit    -0.004913257
## IV Control Unit      1.000000000
```

```
# Boxplot
x11()
boxplot(NO, las=1, col='red', main='Boxplot',grid=T)
```




```
# Matplot + boxplot
x11()
matplot(t(NO), type='l', axes=F)
box()
boxplot(NO, add=T, boxwex=0.1, col='red')
```




```
## Comment
# The variability of the variables are similar, medians of the II and IV control
# units seem lower (as well as the means); almost symmetric distributions, with
# little asymmetry in the measurements of the III control unit. Presence of outliers
# in all the variables. In particular, there is an outlying measurement (NO[5,])
# for the CUs I-II-III, not for the CU IV.
```

```
## ------------------------------------------------------------------------
# (a) Compute the loadings
# (b) Compute the variances along the PCs
# (c) Comment and interpret the results at points (a) and (b)
## ------------------------------------------------------------------------
```

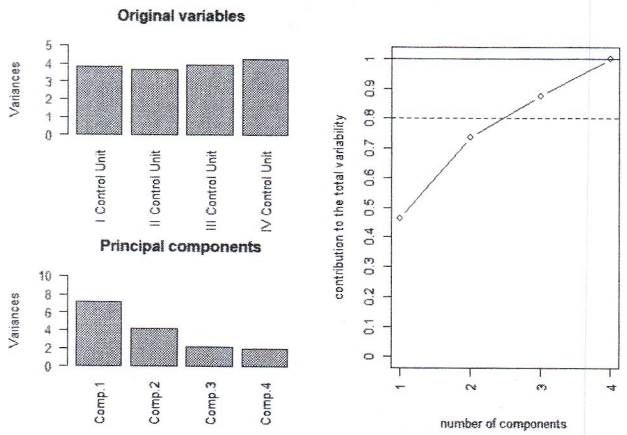
```
## ------------------------------------------------------------------------
## PRINCIPAL COMPONENT ANALYSIS
## ------------------------------------------------------------------------

## Comment
# The original variability along the 3 variables is similar; the units of measure
# of the variables are homogeneous. We thus perform a PCA based on S.

pca.NO <- princomp(NO, scores=T)
pca.NO
```

```
## Call:
## princomp(x = NO, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4
## 2.688024 2.058132 1.471993 1.394733
##
##  4  variables and  365 observations.
```

```
summary(pca.NO)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4
## Standard deviation     2.6880243 2.0581324 1.4719932 1.3947329
## Proportion of Variance 0.4639618 0.2719959 0.1391321 0.1249102
## Cumulative Proportion  0.4639618 0.7359577 0.8750898 1.0000000
```

```
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
barplot(pca.NO$sdev^2, las=2, main='Principal components', ylim=c(0,10), ylab='Variances')
barplot(sapply(NO,sd)^2, las=2, main='Original variables', ylim=c(0,5), ylab='Variances')
plot(cumsum(pca.NO$sdev^2)/sum(pca.NO$sde^2), type='b', axes=F, xlab='number of components',
     ylab='contribution to the total variability', ylim=c(0,1))
abline(h=1, col='blue')
abline(h=0.8, lty=2, col='blue')
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(NO),labels=1:ncol(NO),las=2)
```




```
## Comment
# Dimensionality reduction: the first two components explain 73.6%<80% of the total
# variability. However, we do not see an elbow in the proportion of explained
# variability in correspondence of the III or IV PC. It could be thus sufficient to
# analyse the sample through the first 2 PCs.
# Note. The results of the PCA on the correlation matrix would not give very
# different results
```

```
NO.sd <- scale(NO)
pca.NO.std <- princomp(NO.sd, scores=T)
pca.NO.std
```

```
## Call:
## princomp(x = NO.sd, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4
## 1.3801180 0.9999294 0.7545120 0.7177522
##
##  4  variables and  365 observations.
```

```
summary(pca.NO.std) # PCA on R
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4
## Standard deviation     1.3801180 0.9999294 0.7545120 0.7177522
## Proportion of Variance 0.4774896 0.2506514 0.1427131 0.1291459
## Cumulative Proportion  0.4774896 0.7281410 0.8708541 1.0000000
```

```
summary(pca.NO)     # PCA on S
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4
## Standard deviation     2.6880243 2.0581324 1.4719932 1.3947329
## Proportion of Variance 0.4639618 0.2719959 0.1391321 0.1249102
## Cumulative Proportion  0.4639618 0.7359577 0.8750898 1.0000000
```

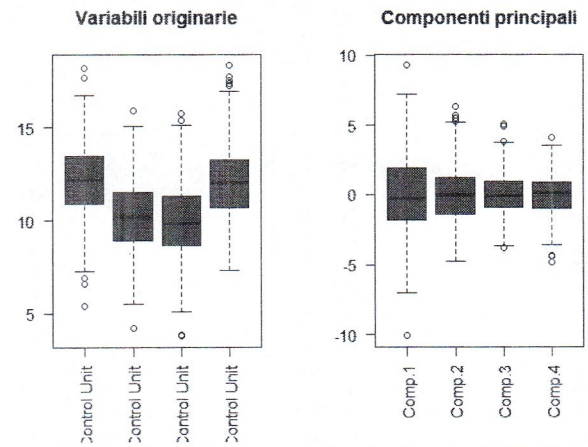
```
## ------------------------------------------------------------------------
## SCORES AND LOADINGS (PCA on S)
## ------------------------------------------------------------------------
# Scores
scores.NO <- pca.NO$scores
head(scores.NO)
```

```
##        Comp.1      Comp.2     Comp.3     Comp.4
## 1  -0.2785931 -0.06730603  0.5600512 -0.3469115
## 2   1.7323036  1.48547641  3.1065049  2.4751860
## 3  -1.7665394  5.30161988 -2.6533004  0.6867818
## 4  -0.3298125  0.67513843  2.4924304  1.0172964
## 5 -10.0835065  0.06676850  0.0746020  0.8027132
## 6   2.8383352 -2.07896691  2.4578184  0.2113521
```

```
summary(scores.NO)
```

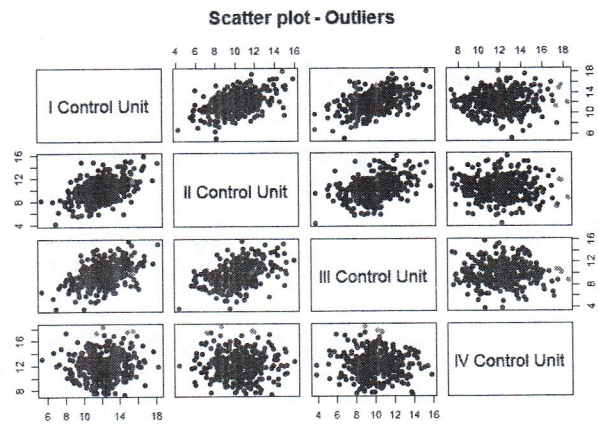
```
##      Comp.1             Comp.2             Comp.3             Comp.4
##  Min.   :-10.0835   Min.   :-4.76672   Min.   :-3.7747   Min.   :-4.8098
##  1st Qu.: -1.7665   1st Qu.:-1.41806   1st Qu.:-0.9286   1st Qu.:-0.9650
##  Median : -0.2492   Median :-0.03467   Median :-0.0831   Median : 0.1561
##  Mean   :  0.0000   Mean   : 0.00000   Mean   : 0.0000   Mean   : 0.0000
##  3rd Qu.:  1.9403   3rd Qu.: 1.26260   3rd Qu.: 0.9549   3rd Qu.: 0.9090
##  Max.   :  9.2573   Max.   : 6.33575   Max.   : 5.0823   Max.   : 4.1276
```

```
layout(matrix(c(1,2),1,2))
boxplot(NO, las=2, col='red', main='Variabili originarie')
scores.NO <- data.frame(scores.NO)
boxplot(scores.NO, las=2, col='red', main='Componenti principali')
```




```
# pairs(scores.NO, col=rainbow(dim(scores.NO)[1]), pch=16)
# cor(scores.NO)

## Comment
# The variability of the first two PCs (looking at the size of the boxes) is greater
# than that of the components 3 and 4. These have very similar variability: we
# should avoid to keep 3 PCs, let's keep 2 or 4 PCs instead.
# The proportion of variability explained by the second PC is influenced by the
# presence of 4 outliers. These represents 4 out of the 5 outliers of the
# IV CU (NO[3,4], NO[59,4], NO[125,4], NO[308,4]). The outlier of PC 1 corresponds
# to the outlying observation mentioned before and to the measurement NO[217,].
```

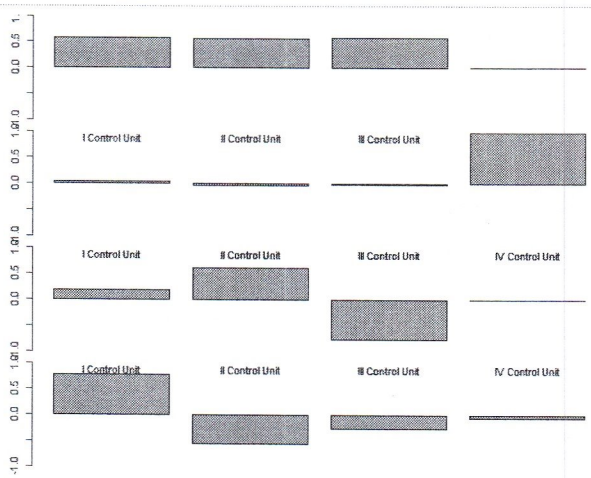
```
# We plot the outlying data for the first 2 PCs
color.outliers <- NULL
for (i in 1:365){
  color.outliers[i] <- 'blue'
}
color.outliers[5] <- 'red'
color.outliers[217] <- 'red'
color.outliers[3] <- 'green'
color.outliers[125] <- 'green'
color.outliers[59] <- 'green'
color.outliers[308] <- 'green'
pairs(NO, col=color.outliers, pch=16, main='Scatter plot - Outliers')
```



```
# Loadings
load.NO     <- pca.NO$loadings
load.NO
```

```
##
## Loadings:
##                   Comp.1 Comp.2 Comp.3 Comp.4
## I Control Unit     0.588         0.189  0.785
## II Control Unit    0.555         0.613 -0.561
## III Control Unit   0.588        -0.767 -0.255
## IV Control Unit           0.998
##
##                 Comp.1 Comp.2 Comp.3 Comp.4
## SS loadings      1.00   1.00   1.00   1.00
## Proportion Var   0.25   0.25   0.25   0.25
## Cumulative Var   0.25   0.50   0.75   1.00
```

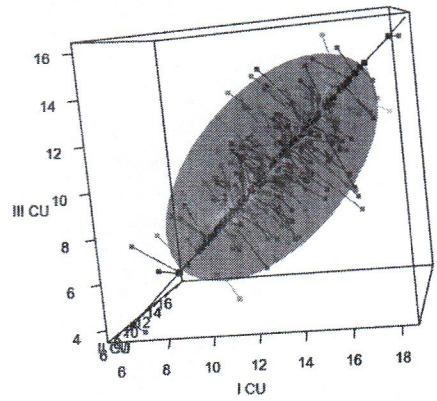
```
x11()
par(mar = c(1,4,0,2), mfrow = c(4,1))
for(i in 1:4)
  barplot(load.NO[,i], ylim = c(-1, 1))
```



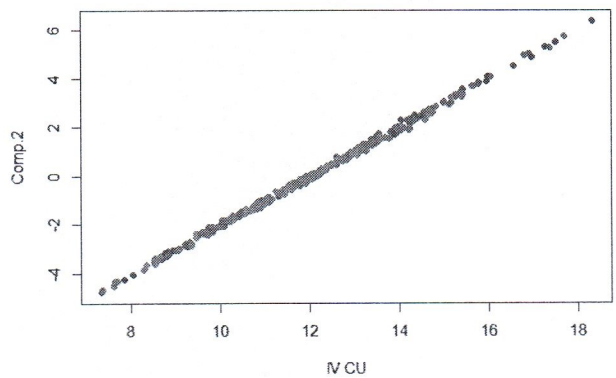

```
## Comment
# The first PC is a weighted mean (with very similar weights) of the measurements
# recorded by the CU I-II-III; the second PC corresponds to the IV CU. The last
# two PCs do not have a clear interpretations, although they are combinations of the
# CU I-II-III.
# Note: it makes sense that the outliers of the PC1 correspond to the ouliers of the
# CU I-II-III, and the outlier of the PC2 are those of CU IV.
```

```
# We can geometrically represent in a 3D space the dimensionality reduction, since
# the loadings of the CU IV are almost 0 in that direction

library(rgl)

M <- sapply(NO[,1:3],mean)
S <- cov(NO[,1:3])

open3d()
```

```
points3d(NO[,1:3], col=rainbow(dim(NO)[1]), asp=1, size=5)
axes3d()
plot3d(ellipse3d(S, centre=M, level= 9/10), alpha=0.25, add = TRUE)
title3d(xlab="I CU",ylab="II CU",zlab="III CU")
pca.NO1 <- NULL
for(i in 1:365)
  pca.NO1 <- rbind(pca.NO1, pca.NO$loadings[1:3,1]*pca.NO$scores[i,1] + M)
points3d(pca.NO1, col='black', size=6)
lines3d(rbind(M + 4*pca.NO$sdev[1] * pca.NO$loadings[1:3,1], M - 4*pca.NO$sdev[1] * pca.NO$loadings[1:3,1]), col='black')
color=rainbow(dim(NO)[1])
for(i in 1:365)
  lines3d(rbind(NO[i,], pca.NO1[i,]),col=color[i])

a = scene3d()
rgl.close()
x11()
rglwidget(a)
```




```
open3d()
points3d(NO[,1:3], col=color.outliers, asp=1, size=5)
axes3d()
plot3d(ellipse3d(S, centre=M, level= 9/10), alpha=0.25, add = TRUE)
title3d(xlab="I CU",ylab="II CU",zlab="III CU")
pca.NO1 <- NULL
for(i in 1:365)
  pca.NO1 <- rbind(pca.NO1, pca.NO$loadings[1:3,1]*pca.NO$scores[i,1] + M)
points3d(pca.NO1, col='black', size=5)

a = scene3d()
rgl.close()
x11()
rglwidget(a)

# We separately represent the PC2, and compare it with the CU IV variable
# (in this case, the loadings corresponding to the CUs I-II-IV are almost zero)

n=5000
x11()
plot(NO[,4],scores.NO[,2],col=rainbow(n),pch=19,xlab='IV CU',ylab='Comp.2')
```




```
## ------------------------------------------------------------------------
## TRY TO REMOVE THE OUTLIERS
## ------------------------------------------------------------------------
NO.outliers <- NO[which(color.outliers=='blue'),]
dim(NO.outliers)
```

```
## [1] 359   4
```

```
pca.NO.outliers <- princomp(NO.outliers, scores=T)
pca.NO.outliers
```

```
## Call:
## princomp(x = NO.outliers, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4
## 2.600639 1.984580 1.474930 1.400978
##
##  4  variables and  359 observations.
```

```
summary(pca.NO.outliers)
```

```
## Importance of components:
##                           Comp.1    Comp.2    Comp.3    Comp.4
## Standard deviation     2.6006390 1.9845804 1.4749302 1.4009782
## Proportion of Variance 0.4557483 0.2654008 0.1465912 0.1322597
## Cumulative Proportion  0.4557483 0.7211491 0.8677403 1.0000000
```

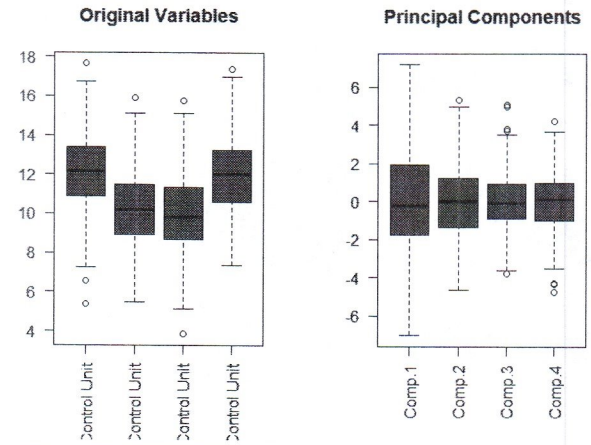
```
## ------------------------------------------------------------------------
## SCORES AND LOADINGS
## ------------------------------------------------------------------------
# Scores
scores.NO.outliers <- pca.NO.outliers$scores
head(scores.NO.outliers)
```

```
##       Comp.1       Comp.2     Comp.3     Comp.4
## 1 -0.2769064  0.004925539 0.55816006 -0.3305162
## 2  1.7013410  1.503092927 3.09734866  2.5501179
## 4 -0.3475351  0.713307423 2.48157359  1.0632869
## 6  2.8488080 -1.984784917 2.47740255  0.1819975
## 7 -0.7406506  4.936470034 5.06293021 -0.7618585
## 8  1.1581293 -3.298881020 0.07744176 -0.7436079
```

```
summary(scores.NO.outliers)
```

```
##      Comp.1             Comp.2             Comp.3             Comp.4
##  Min.   :-7.0450   Min.   :-4.63056   Min.   :-3.76782   Min.   :-4.7252
##  1st Qu.:-1.7567   1st Qu.:-1.34567   1st Qu.:-0.92050   1st Qu.:-0.9785
##  Median :-0.2391   Median : 0.02962   Median :-0.08221   Median : 0.1324
##  Mean   : 0.0000   Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.0000
##  3rd Qu.: 1.9323   3rd Qu.: 1.25459   3rd Qu.: 0.92823   3rd Qu.: 0.9615
##  Max.   : 7.1849   Max.   : 5.31480   Max.   : 5.06293   Max.   : 4.1976
```

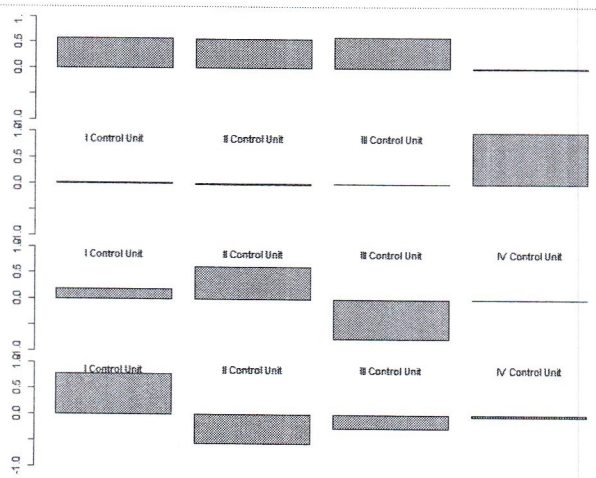
```
layout(matrix(c(1,2),1,2))
boxplot(NO.outliers, las=2, col='red', main='Original Variables')
scores.NO.outliers <- data.frame(scores.NO.outliers)
boxplot(scores.NO.outliers, las=2, col='red', main='Principal Components')
```




```
# Loadings
load.NO.outliers    <- pca.NO.outliers$loadings
load.NO.outliers
```

```
##
## Loadings:
##                  Comp.1 Comp.2 Comp.3 Comp.4
## I Control Unit    0.586         0.189  0.788
## II Control Unit   0.553         0.618 -0.559
## III Control Unit  0.593        -0.763 -0.258
## IV Control Unit          0.999
##
##                Comp.1 Comp.2 Comp.3 Comp.4
## SS loadings      1.00   1.00   1.00   1.00
## Proportion Var   0.25   0.25   0.25   0.25
## Cumulative Var   0.25   0.50   0.75   1.00
```

```
x11()
par(mar = c(1,4,0,2), mfrow = c(4,1))
for(i in 1:4)
  barplot(load.NO.outliers[,i], ylim = c(-1, 1))
```

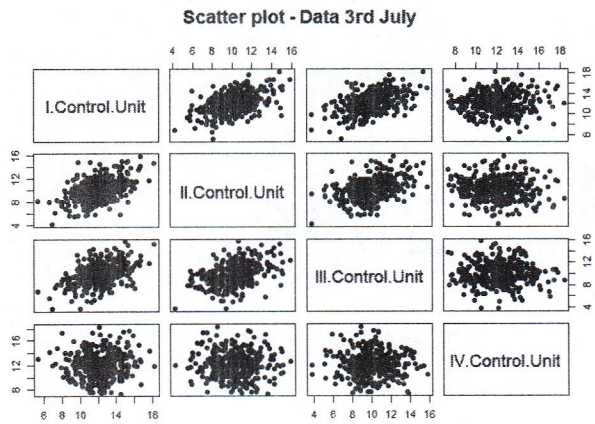


```
## Comment
# The results obtained by removing the outliers are not very different from those
# obtained before. Both the analyses lead to the same choice to keep 2 PCs for the
# purpose of dimensionality reduction.
#
# We finally note that if we keep the first 2 PCs we get that:
#   * the first PC is roughly the mean of the observations of the CUs I-II-III
#   * the second PC corresponds to the IV CU.
#
# From the application viewpoint, this could suggest a malfunctioning of the IV CU,
# whose measurements are actually uncorrelated with those of the CU I-II-III,
# although they are geographically near.
```

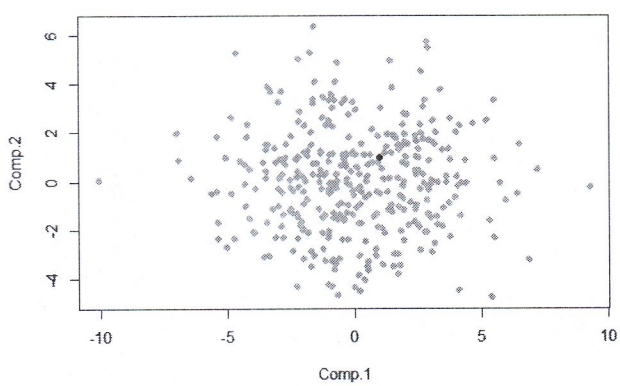
```
## ------------------------------------------------------------------------------
## (d) On the 3rd July the control unites registered the values (13, 10, 11, 13).
##     Compute the corresponding scores
## ------------------------------------------------------------------------------

data.3jul <- c(13, 10, 11, 13)
scores.3jul <- t(pca.NO$loadings)%*%(data.3jul-colMeans(NO))
scores.3jul
```

```
##              [,1]
## Comp.1  0.9792882
## Comp.2  0.9876704
## Comp.3 -0.7493563
## Comp.4  0.4435386
```

```
x11()
pairs(data.frame(rbind(NO,data.3jul)), col=c(rep(1,n),2), pch=16, main='Scatter plot - Data 3rd July')
```

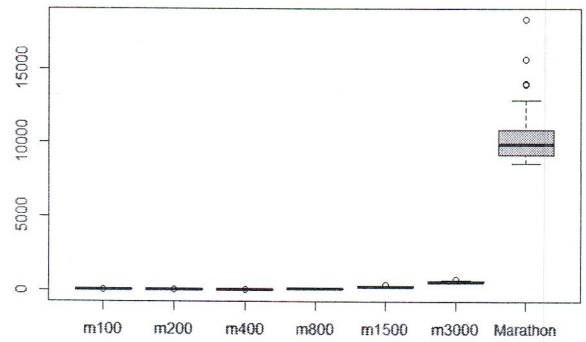



```
x11()
plot(scores.NO[,1],scores.NO[,2],col='grey',pch=19,xlab='Comp.1',ylab='Comp.2')
points(scores.3jul[1],scores.3jul[2],col='black',pch=19)
```



LAB 03 - Ex. 2

```
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### Example 2. PCA: running records
### ------------------------------------------------------------------------
### ------------------------------------------------------------------------
### Principal component analysis of the dataset 'running records'

runrec <- read.table('record_mod.txt', header=T)
n <- dim(runrec)[1]
p <- dim(runrec)[2]

# We make the units of measure homogeneous across the variables
runrec[,4:7] <- runrec[,4:7]*60

# Boxplot
x11()
boxplot(runrec)
```




```
## Comment
# We observe that the variability increases non-linearly for increasing lengths;
# the record times of the marathon have much more variability than that of the other
# disciplines. This could significanlty influence the PCA.
```

```
# We perform the PCA on original data
pc.runrec <- princomp(runrec, scores=T)
pc.runrec
```

```
## Call:
## princomp(x = runrec, scores = T)
##
## Standard deviations:
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
## 1809.7274212   23.0879938    4.9156271    2.8218745    1.1549965    0.4796997
##       Comp.7
##    0.1192800
##
##  7  variables and  55 observations.
```

```
summary(pc.runrec)
```

```
## Importance of components:
##                             Comp.1       Comp.2       Comp.3       Comp.4
## Standard deviation     1809.727421 2.308799e+01 4.915627e+00 2.821874e+00
## Proportion of Variance    0.999827 1.627312e-04 7.376602e-06 2.430938e-06
## Cumulative Proportion     0.999827 9.999897e-01 9.999971e-01 9.999995e-01
##                              Comp.5       Comp.6       Comp.7
## Standard deviation     1.154997e+00 4.796997e-01 1.192800e-01
## Proportion of Variance 4.072488e-07 7.024855e-08 4.343441e-09
## Cumulative Proportion  9.999999e-01 1.000000e+00 1.000000e+00
```

```
# To obtain the rows of the summary: standard deviation of the components
pc.runrec$sd
```

```
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
## 1809.7274212   23.0879938    4.9156271    2.8218745    1.1549965    0.4796997
##       Comp.7
##    0.1192800
```

```
# proportion of variance explained by each PC
pc.runrec$sd^2/sum(pc.runrec$sd^2)
```

```
##       Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
## 9.998270e-01 1.627312e-04 7.376602e-06 2.430938e-06 4.072488e-07 7.024855e-08
##       Comp.7
## 4.343441e-09
```

```
# cumulative proportion of explained variance
cumsum(pc.runrec$sd^2)/sum(pc.runrec$sd^2)
```

```
##     Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6     Comp.7
## 0.9998270 0.9999897 0.9999971 0.9999995 0.9999999 1.0000000 1.0000000
```

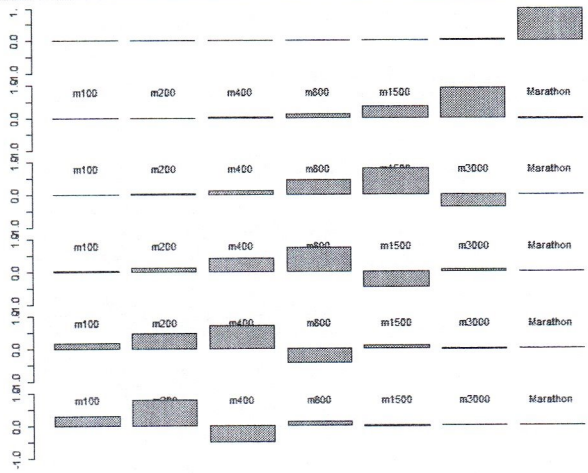
```
# loadings (recall: coefficients of the linear combination of the original variables that defines each principal component)

load.rec <- pc.runrec$loadings
load.rec
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## m100                                0.177  0.304  0.935
## m200                         0.123  0.478  0.794 -0.354
## m400                  0.131  0.435  0.730 -0.508
## m800             0.111  0.445  0.758 -0.443  0.137
## m1500            0.368  0.800 -0.462  0.100
## m3000            0.922 -0.379
## Marathon  1.000
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.143  0.143  0.143  0.143  0.143  0.143  0.143
## Cumulative Var  0.143  0.286  0.429  0.571  0.714  0.857  1.000
```

```
load.rec[,1:7]
```

```
##                Comp.1       Comp.2        Comp.3        Comp.4        Comp.5
## m100     0.0001699579  0.005630797  0.0068596018  4.008225e-02  0.1766771873
## m200     0.0004171012  0.010489600  0.0221061157  1.225880e-01  0.4783674679
## m400     0.0010346402  0.039743391  0.1309810990  4.354082e-01  0.7303280282
## m800     0.0027711040  0.111409800  0.4452572040  7.580689e-01 -0.4426000037
## m1500    0.0095891715  0.368162610  0.8001034816 -4.624252e-01  0.1003525395
## m3000    0.0243720652  0.921752699 -0.3793265361  7.266225e-02 -0.0246028077
## Marathon 0.9996524886 -0.026359732  0.0001929499  5.424241e-05 -0.0001214101
##                Comp.6        Comp.7
## m100      0.303917333  9.352722e-01
## m200      0.793938401 -3.538356e-01
## m400     -0.508183609  7.312232e-03
## m800      0.136951957  2.681640e-03
## m1500    -0.016973341 -1.710239e-03
## m3000     0.001243242 -1.642164e-03
## Marathon -0.000104103  3.006495e-05
```

```
# graphical representation of the loadings of the first six principal components
x11()
par(mar = c(1,4,0,2), mfrow = c(6,1))
for(i in 1:6) barplot(load.rec[,i], ylim = c(-1, 1))
```

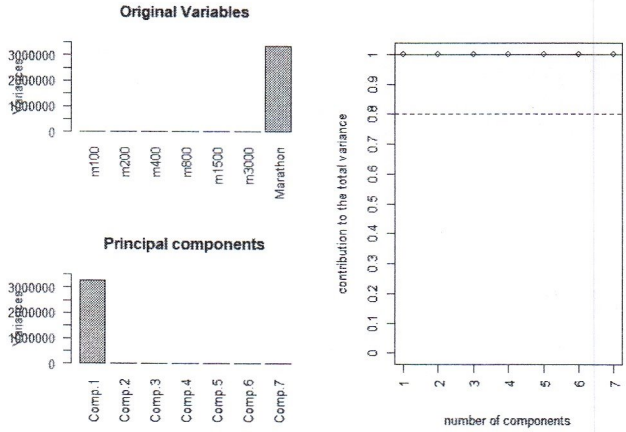


Interpretation of the loadings:

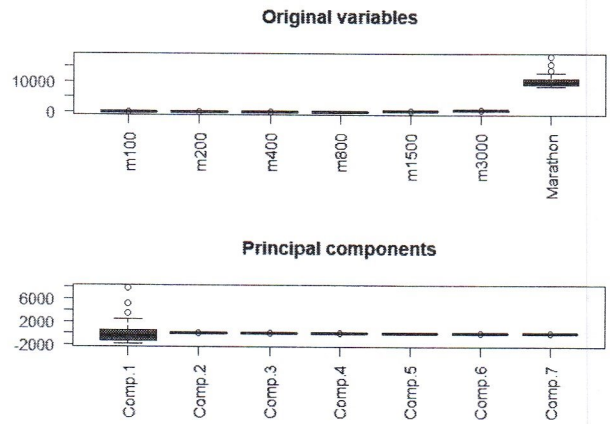
- First PCs: long distances disciplines (I PC: variable 'Marathon')
- Last PCs: short distances disciplines

The loadings reflect the previous observation: the first PC is represented by the variable "Marathon", the second by the long distances, etc. The short distances appear in the last PCs.

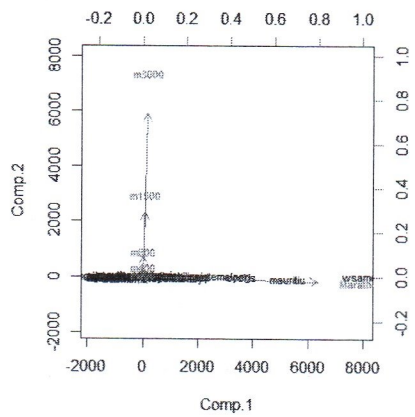
```
# Explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
plot(pc.runrec, las=2, main='Principal components', ylim=c(0,3.5e6))
barplot(sapply(runrec,sd)^2, las=2, main='Original Variables', ylim=c(0,3.5e6), ylab='Variances')
plot(cumsum(pc.runrec$sd^2)/sum(pc.runrec$sd^2), type='b', axes=F, xlab='number of components',
     ylab='contribution to the total variance', ylim=c(0,1))
abline(h=1, col='blue')
abline(h=0.8, lty=2, col='blue')
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(runrec),labels=1:ncol(runrec),las=2)
```




```
# The first PC (var. Marathon) explains more than 99.98% of the total variability. This is due to the masking effect of that
variable over the others

# scores
scores.runrec <- pc.runrec$scores

layout(matrix(c(1,2),2))
boxplot(runrec, las=2, col='red', main='Original variables')
scores.runrec <- data.frame(scores.runrec)
boxplot(scores.runrec, las=2, col='red', main='Principal components')
```




```
# biplot
x11()
biplot(pc.runrec, scale=0, cex=.7)
```




```
graphics.off()
```

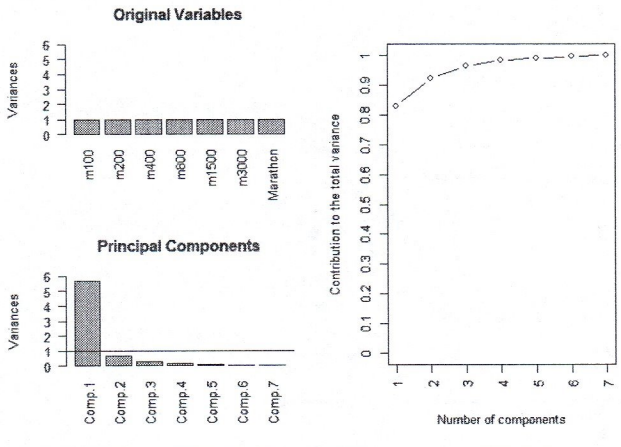
```
### ------------------------------------------------------------------------
### Principal component analysis of the dataset 'runningrecords', but on the
### standardized variables

# We compute the standardized variables
runrec.sd <- scale(runrec)
runrec.sd <- data.frame(runrec.sd)

pc.runrec <- princomp(runrec.sd, scores=T)
pc.runrec
```

```
## Call:
## princomp(x = runrec.sd, scores = T)
##
## Standard deviations:
##    Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6    Comp.7
## 2.3874940 0.8010999 0.5426141 0.3509930 0.2298661 0.1958042 0.1484404
##
##  7  variables and  55 observations.
```

```
summary(pc.runrec)
```

```
## Importance of components:
##                             Comp.1     Comp.2     Comp.3     Comp.4      Comp.5
## Standard deviation     2.3874940 0.80109990 0.54261407 0.35099299 0.229866106
## Proportion of Variance 0.8293837 0.09337793 0.04284035 0.01792536 0.007688131
## Cumulative Proportion  0.8293837 0.92276161 0.96560196 0.98352731 0.991215445
##                             Comp.6      Comp.7
## Standard deviation     0.195804234 0.148440403
## Proportion of Variance 0.005578469 0.003206086
## Cumulative Proportion  0.996793914 1.000000000
```

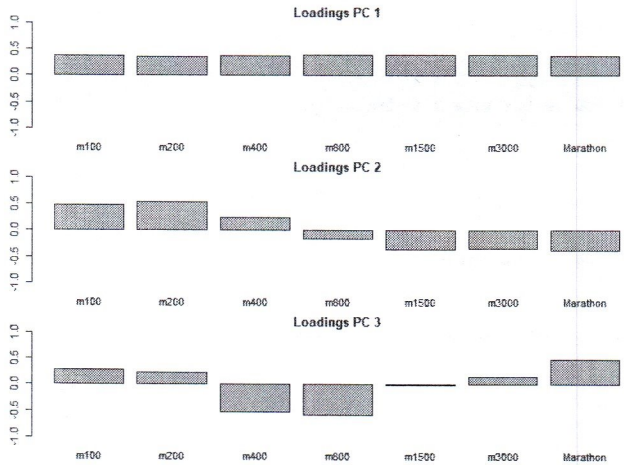
```
# Explained variance
x11()
layout(matrix(c(2,3,1,3),2,byrow=T))
plot(pc.runrec, las=2, main='Principal Components', ylim=c(0,6))
abline(h=1, col='blue')
barplot(sapply(runrec.sd,sd)^2, las=2, main='Original Variables', ylim=c(0,6), ylab='Variances')
plot(cumsum(pc.runrec$sde^2)/sum(pc.runrec$sde^2), type='b', axes=F, xlab='Number of components', ylab='Contribution to the
 total variance', ylim=c(0,1))
box()
axis(2,at=0:10/10,labels=0:10/10)
axis(1,at=1:ncol(runrec.sd),labels=1:ncol(runrec.sd),las=2)
```




```
# If we wanted to perform dimensionality reduction, we could keep 1 or 2 PCs

# Loadings
load.rec <- pc.runrec$loadings
load.rec
```

```
##
## Loadings:
##          Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## m100      0.368  0.490  0.286  0.319  0.231  0.620
## m200      0.365  0.537  0.230                -0.711 -0.109
## m400      0.382  0.247 -0.515 -0.347 -0.572  0.191  0.208
## m800      0.385 -0.155 -0.585         0.620        -0.315
## m1500     0.389 -0.360         0.430        -0.231  0.693
## m3000     0.389 -0.348  0.153  0.363 -0.463        -0.598
## Marathon  0.367 -0.369  0.484 -0.672  0.131  0.142
##
##                Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6 Comp.7
## SS loadings     1.000  1.000  1.000  1.000  1.000  1.000  1.000
## Proportion Var  0.143  0.143  0.143  0.143  0.143  0.143  0.143
## Cumulative Var  0.143  0.286  0.429  0.571  0.714  0.857  1.000
```

```
x11()
par(mar = c(2,2,2,1), mfrow=c(3,1))
for(i in 1:3)barplot(load.rec[,i], ylim = c(-1, 1), main=paste('Loadings PC ',i,sep=''))
```



Interpretation of the loadings: In this case, the first PC represents an average of the times of all the disciplines, taken with very similar (positive) weights. The sencond PC contrasts the short distances (m100, m200, m400) with the long distances (m800, m1500, m3000, Marathon)

- High PC1: long times in all the disciplines
- Low PC1: short times in all the disciplines
- High PC2: long times in short distances, short times in long distances
- Low PC2: short times in short distances, long times in long distances

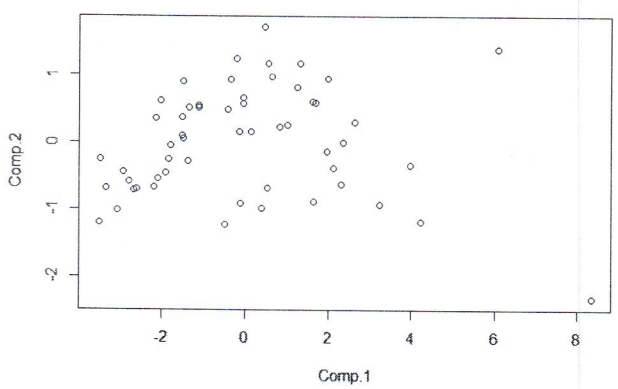
```
# scores
scores.runrec <- pc.runrec$scores
head(scores.runrec)
```

```
##              Comp.1      Comp.2      Comp.3      Comp.4       Comp.5
## argentin  0.5272623 -0.67471906 -0.61558926  0.04552931 -0.002557586
## australi -2.0935512 -0.53279869  0.04317487  0.18737792 -0.218355512
## austria  -1.3804433 -0.27499549  0.53231305  0.42623519 -0.025503919
## belgium  -1.5099897  0.09096395  0.08345684 -0.03942270 -0.030326372
## bermuda   0.3878176 -0.97640948 -0.64887210  0.47445652  0.204321612
## brazil   -0.1183973 -0.91151551 -0.32214131  0.34096557 -0.095961277
##               Comp.6       Comp.7
## argentin  0.45790793 -0.079962873
## australi  0.13796693 -0.009815157
## austria  -0.08168270 -0.106172559
## belgium   0.06287733  0.138490873
## bermuda  -0.04942635  0.047829472
## brazil   -0.30044807 -0.006724259
```

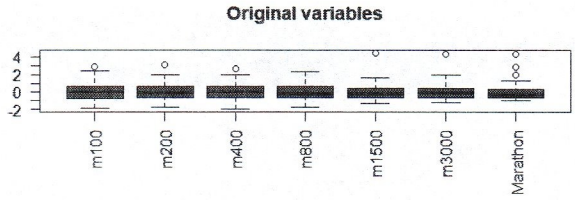
```
summary(scores.runrec)
```

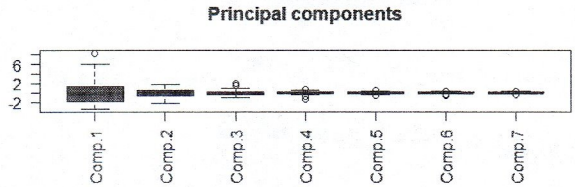
```
##      Comp.1            Comp.2             Comp.3             Comp.4
##  Min.   :-3.506   Min.   :-2.32698   Min.   :-1.09460   Min.   :-1.39411
##  1st Qu.:-1.811   1st Qu.:-0.64317   1st Qu.:-0.35897   1st Qu.:-0.15754
##  Median :-0.143   Median : 0.05938   Median :-0.08087   Median :-0.03942
##  Mean   : 0.000   Mean   : 0.00000   Mean   : 0.00000   Mean   : 0.00000
##  3rd Qu.: 1.455   3rd Qu.: 0.57758   3rd Qu.: 0.20702   3rd Qu.: 0.21349
##  Max.   : 8.333   Max.   : 1.71789   Max.   : 1.91964   Max.   : 0.84655
##      Comp.5              Comp.6             Comp.7
##  Min.   :-0.5847203   Min.   :-0.56188   Min.   :-0.442231
##  1st Qu.:-0.1308812   1st Qu.:-0.08631   1st Qu.:-0.093933
##  Median :-0.0006284   Median : 0.01205   Median :-0.006724
##  Mean   : 0.0000000   Mean   : 0.00000   Mean   : 0.000000
##  3rd Qu.: 0.1361278   3rd Qu.: 0.10482   3rd Qu.: 0.074398
##  Max.   : 0.6034323   Max.   : 0.45791   Max.   : 0.376903
```

```
x11()
plot(scores.runrec[,1:2])
```

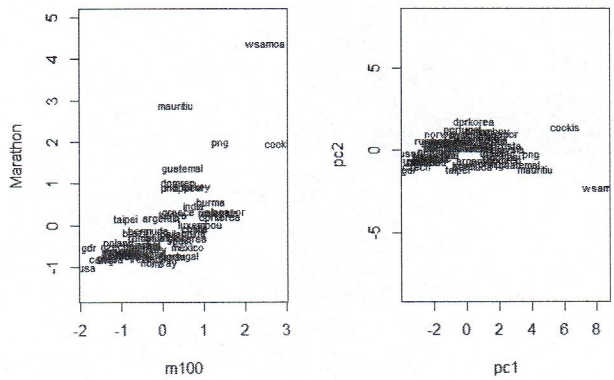



```
x11()
layout(matrix(c(1,2),2))
boxplot(runrec.sd, las=2, col='red', main='Original variables')
scores.runrec <- data.frame(scores.runrec)
boxplot(scores.runrec, las=2, col='red', main='Principal components')
```

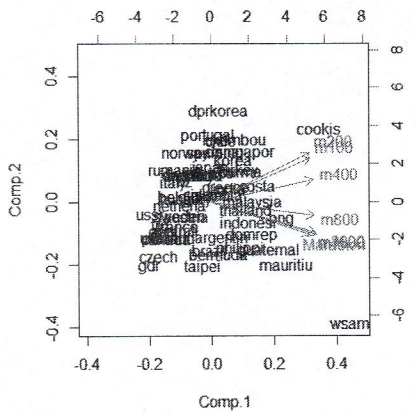





```
x11()
layout(matrix(c(1,2),1))
plot(runrec.sd[,'m100'],runrec.sd[,'Marathon'],type="n",xlab="m100",ylab="Marathon", asp=1)
text(runrec.sd[,'m100'],runrec.sd[,'Marathon'],dimnames(runrec)[[1]],cex=.7)
plot(scores.runrec[,1],scores.runrec[,2],type="n",xlab="pc1",ylab="pc2", asp=1)
text(scores.runrec[,1],scores.runrec[,2],dimnames(runrec)[[1]],cex=.7)
```




```
x11()
biplot(pc.runrec)
```




```
graphics.off()
```