# Índex

# 1 Pràctica 1

# 2 Pràctica 2

## 2.1 Introducció

Per a fer aquesta pràctica necessitem instalar i carregar els paquets `MVA` i `HSAUR2`.

```
library(HSAUR2)
library(MVA)
```

Introduim una llista, i li donem estructura de `data frame`, en aquestes dades anomenades `measure` hi ha 4 variables, tres numèriques i una categòrica per a 20 individus.

Nota que l'aventatge d'usar llistes respecte a vectors és que en llistes podem introduir variables categoriques i numèriques al mateix temps, i en vectors no.

```
measure <-
  structure(list(V1 = 1:20,
                 V2 = c(34L, 37L, 38L, 36L, 38L, 43L, 40L, 38L, 40L,
                 41L, 36L, 36L, 34L, 33L, 36L, 37L, 34L, 36L, 38L,35L),
                 V3 = c(30L, 32L, 30L, 33L, 29L, 32L, 33L, 30L, 30L, 32L,
                  24L, 25L, 24L, 22L, 26L, 26L, 25L, 26L, 28L, 23L),
                 V4 = c(32L, 37L, 36L, 39L, 33L, 38L, 42L, 40L, 37L, 39L,
                  35L, 37L, 37L, 34L, 38L, 37L, 38L, 37L, 40L, 35L)),
            .Names = c("V1", "V2", "V3","V4"),
            class = "data.frame", row.names = c(NA, -20L))
```

Que ens torna el data frame:

```
> measure
   V1 V2 V3 V4
1   1 34 30 32
2   2 37 32 37
3   3 38 30 36
4   4 36 33 39
5   5 38 29 33
6   6 43 32 38
7   7 40 33 42
8   8 38 30 40
9   9 40 30 37
10 10 41 32 39
11 11 36 24 35
12 12 36 25 37
13 13 34 24 37
14 14 33 22 34
15 15 36 26 38
16 16 37 26 37
17 17 34 25 38
18 18 36 26 37
19 19 38 28 40
20 20 35 23 35
```

Per a eliminar la primera variable i posar nom a les aldres dues ho fem així:

```
measure1<-measure[,-1]
names(measure1)<-c("chest","waist","hips")
```

i per afegir la variagle gènere:

```
measure1$gender <-gl(2,10)
levels(measure1$gender)<-c("male","female")
```

Així doncs, les dades ens queden

```
> measure1
   chest  waist  hips
1     34     30    32
2     37     32    37
3     38     30    36
4     36     33    39
5     38     29    33
6     43     32    38
7     40     33    42
8     38     30    40
9     40     30    37
10    41     32    39
11    36     24    35
12    36     25    37
13    34     24    37
14    33     22    34
15    36     26    38
16    37     26    37
17    34     25    38
18    36     26    37
19    38     28    40
20    35     23    35
```

Podem fer subconjunts d'auqestes dades amb instruccions de l'estil:

```
x=measure1[1:5,c("chest","waist")]
```

que ens torna

```
> x
  chest  waist
1    34     30
2    37     32
3    38     30
4    36     33
5    38     29
```

També ho podem fer introduint condicions i seleccionant variablems amb la comanda subset:

```
z=subset(measure1,gender=="female")[,c("chest","waist","hips")]
```

que ens torna

```
> z
   chest waist hips
11    36    24   35
12    36    25   37
13    34    24   37
14    33    22   34
15    36    26   38
16    37    26   37
17    34    25   38
18    36    26   37
19    38    28   40
20    35    23   35
```

## 2.2 Qüestió 1

*De les daedes completes (atenció, has d'eliminar la variable gènere que és categòrica), troba la matriu de covariàncies i la matriu de correlacions, usant per a aquesta darrera la comanda `cor(nom)`.*

```
y=measure1[,-4] #Eliminem el genere (dada categorica)
cor(y)
cov(y)
```

Que ens torna

```
> cov(y)
         chest     waist     hips
chest 6.631579  6.368421 3.000000
waist 6.368421 12.526316 3.578947
hips  3.000000  3.578947 5.944737
```

```
> cor(y)
          chest     waist      hips
chest 1.0000000 0.6987336 0.4778004
waist 0.6987336 1.0000000 0.4147413
hips  0.4778004 0.4147413 1.0000000
```

## 2.3 Questió 2

*Troba la matriu de correlació a partir de la matriu de covariàncies, multiplicant les matrius adequades, i comprova que obtens el mateix resultat que amb la instrucció `cor`. Per fer això, considera les comandes `diag(matriu)` i `diag(diag(matriu))`.*

Per definició, tenim que donada una matriu de covariàncies

$$S = \begin{pmatrix} s_{11} & \dots & \dots & \dots & s_{1p} \\ \vdots & \ddots & & & \vdots \\ \vdots & & s_{ii} & & \vdots \\ \vdots & \ddots & & & \vdots \\ s_{p1} & \dots & \dots & \dots & s_{pp} \end{pmatrix} \text{ tenim la matriu } D = \begin{pmatrix} s_1 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & s_p \end{pmatrix}$$

on $s_i^2 = s_{ii}$. la matriu de correlacions $R$ és

$$R = D^{-1} S D^{-1}$$

Càlculs en R:

```
S=cov(y)
D=sqrt(diag(diag(S)))
invD=solve(D)
R=invD%*%S%*%invD
```

Comprovem doncs que `R` ens dona la matriu de covariàncies:

```
> R
          [,1]      [,2]      [,3]
[1,] 1.0000000 0.6987336 0.4778004
[2,] 0.6987336 1.0000000 0.4147413
[3,] 0.4778004 0.4147413 1.0000000
```

## 2.4 Qüestió 3

*Troba les matrius de correlacions dels individus que són homes i de les dones per separat, i compara-les.*

```
measureMale=subset(measure1,gender=="male")[,-4]
measureFemale=subset(measure1,gender=="female")[,-4]
cor(measureMale)
cor(measureFemale)
```

Que ens torna les matrius de covariàncies dels homes i de les dones per separat:

```
> cor(measureMale)
          chest     waist      hips
chest 1.0000000 0.2513682 0.4976828
waist 0.2513682 1.0000000 0.6947857
hips  0.4976828 0.6947857 1.0000000
> cor(measureFemale)
          chest     waist      hips
chest 1.0000000 0.8303889 0.5885679
waist 0.8303889 1.0000000 0.9101668
hips  0.5885679 0.9101668 1.0000000
```

És directe observar que les variables estan més correlacionades entre les dones que entre els homes, especialment el `chest-waist` de les dones.

## 2.5 Qüestió 3 BIS

*Com a matriu de dades agafa ara la total eliminant els dos darrers homes i les dues darreres dones.*

```
measure2=measure1[c(-20,-19,-10,-9),c("chest","waist","hips")]
matrixMeasure2=as.matrix(measure2)
```

La matriu ens queda

```
> matrixMeasure2
  chest waist hips
1    34    30   32
2    37    32   37
3    38    30   36
4    36    33   39
5    38    29   33
6    43    32   38
7    40    33   42
8    38    30   40
11   36    24   35
12   36    25   37
13   34    24   37
14   33    22   34
15   36    26   38
16   37    26   37
17   34    25   38
18   36    26   37
```

No considero el gènere a la matriu de dades per fer els següents exercicis amb facilitat.

## 2.6 Qüestió 4

*Troba la matriu $\hat{\hat{X}}$ (matriu amb les dades estendaritzades centrades i amb variància unitat) amb la instrucció `scale`, i per altra banda fent el càldul de matrius apropiat. Comprova que dóna el mateix.*
Amb la instrucció scale ho fem així:

```
dfmat=matrixMeasure2
scale(dfmat, center=TRUE)
```

(Nota que cal posar) `center=TRUE`.
Ens torna:

```
> scale(dfmat, center=TRUE)
   chest      waist        hips
1  -1.05  0.5750840 -1.92952699
2   0.15  1.1327412  0.04947505
3   0.55  0.5750840 -0.34632536
4  -0.25  1.4115697  0.84107587
5   0.55  0.2962554 -1.53372658
6   2.55  1.1327412  0.44527546
7   1.35  1.4115697  2.02847709
8   0.55  0.5750840  1.23687628
11 -0.25 -1.0978876 -0.74212577
12 -0.25 -0.8190590  0.04947505
13 -1.05 -1.0978876  0.04947505
14 -1.45 -1.6555448 -1.13792617
15 -0.25 -0.5402304  0.44527546
16  0.15 -0.5402304  0.04947505
17 -1.05 -0.8190590  0.44527546
18 -0.25 -0.5402304  0.04947505
attr(,"scaled:center")
  chest    waist     hips
36.6250  27.9375  36.8750
attr(,"scaled:scale")
   chest     waist      hips
2.500000  3.586433  2.526526
```

Anem a fer el càlul matricialment, perimer cal recordar que si tenim una matriu de dades $X$,

$$X = \begin{pmatrix} \vec{x}_1{}^t \\ \vdots \\ \vec{x}_n{}^t \end{pmatrix},\ \overline{\vec{x}} = \begin{pmatrix} \overline{x_1}^t \\ \vdots \\ \overline{x_p}^t \end{pmatrix} = \frac{1}{n} X^t \vec{1},\ \hat{X} = \begin{pmatrix} x_{11} - \overline{x_1} & \dots & x_{1p} - \overline{x_p} \\ \vdots & \ddots & \vdots \\ x_{n1} - \overline{x_1} & \dots & x_{np} - \overline{x_p} \end{pmatrix} = X - \vec{1}(\overline{\vec{x}})^t$$

$$\hat{\hat{X}} = \begin{pmatrix} \frac{x_{11}-\overline{x_1}}{s_1} & \dots & \frac{x_{1p}-\overline{x_p}}{s_p} \\ \vdots & \ddots & \vdots \\ \frac{x_{n1}-\overline{x_1}}{s_1} & \dots & \frac{x_{np}-\overline{x_p}}{s_p} \end{pmatrix} = X - \vec{1}(\overline{\vec{x}})^t,\ S = \begin{pmatrix} s_{11} & \dots & \dots & \dots & s_{1p} \\ \vdots & \ddots & & & \vdots \\ \vdots & & s_{ii} & & \vdots \\ \vdots & \ddots & & & \vdots \\ s_{p1} & \dots & \dots & \dots & s_{pp} \end{pmatrix} = \frac{1}{n-1} \hat{X}^t X$$

```
dfmat=as.matrix(measure2)
n=20-4 #cal treure les 4 persones de les 20 totals
vec1=matrix(rep(1,n),n)
vecMitjanes=(1/n)*t(dfmat)%*%vec1
vecMitjanes
dfmatBarret=dfmat-vec1%*%t(vecMitjanes)
dfmatBarret
S=1/((n-1))*t(dfmatBarret)%*%dfmatBarret
S
invD=solve(sqrt(diag(diag(S))))
invD
Xbarretbarret=dfmatBarret%*%invD
```

I podem comprovar que `Xbarretbarret=`$\hat{\hat{X}}$.

```
> Xbarretbarret
    [,1]       [,2]        [,3]
1  -1.05  0.5750840 -1.92952699
2   0.15  1.1327412  0.04947505
3   0.55  0.5750840 -0.34632536
4  -0.25  1.4115697  0.84107587
5   0.55  0.2962554 -1.53372658
6   2.55  1.1327412  0.44527546
7   1.35  1.4115697  2.02847709
8   0.55  0.5750840  1.23687628
11 -0.25 -1.0978876 -0.74212577
12 -0.25 -0.8190590  0.04947505
13 -1.05 -1.0978876  0.04947505
14 -1.45 -1.6555448 -1.13792617
15 -0.25 -0.5402304  0.44527546
16  0.15 -0.5402304  0.04947505
17 -1.05 -0.8190590  0.44527546
18 -0.25 -0.5402304  0.04947505
```

## 2.7 Qüstió 5

*Comprova que la matriu de correlacións de les nostres dades és la matriu de covariàncies de les dades estandaritzades.*

```
cov(Xbarretbarret)
cor(dfmat)
```

```
> cov(Xbarretbarret)
          [,1]      [,2]      [,3]
[1,] 1.0000000 0.6589649 0.4353804
[2,] 0.6589649 1.0000000 0.3669486
[3,] 0.4353804 0.3669486 1.0000000
> cor(dfmat)
          chest     waist      hips
chest 1.0000000 0.6589649 0.4353804
waist 0.6589649 1.0000000 0.3669486
hips  0.4353804 0.3669486 1.0000000
```

## 2.8 Qüestió 6

*Troba la distancia euclídea entre els individus sense estandaritzar i també entre els estandaritzats.*

- *Quan valen aquestes distàncies entre els individus* 2 *i* 4*?*
- *Quina és la mitjana de les distàncies entre els individus mascles sense estandaritzar?*

```
dfq6=measure2
dist(dfq6)
dist(scale(dfq6,center=TRUE))
```

A les dues pàgines de continuació es pot veure el output de les distàncies entre els individus estandaritzades i sense estandaritzar.
Mirant a aquestes dades es pot dir que la distància entre els individus 2 i 4 sense estandaritzar és 2.449490, i estandaritzada és 0.9297189.
Si volem fer-ho de forma més elegant, ho podem fer amb R directament:

```
as.matrix(dist(dfq6))[2,4]
as.matrix(dist(scale(dfq6,center=TRUE)))[2,4]
```

que ens torna el que volem:

```
as.matrix(dist(dfq6))[2,4]
[1] 2.44949
> as.matrix(dist(scale(dfq6,center=TRUE)))[2,4]
[1] 0.9297189
```

```
> dist(dfq6)
           1         2         3         4         5         6         7         8        11
2   6.164414
3   5.656854  2.449490
4   7.874008  2.449490  4.690416
5   4.242641  5.099020  3.162278  7.483315
6  11.000000  6.082763  5.744563  7.141428  7.681146
7  12.041595  5.916080  7.000000  5.000000 10.049876  5.099020
8   8.944272  3.741657  4.000000  3.741657  7.071068  5.744563  4.123106
11  7.000000  8.306624  6.403124  9.848858  5.744563 11.045361 12.083046  8.062258
12  7.348469  7.071068  5.477226  8.246211  6.000000  9.949874 10.246951  6.164414  2.236068
13  7.810250  8.544004  7.280110  9.433981  7.549834 12.083046 11.916375  7.810250  2.828427
14  8.306624 11.180340  9.643651 12.449900  8.660254 14.696938 15.297059 11.180340  3.741657
15  7.483315  6.164414  4.898979  7.071068  6.164414  9.219544  9.000000  4.898979  3.605551
16  7.071068  6.000000  4.242641  7.348469  5.099020  8.544004  9.110434  5.099020  3.000000
17  7.810250  7.681146  6.708204  8.306624  7.549834 11.401754 10.770330  6.708204  3.741657
18  6.708204  6.082763  4.582576  7.280110  5.385165  9.273618  9.486833  5.385165  2.828427
          12        13        14        15        16        17
2
3
4
5
6
7
8
11
12
13  2.236068
14  5.196152  3.741657
15  1.414214  3.000000  6.403124
16  1.414214  3.605551  6.403124  1.414214
17  2.236068  1.414214  5.099020  2.236068  3.316625
18  1.000000  2.828427  5.830952  1.000000  1.000000  2.449490
```

```
> dist(scale(dfq6, center=TRUE))
           1         2         3         4         5         6         7         8        11
2  2.3806366
3  2.2508948 0.7922370
4  3.0026569 0.9297189 1.6582009
5  1.6716469 1.8347305 1.2196996 2.7429205
6  4.3486398 2.4324181 2.2220741 2.8415495 2.9353292
7  4.7037756 2.3311359 2.6418545 1.9924662 3.8174889 2.0060590
8  3.5476908 1.3714602 1.5832016 1.2232606 2.7845979 2.2220741 1.4022626
11 2.2019890 2.4004867 1.8961782 2.9671373 1.7917216 3.7716875 4.0661550 2.7120625
12 2.5495262 1.9923664 1.6553829 2.3669255 2.0953409 3.4360125 3.3841031 1.9983884 0.8392718
13 2.5913863 2.5329241 2.3485084 2.7502742 2.6476711 4.2535118 3.9967268 2.6016832 1.1254474
14 2.4004867 3.4270191 3.0987637 3.8423483 2.8224425 5.1265062 5.2223846 3.8230081 1.3811732
15 2.7429205 1.7650756 1.5844740 1.9915275 2.2926311 3.2617225 2.9792703 1.5844740 1.3118320
16 2.5691195 1.6729716 1.2492334 2.1438647 1.8347305 2.9522012 3.0275358 1.6774528 1.0476705
17 2.7537831 2.3251197 2.2650092 2.4025741 2.7785563 4.0950609 3.6389878 2.2650092 1.4586525
18 2.4083968 1.7201261 1.4284901 2.1062183 1.9611823 3.2856494 3.2071752 1.8148961 0.9683044
          12        13        14        15        16        17
2
3
4
5
6
7
8
11
12
13 0.8471986
14 1.8840462 1.3714602
15 0.4841522 1.0524445 2.2782567
16 0.4875914 1.3232466 2.2833852 0.5627237
17 0.8925570 0.4841522 1.8347305 0.8471986 1.2939874
18 0.2788286 0.9751828 2.0233259 0.3958004 0.4000000 0.9350954
```

Finalment ens demanen calcular la mitjana de les distàncies entre els individus mascles sense estandaritzar,

```
datahomes=subset(measure1,gender=="male")[,-4]
dim=dim(as.matrix(dist(datahomes)))[1]
sum(as.matrix(dist(datahomes)))/(dim*(dim-1))
```

Ens torna

```
> sum(as.matrix(dist(datahomes)))/(dim*(dim-1))
[1] 5.55992
```

Per tant, la mitjana de les ditàncies entre els individus mascles sense estandaritzar és 5.55992.

# Codi complert:

```
library(HSAUR2)
library(MVA)
#en llista podem barrejar dades categoriques amb numeriques, en vectors no
measure <-
  structure(list(V1 = 1:20,
                 V2 = c(34L, 37L, 38L, 36L, 38L, 43L, 40L, 38L, 40L, 41L,
                  36L, 36L, 34L, 33L, 36L, 37L, 34L, 36L, 38L,35L),
                 V3 = c(30L, 32L, 30L, 33L, 29L, 32L, 33L, 30L, 30L, 32L,
                  24L, 25L, 24L, 22L, 26L, 26L, 25L, 26L, 28L, 23L),
                 V4 = c(32L, 37L, 36L, 39L, 33L, 38L, 42L, 40L, 37L, 39L,
                  35L, 37L, 37L, 34L, 38L, 37L, 38L, 37L, 40L, 35L)),
            .Names = c("V1", "V2", "V3","V4"),
            class = "data.frame", row.names = c(NA, -20L))
measure
measure1<-measure[,-1]
measure1
names(measure1)<-c("chest","waist","hips")
measure1

measure1$gender <-gl(2,10)
measure1
levels(measure1$gender)<-c("male","female")
measure1

x=measure1[1:5,c("chest","waist")]
x
subset(measure1,gender=="female")
z=subset(measure1,gender=="female")[,c("chest","waist","hips")]
z

#QUESTIO 1
measure1
measure1[,-4]
y=measure1[,-4]
cor(y)
cov(y)
#QUESTIO 2
S=cov(y)
D=sqrt(diag(diag(S)))
invD=solve(D)
R=invD%*%S%*%invD
R
cor(y)
#QUESTIO 3
measureMale=subset(measure1,gender=="male")[,-4]
```

```
measureFemale=subset(measure1,gender=="female")[,-4]
cor(measureMale)
cor(measureFemale)
#------------------------------------------------
#treiem 2 Ultims homes i 2 dones
measure2=measure1[c(-20,-19,-10,-9),c("chest","waist","hips")]

#QUESTIO 4
dfmat=as.matrix(measure2)
n=20-4 #cal treure les 4 persones de les 20 totals
vec1=matrix(rep(1,n),n)
vecMitjanes=(1/n)*t(dfmat)%*%vec1
vecMitjanes
dfmatBarret=dfmat-vec1%*%t(vecMitjanes)
dfmatBarret
S=1/((n-1))*t(dfmatBarret)%*%dfmatBarret
S

invD=solve(sqrt(diag(diag(S))))
DS
Xbarretbarret=dfmatBarret%*%invD
Xbarretbarret

#donen el mateix
Xbarretbarret
scale(dfmat, center=TRUE)
#si posem false treu el producte escalar

#QUESTIO 5
cov(Xbarretbarret)
cor(dfmat)

#QUESTIO 6
dfq6=measure2
dist(dfq6)
dist(scale(dfq6,center=TRUE))

as.matrix(dist(dfq6))[2,4]
as.matrix(dist(scale(dfq6,center=TRUE)))[2,4]


#QUESTIO 6 b)

datahomes=subset(measure1,gender=="male")[,-4]
dim=dim(as.matrix(dist(datahomes)))[1]
sum(as.matrix(dist(datahomes)))/(dim*(dim-1))
```

# 3 Pràctica 3

## 3.1 Introducció

En aquesta pràctica usarem els paquets `MVA` i `HSAUR` :

```
library(HSAUR2)
library(MVA)
```

Usarem les dades `pottery`

```
pottery
```

```
   Al2O3 Fe2O3  MgO  CaO Na2O  K2O TiO2   MnO   BaO kiln
1   18.8  9.52 2.00 0.79 0.40 3.20 1.01 0.077 0.015    1
2   16.9  7.33 1.65 0.84 0.40 3.05 0.99 0.067 0.018    1
3   18.2  7.64 1.82 0.77 0.40 3.07 0.98 0.087 0.014    1
4   16.9  7.29 1.56 0.76 0.40 3.05 1.00 0.063 0.019    1
5   17.8  7.24 1.83 0.92 0.43 3.12 0.93 0.061 0.019    1
6   18.8  7.45 2.06 0.87 0.25 3.26 0.98 0.072 0.017    1
7   16.5  7.05 1.81 1.73 0.33 3.20 0.95 0.066 0.019    1
8   18.0  7.42 2.06 1.00 0.28 3.37 0.96 0.072 0.017    1
9   15.8  7.15 1.62 0.71 0.38 3.25 0.93 0.062 0.017    1
10  14.6  6.87 1.67 0.76 0.33 3.06 0.91 0.055 0.012    1
11  13.7  5.83 1.50 0.66 0.13 2.25 0.75 0.034 0.012    1
12  14.6  6.76 1.63 1.48 0.20 3.02 0.87 0.055 0.016    1
13  14.8  7.07 1.62 1.44 0.24 3.03 0.86 0.080 0.016    1
14  17.1  7.79 1.99 0.83 0.46 3.13 0.93 0.090 0.020    1
15  16.8  7.86 1.86 0.84 0.46 2.93 0.94 0.094 0.020    1
16  15.8  7.65 1.94 0.81 0.83 3.33 0.96 0.112 0.019    1
17  18.6  7.85 2.33 0.87 0.38 3.17 0.98 0.081 0.018    1
18  16.9  7.87 1.83 1.31 0.53 3.09 0.95 0.092 0.023    1
19  18.9  7.58 2.05 0.83 0.13 3.29 0.98 0.072 0.015    1
20  18.0  7.50 1.94 0.69 0.12 3.14 0.93 0.035 0.017    1
21  17.8  7.28 1.92 0.81 0.18 3.15 0.90 0.067 0.017    1
22  14.4  7.00 4.30 0.15 0.51 4.25 0.79 0.160 0.019    2
23  13.8  7.08 3.43 0.12 0.17 4.14 0.77 0.144 0.020    2
24  14.6  7.09 3.88 0.13 0.20 4.36 0.81 0.124 0.019    2
25  11.5  6.37 5.64 0.16 0.14 3.89 0.69 0.087 0.009    2
26  13.8  7.06 5.34 0.20 0.20 4.31 0.71 0.101 0.021    2
27  10.9  6.26 3.47 0.17 0.22 3.40 0.66 0.109 0.010    2
28  10.1  4.26 4.26 0.20 0.18 3.32 0.59 0.149 0.017    2
29  11.6  5.78 5.91 0.18 0.16 3.70 0.65 0.082 0.015    2
30  11.1  5.49 4.52 0.29 0.30 4.03 0.63 0.080 0.016    2
31  13.4  6.92 7.23 0.28 0.20 4.54 0.69 0.163 0.017    2
32  12.4  6.13 5.69 0.22 0.54 4.65 0.70 0.159 0.015    2
33  13.1  6.64 5.51 0.31 0.24 4.89 0.72 0.094 0.017    2
34  11.6  5.39 3.77 0.29 0.06 4.51 0.56 0.110 0.015    3
35  11.8  5.44 3.94 0.30 0.04 4.64 0.59 0.085 0.013    3
```

```
36   18.3   1.28  0.67  0.03  0.03  1.96  0.65  0.001  0.014     4
37   15.8   2.39  0.63  0.01  0.04  1.94  1.29  0.001  0.014     4
38   18.0   1.50  0.67  0.01  0.06  2.11  0.92  0.001  0.016     4
39   18.0   1.88  0.68  0.01  0.04  2.00  1.11  0.006  0.022     4
40   20.8   1.51  0.72  0.07  0.10  2.37  1.26  0.002  0.016     4
41   17.7   1.12  0.56  0.06  0.06  2.06  0.79  0.001  0.013     5
42   18.3   1.14  0.67  0.06  0.05  2.11  0.89  0.006  0.019     5
43   16.7   0.92  0.53  0.01  0.05  1.76  0.91  0.004  0.013     5
44   14.8   2.74  0.67  0.03  0.05  2.15  1.34  0.003  0.015     5
45   19.1   1.64  0.60  0.10  0.03  1.75  1.04  0.007  0.018     5
```

De les dades de l'arxiu `pottery` usarem en partiuclar les 3 primeres variables, la cinquena i sisena. De les unitats experimentals tria els 10 primers individus del forn 1, els 4 priemrs del forn 2, més tots els que venen del forn 4.

```
#Borrem els noms i posem numeros
measure1<-pottery
names(measure1)<-c(seq(1,9,1),"forn")
measure1
#10 primeres del forn 1
frameA<-subset(measure1,forn==1)[seq(1,10,1),c(1,2,3,5,6)]
#4 primeres del forn 2
frameB<-subset(measure1,forn==2)[seq(1,4,1),c(1,2,3,5,6)]
#tots els que venen del forn 4
frameC<-subset(measure1,forn==4)[,c(1,2,3,5,6)]
dataframe<-rbind(frameA,frameB,frameC)  #ajuntem
dataframe
```

```
      1     2     3     5     6
1   18.8  9.52  2.00  0.40  3.20
2   16.9  7.33  1.65  0.40  3.05
3   18.2  7.64  1.82  0.40  3.07
4   16.9  7.29  1.56  0.40  3.05
5   17.8  7.24  1.83  0.43  3.12
6   18.8  7.45  2.06  0.25  3.26
7   16.5  7.05  1.81  0.33  3.20
8   18.0  7.42  2.06  0.28  3.37
9   15.8  7.15  1.62  0.38  3.25
10  14.6  6.87  1.67  0.33  3.06
22  14.4  7.00  4.30  0.51  4.25
23  13.8  7.08  3.43  0.17  4.14
24  14.6  7.09  3.88  0.20  4.36
25  11.5  6.37  5.64  0.14  3.89
36  18.3  1.28  0.67  0.03  1.96
37  15.8  2.39  0.63  0.04  1.94
38  18.0  1.50  0.67  0.06  2.11
39  18.0  1.88  0.68  0.04  2.00
40  20.8  1.51  0.72  0.10  2.37
```

## 3.2 Qüestió 1

Troba les matrius de covariància i correlació mostral d'aquest subconjunt de dades. (nota que has tret la variable categòrica!)

```
cov(dataframe)
cor(dataframe)
```

```
> cov(dataframe)
            1          2           3           5           6
1  4.88988304 -1.6651696 -2.29146491 -0.02513743 -1.00663158
2 -1.66516959  6.8379374  1.85845088  0.33478684  1.48822310
3 -2.29146491  1.8584509  1.86458947  0.05671901  0.89993392
5 -0.02513743  0.3347868  0.05671901  0.02420936  0.06234678
6 -1.00663158  1.4882231  0.89993392  0.06234678  0.56275614
> cor(dataframe)
            1          2          3           5          6
1  1.00000000 -0.2879696 -0.7588787 -0.07306005 -0.6068218
2 -0.28796962  1.0000000  0.5204714  0.82283773  0.7586568
3 -0.75887870  0.5204714  1.0000000  0.26695949  0.8785346
5 -0.07306005  0.8228377  0.2669595  1.00000000  0.5341489
6 -0.60682183  0.7586568  0.8785346  0.53414886  1.0000000
```

Estandaritza amb la instrucció `scale` aquestes dades. Quin és el valor de l'individu 5 escalat?

```
scale(dataframe)
scale(dataframe)[5,]
```

```
> scale(dataframe)[5,]
         1          2           3          5          6
0.49268292 0.53337138 -0.15147722 1.10950335 0.04420046
```

## 3.3 Qüestió 2

A quina distància estan els individus 2 i 4? (recorda que per trobar la distància euclídea entre els individus pots utilitzar la comanda `dist`).

```
Distancia=as.matrix(dist(dataframe))
Distancia[2,4]
```

```
> Distancia[2,4]
[1] 0.09848858
```

Quina és la distància entre els individus sense estandaritzar que venen del forn 4?

```
framef4<-subset(measure1, forn==4)[]
framef4
dist(framef4)
Matdist=as.matrix(dist(framef4))
dim=dim(as.matrix(dist(framef4)))[1]
sum(Matdist)/(dim*(dim-1))
```

```
> sum(Matdist)/(dim*(dim-1))
[1] 2.262614
```

## 3.4 Qüestió 3

Troba ara, operant matrius, la distància de Mahalanobis entre l'individu 2 i el 4 (la inversa d'una matriu s'obté aplicant `solve()`).

```
S=as.matrix(cov(dataframe))
dfmat=as.matrix(dataframe)
invS=solve(S)
dfmat[2,]%*%invS%*%dfmat[4,]
v=dfmat[2,]-dfmat[4,]
distMaha24=sqrt(t(v)%*%invS%*%v)
distMaha24
```

```
> distMaha24
          [,1]
[1,] 0.1951034
```

## 3.5 Qüestió 4

Per a iterar per a totes les files utilitza la comanda `apply` amb `(MARGIN=1)`. Si poses `MARGIN=2`, operarà per a columnes en lloc de per files, que és el que fa la comanda `sapply()`.
Troba la matriu de distàncies de Mahalanobis al quadrat, entre els individus del teu estudi i el vector de mitjanes. Crea doncs la funció adequada emprant la comanda `apply`.

M'ha semblat més sensill fer-ho palicant for() :

```
nfiles=dim(dfmat)[1]
MatDistMahaMeans=matrix(0,nfiles,1)
CM=colMeans(dataframe)
for (i in 1:nfiles){
    v=dfmat[i,]-CM
    distMaha=t(v)%*%invS%*%v
    MatDistMahaMeans[i,1]=distMaha
}
MatDistMahaMeans
```

```
> MatDistMahaMeans
            [,1]
 [1,]  6.4372083
 [2,]  1.4265886
 [3,]  1.9741851
 [4,]  1.5009253
 [5,]  1.9559799
 [6,]  3.9873669
 [7,]  0.7266228
 [8,]  1.9814315
 [9,]  2.8511866
[10,]  4.2643321
[11,] 11.9596914
[12,]  7.1131258
[13,]  6.7681577
[14,] 14.6641566
[15,]  3.2554124
[16,]  5.6556403
[17,]  2.8238443
[18,]  2.7122971
[19,]  7.9418471
```

Si volem trobar la matriu de distàncies de Mahalanobis ho podem fer així:

```
nfiles=dim(dfmat)[1]
MatDistMaha=matrix(1,nfiles ,nfiles)

for (i in 1:nfiles){
  for (j in 1:nfiles){
    v=dfmat[i,]-dfmat[j,]
    distMaha=sqrt(t(v)%*%invS%*%v)
    MatDistMaha[i,j]=distMaha
  }
}
MatDistMaha
```

```
El output es molt gran
```

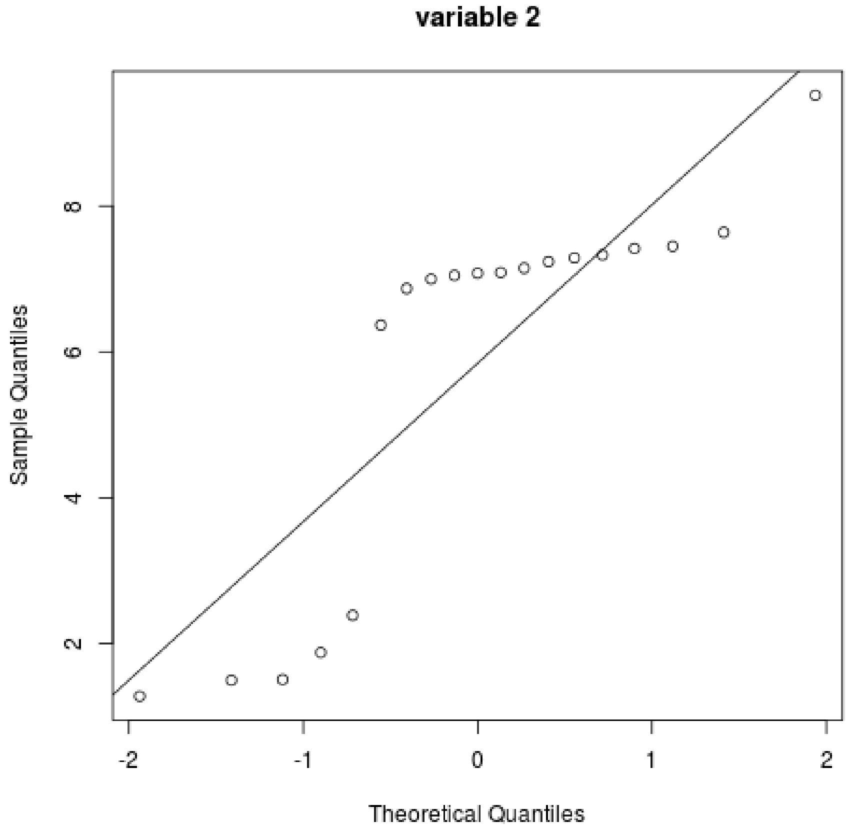
### 3.6 Qüestió 5

Per a veure gràficament si el mecanisme aleatori pot ser el d'una llei determinada podem fer un Q-Q plot. Per a recordar el què son aquets gràfics pots veure l'excel·lent explicació que trobaràs a Wikipèdia:

enllaç

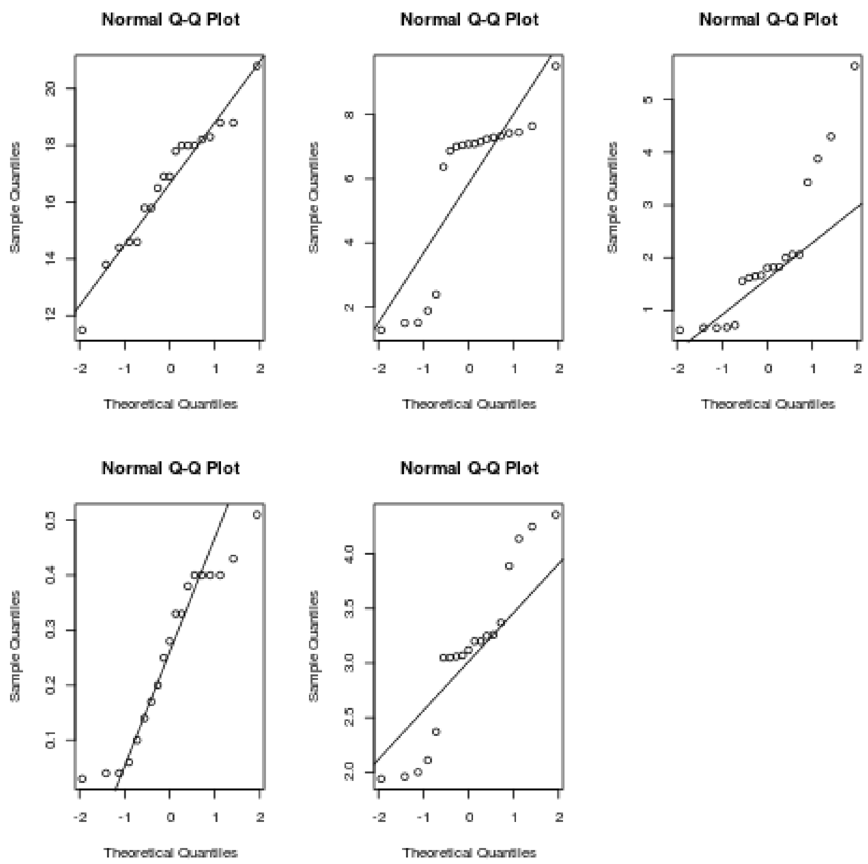
Amb la instrucció `qqnorm` i `qqline` crea un Q-Q plot de la primera variable del subframe de les dades `potteri` que has creat per a veure si poden venir d'una normal. (consulta amb `?qqnorm` l'ajuda que necessitis). Per una altra banda, a la pàgina 19 del llibre *d'Everit i Horton*, o al paquet `MVA`, hi trobaràs un exemple de codis per a fer això.)

```
par(mfrow=c(1,1))
qqnorm(dataframe[,1],main="variable 1"); qqline(dataframe[,1])
qqnorm(dataframe[,2],main="variable 2"); qqline(dataframe[,2])
```




Podries fer els gràfics `QQ` de totes les variables del subframe de manera simultània amb la instrucció `layout(matrix(,nc=`, utilitzant la instrucció `sapply`. Mira l'exemple de la pàgina 20 del llibre citat, i fer-ho.

```
QQPLOT <-function(x){
  qqnorm(x)
  qqline(x)
}
par(mfrow=c(2,3))
sapply(dataframe,QQPLOT)
```




També es pot fer en lloc d'usar apply, usar la comanda més general `for`:

```
dim(dataframe[2])[1]
par(mfrow=c(2,3))
for(i in 1:dim(dataframe[2])[1]){
  qqnorm(dataframe[,i],main="variable i"); qqline(dataframe[,i])
}
```

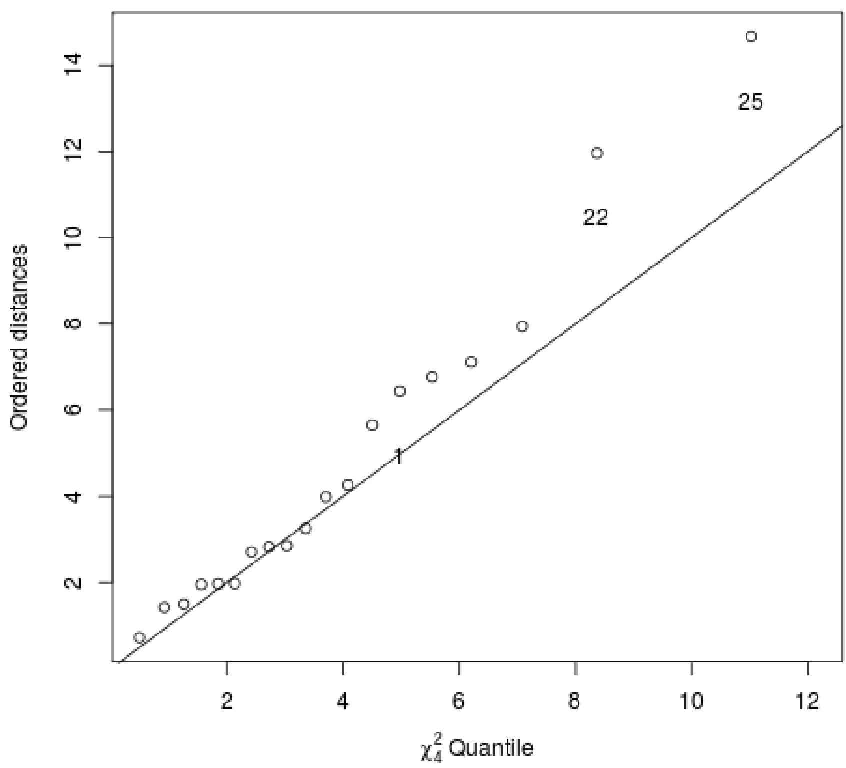
El resultat és el mateix.

### 3.7 Qüestió 6

En el cas que les dades vinguin d'una llein normal multivariant, els quadrats de les distàncies de Mahalanobis entre els individus i el vector de mitjanes tenen aproximadament una distribució $\chi^2$ amb $n-1$ graus de llibertat, on $n$ és el nombre de variables que utilitzis.

Amb la instrucció `plot` genera el `Q-Q` plot dels quadrats de les distàncies de Mahalanobs entre els individus i el vector de mitjanes pel subframe que has creat en la primera qüestió, per veure si segueixen la llei $\chi^2$ (recorda que treballes amb 5 variables, i que els quantils de la $\chi^2$) els trobes amb la comanda `(qchisq)`. A la pàgina 20 del llibre *d'Everitt i Horthorn*, o al paquet `MVA` hi trobaràs un exemple de codis per a fer això.

```
png(file = "P3P33.png") #exporta al fitxer indicat el plot

library(HSAUR2)
library(MVA)
ngrausllibertat=dim(dataframe)[2]-1 #nombre de variables menys 1
x <- dataframe
cm <- colMeans(x)
S <- cov(x)
d <- apply(x, 1, function(x) t(x - cm) %*% solve(S) %*% (x - cm))
plot(qc <- qchisq((1:nrow(x) - 1/2) / nrow(x), df = ngrausllibertat),
     sd <- sort(d),
     xlab = expression(paste(chi[4]^2, " Quantile")),
     ylab = "Ordered distances", xlim = range(qc) * c(1, 1.1))
oups <- which(rank(abs(qc - sd), ties = "random") > nrow(x) - 3)
text(qc[oups], sd[oups] - 1.5, names(oups))
abline(a = 0, b = 1)
dev.off() #plots off
```




Aquest plot ens permet detectar que 22 i 25 són outliers.

## 3.8 Codi complert

```
\newpage
\begin{lstlisting}[frame=single]
# margin =1 ho fem per file ,s margin=2 ho fem per columnes
# qqplot si sacosta a una recta la distribucio que proposem es bona4
# instruccions final capitol 1 eveerit
# avans 21 octubre
wd <- getwd()
print(paste0("Current working dir: ", wd))
rm(list=ls())
dev.off()
library(HSAUR2)
library(MVA)
pottery
#data(pottery,package="HSAUR2")

#3 primeres variables i la 5a i 6a

#Borrem els noms i posem numeros
measure1<-pottery
names(measure1)<-c(seq(1,9,1),"forn")
measure1
#10 primeres del forn 1
frameA<-subset(measure1,forn==1)[seq(1,10,1),c(1,2,3,5,6)]
#4 primeres del forn 2
frameB<-subset(measure1,forn==2)[seq(1,4,1),c(1,2,3,5,6)]
#tots els que venen del forn 4
frameC<-subset(measure1,forn==4)[,c(1,2,3,5,6)]

dataframe<-rbind(frameA,frameB,frameC)  #ajuntem
dataframe

#QUESTIO 1
cov(dataframe)
cor(dataframe)

scale(dataframe)
scale(dataframe)[5,]

#QUESTIO 2
Distancia=as.matrix(dist(dataframe))
Distancia[2,4]

framef4<-subset(measure1,forn==4)[]
framef4
dist(framef4)
Matdist=as.matrix(dist(framef4))
```

```
dim=dim(as.matrix(dist(framef4)))[1]
sum(Matdist)/(dim*(dim-1))

#QUESTIO 3
#mean<-colMeans(dataframe) <- potser borra?
dataframe
S=as.matrix(cov(dataframe))

dfmat=as.matrix(dataframe)
dfmat
invS=solve(S)
dfmat[2,]%*%invS%*%dfmat[4,]
v=dfmat[2,]-dfmat[4,]
distMaha24=sqrt(t(v)%*%invS%*%v)
distMaha24

#Per que usar apply si pots usar for ?
nfiles=dim(dfmat)[1]
MatDistMaha=matrix(1,nfiles,nfiles)

for (i in 1:nfiles){
  for (j in 1:nfiles){
    v=dfmat[i,]-dfmat[j,]
    distMaha=sqrt(t(v)%*%invS%*%v)
    MatDistMaha[i,j]=distMaha
  }
}
MatDistMaha

#intnant entendre que vol que fagi amb apply:
dfmat
apply(dfmat, MARGIN=1,sum )
apply(dfmat, MARGIN=1, df )
dfmat[,1]
dfmat
distMaha24
dfmat[4,]
dfmat
mahalanobis(dataframe)

#QUESTIO 4
nfiles=dim(dfmat)[1]
MatDistMahaMeans=matrix(0,nfiles,1)
CM=colMeans(dataframe)
for (i in 1:nfiles){
    v=dfmat[i,]-CM
    distMaha=t(v)%*%invS%*%v
```

```
    MatDistMahaMeans[i,1]=distMaha
}
MatDistMahaMeans
#no ho ser fer amb apply
x=dataframe;
apply(x, MARGIN=1, sum)
apply(x, MARGIN=2, sum)
sapply(x,sum)

apply(x,MARGIN=1,t(x)%*%invS%*%x)

#QUESTIO 5
par(mfrow=c(1,1))
qqnorm(dataframe[,1],main="variable 1"); qqline(dataframe[,1])
qqnorm(dataframe[,2],main="variable 2"); qqline(dataframe[,2])
dim(dataframe[2])[1]
par(mfrow=c(2,3))
for(i in 1:dim(dataframe[2])[1]){
  qqnorm(dataframe[,i],main="variable i"); qqline(dataframe[,i])
}

#no es millor fer "for" que sapply? <- Depen dels gustos
QQPLOT <-function(x){
  qqnorm(x)
  qqline(x)
}
par(mfrow=c(2,3))
sapply(dataframe,QQPLOT)
#--------------------------------------EVERIT
matrix(1:8,nc=3)
layout(matrix(1:8,nc=2))
sapply(colnames(USairpollution),function(x){
  qqnorm(USairpollution[[x]],main=x)
  qqline(USairpollution[[x]])
})
#QUESTIO 6
  png(file = "P3P333.png") #exporta al fitxer indicat el plot

  library(HSAUR2)
  library(MVA)
  ngrausllibertat=dim(dataframe)[2]-1 #nombre de variables menys 1
  x <- dataframe
  cm <- colMeans(x)
  S <- cov(x)
  d <- apply(x, 1, function(x) t(x - cm) %*% solve(S) %*% (x - cm))
  plot(qc <- qchisq((1:nrow(x) - 1/2) / nrow(x), df = ngrausllibertat),
       sd <- sort(d),
```

```
        xlab = expression(paste(chi[4]^2, " Quantile")),
        ylab = "Ordered distances", xlim = range(qc) * c(1, 1.1))
  oups <- which(rank(abs(qc - sd), ties = "random") > nrow(x) - 3)
  text(qc[oups], sd[oups] - 1.5, names(oups))
  abline(a = 0, b = 1)
  dev.off()
##EXEMPLE EVERIT
USairpollution
rm(list=ls())
dev.off()
png(file = "myplot.png")


x <- USairpollution
cm <- colMeans(x)
S <- cov(x)
d <- apply(x, 1, function(x) t(x - cm) %*% solve(S) %*% (x - cm))
plot(qc <- qchisq((1:nrow(x) - 1/2) / nrow(x), df = 6),
     sd <- sort(d),
     xlab = expression(paste(chi[6]^2, " Quantile")),
     ylab = "Ordered distances", xlim = range(qc) * c(1, 1.1))
oups <- which(rank(abs(qc - sd), ties = "random") > nrow(x) - 3)
text(qc[oups], sd[oups] - 1.5, names(oups))
abline(a = 0, b = 1)
dev.off()
dataframe
x

dim(USairpollution)
dataframe
```

# 4 Pràctica 4

# 5 Pràctica 5

## 5.1 Introducció

L'objectiu d'aquesta pràctica és començar a aplicar les tècniques de **components principals** a les dades anomenades `USairpollution`, amb les quals ja havíem treballat en sessions anteriors.
Aquestes dades estan en el paquet `HSAUR2`.

Ens basarem en el capítol 3 del llibre d'Everitt i Hothorn. Pots trobar les comandes dels exemples que ell proposa a `Ch-PCA` en el paquet `MVA`.

Per aplicar components principals usarem la instrucció

```
nom del resultat=princomp(covmat=nom de la matriu de covariancies o correlacions
```

Fixa't que has de decidir si surts de la matriu de covariàncies o de la de correlacions, ja que els resultats són completament diferents. En general sortim de la matriu de correlacions si les variables estan en unitats de mesura diferents.

També pots sortir de les dades directament. Si aquestes estan escalades i centrades recorda que la matriu de covariàncies serà la de correlacions.

```
nom del resultat=princomp(x=nom de les dades)
```

En aquest darrer cas, sortirà de la matriu de covariàncies. Si vols sortir de la matriu de correlacions, cal afegir la comanda `cor=TRUE`.

Podem veure un primer resúm de la sortida amb la comanda `summary(nom)`, i si volem que al fer això ens mostri els vectors propis (matriu V) cal afegir `loadings=TRUE`.

Veiem doncs les dades en que treballarem:

```
library(MVA)
library(HSAUR2)

USairpollution
```

```
> USairpollution
               SO2 temp manu popul wind precip predays
Albany          46 47.6   44   116  8.8  33.36     135
Albuquerque     11 56.8   46   244  8.9   7.77      58
Atlanta         24 61.5  368   497  9.1  48.34     115
Baltimore       47 55.0  625   905  9.6  41.31     111
Buffalo         11 47.1  391   463 12.4  36.11     166
Charleston      31 55.2   35    71  6.5  40.75     148
Chicago        110 50.6 3344  3369 10.4  34.44     122
Cincinnati      23 54.0  462   453  7.1  39.04     132
Cleveland       65 49.7 1007   751 10.9  34.99     155
Columbus        26 51.5  266   540  8.6  37.01     134
Dallas           9 66.2  641   844 10.9  35.94      78
Denver          17 51.9  454   515  9.0  12.95      86
Des Moines      17 49.0  104   201 11.2  30.85     103
Detroit         35 49.9 1064  1513 10.1  30.96     129
Hartford        56 49.1  412   158  9.0  43.37     127
Houston         10 68.9  721  1233 10.8  48.19     103
Indianapolis    28 52.3  361   746  9.7  38.74     121
Jacksonville    14 68.4  136   529  8.8  54.47     116
Kansas City     14 54.5  381   507 10.0  37.00      99
Little Rock     13 61.0   91   132  8.2  48.52     100
Louisville      30 55.6  291   593  8.3  43.11     123
Memphis         10 61.6  337   624  9.2  49.10     105
Miami           10 75.5  207   335  9.0  59.80     128
Milwaukee       16 45.7  569   717 11.8  29.07     123
Minneapolis     29 43.5  699   744 10.6  25.94     137
Nashville       18 59.4  275   448  7.9  46.00     119
New Orleans      9 68.3  204   361  8.4  56.77     113
Norfolk         31 59.3   96   308 10.6  44.68     116
Omaha           14 51.5  181   347 10.9  30.18      98
Philadelphia    69 54.6 1692  1950  9.6  39.93     115
Phoenix         10 70.3  213   582  6.0   7.05      36
Pittsburgh      61 50.4  347   520  9.4  36.22     147
Providence      94 50.0  343   179 10.6  42.75     125
Richmond        26 57.8  197   299  7.6  42.59     115
Salt Lake City  28 51.0  137   176  8.7  15.17      89
San Francisco   12 56.7  453   716  8.7  20.66      67
Seattle         29 51.1  379   531  9.4  38.79     164
St. Louis       56 55.9  775   622  9.5  35.89     105
Washington      29 57.3  434   757  9.3  38.89     111
Wichita          8 56.6  125   277 12.7  30.58      82
Wilmington      36 54.0   80    80  9.0  40.25     114
```

## 5.2 Qüestó 1

**Elimina la variable SO2, i troba la matriu de covariàncies i la de correlacions de les 6 variables que queden**

```
dades=USairpollution[-1]
#si treballem en components principals hem de treballar amb dades centrades:
data=scale(dades, center=TRUE, scale=FALSE)

CVm=cov(data)
CRm=cor(data
```

```
> CVm
             temp         manu         popul         wind        precip     predays
temp    52.239878    -773.9713    -262.3496   -3.6113537    32.8629884   -82.42616
manu  -773.971341  317502.8902  311718.8140  191.5481098  -215.0199024  1968.95976
popul -262.349634  311718.8140  335371.8939  175.9300610  -178.0528902   645.98598
wind    -3.611354     191.5481     175.9301    2.0410244    -0.2185311     6.21439
precip  32.862988    -215.0199    -178.0529   -0.2185311   138.5693840   154.79290
predays -82.426159   1968.9598     645.9860    6.2143902   154.7929024   702.59024
> CRm
              temp         manu        popul         wind       precip     predays
temp    1.00000000  -0.19004216  -0.06267813  -0.34973963   0.38625342  -0.43024212
manu   -0.19004216   1.00000000   0.95526935   0.23794683  -0.03241688   0.13182930
popul  -0.06267813   0.95526935   1.00000000   0.21264375  -0.02611873   0.04208319
wind   -0.34973963   0.23794683   0.21264375   1.00000000  -0.01299438   0.16410559
precip  0.38625342  -0.03241688  -0.02611873  -0.01299438   1.00000000   0.49609671
predays -0.43024212   0.13182930   0.04208319   0.16410559   0.49609671   1.00000000
```

**Quines són les variàncies de les variables originals?**
Són els elements de la diagonal de la matriu de covariàncies:

```
VARm=diag(CVm)
VARm
```

```
> VARm
        temp         manu        popul         wind       precip      predays
5.223988e+01 3.175029e+05 3.353719e+05 2.041024e+00 1.385694e+02 7.025902e+02
```

## 5.3 Qüestó 2

**Calcula les components principals sortint de la matriu de covariàncies i de la matriu de correlacions. Dóna el mateix? Comenta la resposta.**

Ho podem fer de formes diferents

```
PC1=princomp(x=data,cor=FALSE)
PC2=princomp(x=data,cor=TRUE)
PC11=princomp(covmat=CVm) #perque PC1 es diferent a PC11???
PC22=princomp(covmat=CRm)
```

`PC1` i `PC11` són equivalents, de la mateixa manera, `PC2` i `PC22` són equivalents.
En general no és el mateix sortir de la matriu de covariàncies que la de correlacions, només és equivalent cuan les dades són escalades (centrades i escalades).

```
> PC1
Call:
princomp(x = data, cor = FALSE)

Standard deviations:
    Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
789.127681 119.619735  25.756133  10.768802   3.511540   1.246716
 6  variables and  41 observations.
> PC11
Call:
princomp(covmat = CVm)

Standard deviations:
    Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
798.930886 121.105752  26.076097  10.902581   3.555163   1.262204
 6  variables and  NA observations.
> PC2
Call:
princomp(x = data, cor = TRUE)

Standard deviations:
   Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6
1.4819456 1.2247218 1.1809526 0.8719099 0.3384829 0.1855998
 6  variables and  41 observations.
> PC22
Call:
princomp(covmat = CRm)

Standard deviations:
   Comp.1    Comp.2    Comp.3    Comp.4    Comp.5    Comp.6
1.4819456 1.2247218 1.1809526 0.8719099 0.3384829 0.1855998
 6  variables and  NA observations.
```

Per al nostre estudi és més apropiat treballar amb la matriu de correlacions, ja que les variables amb les que treballem tenen valors en unitats de mesura diferents.

## 5.4 Qüestó 3

**Si surts de la matriu de correlacions, quines seran les variàncies de les components principals? Quan sumaran?**

La suma de les variàncies de les components principals és el nombre de components principals que tenim (i.e. el nombre de vairables $p$), en el nostre cas $p = 6$.

```
PC2=princomp(covmat=CRm)
summary(PC2)
SDm=summary(PC2)$sd
(VARm=SDm^2) #elevem element a element al cuardat (sd^2=VAR)
sum(VARm)
```

El summary és

```
> summary(PC2)
Importance of components:
                          Comp.1    Comp.2    Comp.3    Comp.4     Comp.5
Comp.6
Standard deviation     1.4819456 1.2247218 1.1809526 0.8719099 0.33848287 0.18559
Proportion of Variance 0.3660271 0.2499906 0.2324415 0.1267045 0.01909511 0.00574
Cumulative Proportion  0.3660271 0.6160177 0.8484592 0.9751637 0.99425879 1.00000
```

Les variàncies de les components principals sortint de la matriu de correlacions són

```
> (VARm=SDm^2) #elevem element a element al cuardat (sd^2=VAR)
    Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
2.19616264 1.49994343 1.39464912 0.76022689 0.11457065 0.03444727
```

i sumen 6:

```
> sum(VARm)
[1] 6
```

## 5.5 Qüestó 4

**Escriu explícitament les tres primeres components principals a partir de les variables estandaritzades.**

```
V=summary(PC2,loadings=TRUE)$loadings
sdata=scale(data,center=TRUE,scale=TRUE)
Y=sdata%*%V #matriu de totes les CP

(TREScompPRINCIPALS=matrix(c(Y[,1],Y[,2],Y[,3]),ncol=3))
```

```
> (TREScompPRINCIPALS=matrix(c(Y[,1],Y[,2],Y[,3]),ncol=3))
              [,1]        [,2]        [,3]
 [1,]  0.532332536  0.78235037 -1.34588548
 [2,]  1.399704841 -2.83072462 -1.25975067
 [3,]  0.591644847  0.58003001  0.98319718
 [4,] -0.503129734  0.02840114  0.35913801
 [5,] -1.373644350  1.85722898 -1.75439732
 [6,]  1.412209670  1.19572930  0.07846869
 [7,] -6.433953797 -1.64791000  2.25824210
 [8,]  0.501984723  0.48004666  0.26291553
 [9,] -1.744606366  1.02667240 -0.73747965
[10,]  0.117376877  0.63253569 -0.41783490
[11,]  0.006794393 -1.19694149  0.98605184
[12,]  0.204888809 -1.93911999 -1.25067664
[13,]  0.130360840 -0.06045504 -1.62986106
[14,] -2.140108638 -0.26728251 -0.14523857
[15,]  0.216417826  0.96432618 -0.58731715
[16,] -0.502181285 -0.11133272  1.96914319
[17,] -0.304540747  0.35606906 -0.28191322
[18,]  1.212783658  0.83870887  1.85309045
[19,]  0.129692318 -0.24896688 -0.27211558
[20,]  1.591824782  0.33828439  0.82940456
[21,]  0.418539813  0.53392055  0.36987457
[22,]  0.570758375  0.32107784  1.10120305
[23,]  1.514348054  1.38746239  2.57462175
[24,] -1.373956086  0.15568468 -1.67052547
[25,] -1.481626779  0.24372790 -1.72933009
[26,]  0.898885578  0.53679592  0.84868801
[27,]  1.436017653  0.88969965  1.96743320
[28,]  0.581912895  0.74296084  0.06018035
[29,]  0.131997884 -0.38006213 -1.22064632
[30,] -2.762753441 -0.65039847  1.39775143
[31,]  2.410155830 -4.13972216  0.92999906
[32,] -0.318310511  1.01403955 -0.73890286
[33,] -0.069078328  1.02122065 -0.87684709
[34,]  1.157618046  0.33083906  0.50237939
[35,]  0.901198527 -1.52836100 -1.54589761
[36,]  0.495913196 -2.22761389 -0.22385895
[37,] -0.475769037  1.57782470 -0.60124290
[38,] -0.282675696 -0.37966586  0.15376168
[39,]  0.022647076 -0.05389785  0.34953072
[40,]  0.194408059 -0.66777870 -1.11734899
[41,]  0.983917689  0.49459653 -0.42800426
```

## 5.6 Qüestó 5

**Quina és la correlació entre la variable wind estandarditzada i la primera component principal?**
Com que les dades estan estandaritzades (wind és la variable 4) :

$$\mathrm{Corr}(X_4, Y_1) = v_{4,1}\sqrt{\lambda_1}$$

Per tant podem fer

```
> V[4,1]*sqrt(VARm[1])
```

```
> V[4,1]*sqrt(VARm[1])
     Comp.1
-0.5243698
```

Una manera més general és trobar la matriu de correlacions $C$ i seleccionar l'element $c_{4,1}$:

```
vaps=eigen(cor(data))$value
veps.matrix=eigen(cor(data))$vectors
VapMatrix<-diag(c(vaps))
sigma=sqrt(VapMatrix)
#C=V%*%sigma    #tambe es pot usar aixi ja que V=VapMatrix
C=veps.matrix%*%sigma

C[4,1]
```

```
> C[4,1]
[1] -0.5243698
```

## 5.7 Qüestó 6

**Quins són els valors de les dues primers components principals corresponents a Albany i a Kansas City.** Tenint en compte que la matriu de les components principals $Y$ ja l'hem calculat:

```
> round(Y,4)
                 Comp.1  Comp.2  Comp.3  Comp.4  Comp.5  Comp.6
Albany           0.5323  0.7824 -1.3459  0.8719  0.0457  0.0618
Albuquerque      1.3997 -2.8307 -1.2598 -0.0948 -0.2398 -0.0302
Atlanta          0.5916  0.5800  0.9832 -0.2264  0.1166 -0.0587
Baltimore       -0.5031  0.0284  0.3591 -0.0863  0.3186  0.1910
Buffalo         -1.3736  1.8572 -1.7544 -0.8459 -0.7307 -0.0188
Charleston       1.4122  1.1957  0.0785  1.9868 -0.2821 -0.0574
Chicago         -6.4340 -1.6479  2.2582  0.8112  0.1829 -0.2209
Cincinnati       0.5020  0.4800  0.2629  1.6610  0.0564 -0.1151
Cleveland       -1.7446  1.0267 -0.7375  0.0294 -0.5531 -0.4617
Columbus         0.1174  0.6325 -0.4178  0.8704 -0.0585  0.2430
Dallas           0.0068 -1.1969  0.9861 -1.6656 -0.1752 -0.1358
Denver           0.2049 -1.9391 -1.2507  0.4386 -0.1947 -0.1201
Des Moines       0.1304 -0.0605 -1.6299 -0.9621  0.3258  0.0349
Detroit         -2.1401 -0.2673 -0.1452  0.3702 -0.2231  0.3919
Hartford         0.2164  0.9643 -0.5873  0.5417  0.6332 -0.3204
Houston         -0.5022 -0.1113  1.9691 -1.5356 -0.3694  0.2242
Indianapolis    -0.3045  0.3561 -0.2819  0.0381  0.1499  0.3573
Jacksonville     1.2128  0.8387  1.8531 -0.4661 -0.1646  0.1750
Kansas City      0.1297 -0.2490 -0.2721 -0.4959  0.3895  0.0083
Little Rock      1.5918  0.3383  0.8294  0.0508  0.5626 -0.1128
Louisville       0.4185  0.5339  0.3699  0.6740  0.1812  0.2438
Memphis          0.5708  0.3211  1.1012 -0.4116  0.3338  0.1409
Miami            1.5143  1.3875  2.5746 -0.8324 -0.6950 -0.2660
Milwaukee       -1.3740  0.1557 -1.6705 -0.7491  0.0241  0.0999
Minneapolis     -1.4816  0.2437 -1.7293  0.3194 -0.1484  0.0044
Nashville        0.8989  0.5368  0.8487  0.6453  0.1632  0.0433
New Orleans      1.4360  0.8897  1.9674 -0.2916  0.0864 -0.0883
Norfolk          0.5819  0.7430  0.0602 -1.0707 -0.0547  0.0499
Omaha            0.1320 -0.3801 -1.2206 -0.8976  0.2279  0.0727
Philadelphia    -2.7628 -0.6504  1.3978  0.3876  0.2477  0.1105
Phoenix          2.4102 -4.1397  0.9300  0.9326 -0.6209 -0.0232
Pittsburgh      -0.3183  1.0140 -0.7389  0.6115 -0.3422  0.1101
Providence      -0.0691  1.0212 -0.8768 -0.4987  0.4354 -0.2551
Richmond         1.1576  0.3308  0.5024  0.8607  0.2394 -0.0171
Salt Lake City   0.9012 -1.5284 -1.5459  0.5429 -0.0716 -0.1002
San Francisco    0.4959 -2.2276 -0.2239  0.1074  0.2191  0.0924
Seattle         -0.4758  1.5778 -0.6012  0.7533 -0.6399  0.0694
St. Louis       -0.2827 -0.3797  0.1538 -0.0543  0.1921 -0.3737
Washington       0.0226 -0.0539  0.3495 -0.0310  0.0334  0.2011
Wichita          0.1944 -0.6678 -1.1173 -2.4253  0.0601 -0.0531
Wilmington       0.9839  0.4946 -0.4280  0.1363  0.3391 -0.0974
```

```
> Y[1,]
     Comp.1      Comp.2      Comp.3      Comp.4      Comp.5      Comp.6
 0.53233254  0.78235037 -1.34588548  0.87190217  0.04566668  0.06180699
```

```
> Y[19,]
      Comp.1       Comp.2       Comp.3       Comp.4       Comp.5       Comp.6
 0.129692318 -0.248966883 -0.272115581 -0.495889617  0.389528211  0.008337687
```

És a dir, el valor de la priemra component principal per Albany és 0.53233254 i el de la segona 0.78235037. Similarment els valors per a Kansas City són 0.12962318 per a la primera component principal i -0.248966883 per a la segona component principal.

## 5.8 Qüestó 7

Quins són els valors de les dues primers components principals estandarditzades corresponents a Albany i a Kansas City

```
> scale(Y)[1,]
    Comp.1     Comp.2     Comp.3     Comp.4     Comp.5     Comp.6
 0.3592119  0.6387984 -1.1396608  0.9999911  0.1349158  0.3330123
```

```
> scale(Y)[19,]
     Comp.1      Comp.2      Comp.3      Comp.4      Comp.5      Comp.6
 0.08751490 -0.20328444 -0.23042040 -0.56873951  1.15080628  0.04492294
```

És a dir, el valor de la priemra component principal per Albany és 0.3592119 i el de la segona 0.6387984. Similarment els valors per a Kansas City són 0.08751490 per a la primera component principal i -0.20328444 per a la segona component principal.

# 6 Pràctica 6

## 6.1 Introducció

L'objectiu d'aquesta pràctica és continuar practicant les tècniques de components principals, que iniciàrem a la pràctica 5. Utilitzarem les dades `pottery`, amb les quals ja havíem treballat en sessions anteriors, sense la variable `kiln` (forn de procedència), ja que és una variable categòrica.

Per a aplicar el prodediment de components principals vam utilitzar la comanda `princomp()`. També podem fer servir la instrucció `prcomp()`. Mira a l'ajuda de R les diferències entre elles. Observa que si vols sortir de la matriu de correlacions pots dir

```
y=princomp(data,cor=TRUE)
```

però en el segon cas has d'escirure

```
y=prcomp(data,scale=TRUE)
```

Per a calcular els scores dels individus podries definir la funció apropiada (veure la pàgina 27 del manual *A Little Book of R for Multivariate Analysis*, però també són una sortida del procediment de components principals. En particular, si poses `prcomp()`, els obtindràs amb la comanda

```
nomdelprocediment$x[,]
```

o, si fem servir `princomp()`, amb les comandes

```
nomdelprocediment$x[,]
```

Els vectors propis els puc reclamar amb la comanda

```
nom_pca$rotation[,]
```

en el primer cas, i amb

```
nom_pca$load[,]
```

en el segon. Finalment els screegraph els puc fer amb

```
screeplot(nom de la sortida del procediment de components principals, type="lini
```

o també amb el plot adequat.

## 6.2 Qüestió 1

**Sortint de la matriu de correlacions, quines serna les vari'ancies de les components principals? Comprova que la suma és la correcta.**

```
PC1=princomp(data, cor=TRUE)
PC11=prcomp(data, scale=TRUE) #equival a PC1
sd=summary(PC1)$sd
var=sd^2
var
sum(var)
```

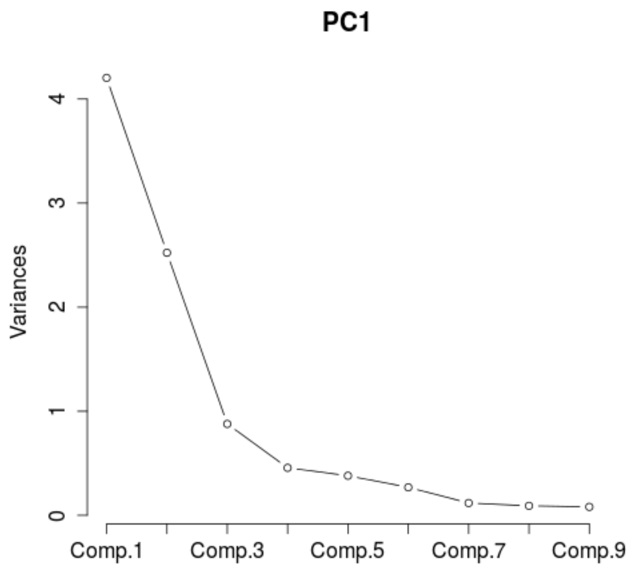
```
> sum(var)
[1] 6
```

Que coincideix amb el nombre de variables .

## 6.3 Qüestió 2

**Dibuixa el `screegraph` amb la companda plot, amb línies entre els punts de la gràfica.**

```
dev.copy(png,'PP66.png')
#screeplot(PC1,type="lines")
plot(PC1, type="l")
dev.off()
graphics.off()
```



## 6.4 Qüestió 3

Escriu l'expressió general de la primera i de la segona components principals i comprova que els escalars que multipliquen les variables formen vectors de mòdul unitat. (deixa ben clar com has d'escriure les dades inicials)

$$Y_j = v_{1,j}X_1 + v_{2,j}X_2 + \dots v_{9,j}X_9 \text{ per } j = 1, 2, \dots, 9$$

Comprovem que els escalars que multipliquen les variables formen vectors de mòdul unitat, de fet, aquest escalars són els vectors propis unitaris de la matiru de covariàncies (o correlació, com és aquest cas):

```
veps=eigen(cor(data))$vector #veps = V
sqrt(sum(veps[,1]^2)) #modul 1r vector (1a CP)
sqrt(sum(veps[,2]^2)) #modul 2n vector (2a CP)
```

```
> sqrt(sum(V[,1]^2))
[1] 1
> sqrt(sum(V[,2]^2))
[1] 1
```

### 6.5 Qüestió 4

**Calcula programant tu els càlculs, el valor de la primera i la segona component principal per a l'individu 1 i 2 (atenció com utilitzes les dades d'aquest individu).**

```
PC11=prcomp(data, scale=TRUE)
PC1=princomp(data, cor=TRUE)

veps=eigen(cor(data))$vector
Y=t(veps)%*%t(scale(data))
Y[1,1] #individu 1 1a CP a ma
PC11$x[1,1] # directe
Y[2,1] #individu 1 2a CP
PC11$x[1,2] #directe
Y[1,2] #individu 2 1a CP
PC11$x[2,1] #directe
Y[2,2] #individu 2 2a CP
PC11$x[2,2] #directe
Y
PC11$x

Y[1,2]
PC1$scores[1,1]
PC11$x[1,1]
PC1$scores[2,1]
PC11$x[2,1]
PC1$scores[1,2]
PC1$scores[2,2]
Y1[1]
Y1[2]
Y2[1]
Y2[2]
```

```
> Y[1,1] #individu 1 1a CP a ma
          1
-0.02358274
> PC11$x[1,1] # directe
[1] 0.02358274
> Y[2,1] #individu 1 2a CP
       1
1.796329
> PC11$x[1,2] #directe
[1] 1.796329
> Y[1,2] #individu 2 1a CP
        2
0.2306258
> PC11$x[2,1] #directe
```

```
[1] −0.2306258
> Y[2,2] #individu 2 2a CP
      2
1.631135
> PC11$x[2,2] #directe
[1] 1.631135
```

NO CUADRA

## 6.6 Qüestió 5

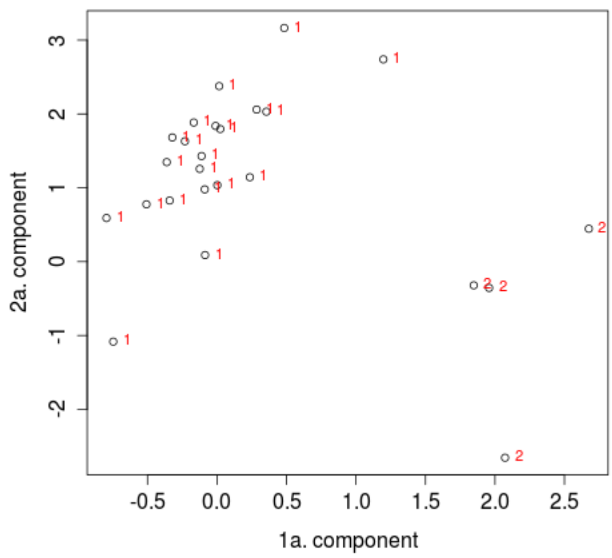
**Troba els valors de la primera i segona component principal pels primers 25 individus, amb les comandes que estan dintre dels procediments de components principals.**

```
PC1$scores[c(1:25),1] #1a CP dels 25 primers individus
PC1$scores[c(1:25),2] #2a CP dels 25 primers individus
```

```
> round(PC1$scores[c(1:25),1],2) #1a CP dels 25 primers individus
    1     2     3     4     5     6     7     8     9    10    11    12
13    14    15    16    17    18    19
−0.02  0.23  0.13  0.33  0.17  0.37 −0.02  0.11  0.00  0.09  0.75  0.09 −0.24 −0
0.01 −0.49  0.51
   20    21    22    23    24    25
 0.80  0.34 −2.71 −1.98 −1.87 −2.10
> round(PC1$scores[c(1:25),2],2) #2a CP dels 25 primers individus
    1     2     3     4     5     6     7     8     9    10    11    12
13    14    15    16    17    18    19
 1.82  1.65  1.27  1.70  1.91  1.36  2.40  1.45  1.05  0.09 −1.10  0.99  1.16
2.05  2.08  2.77  1.86  3.20  0.79
   20    21    22    23    24    25
 0.60  0.84  0.45 −0.36 −0.32 −2.69
```

## 6.7 Qüestió 6

Dibuixa en un plot el diagrama de dispersió de les dues components principals corresponents al primers 25 individus, posant en el gràfic el forn del qual prové.



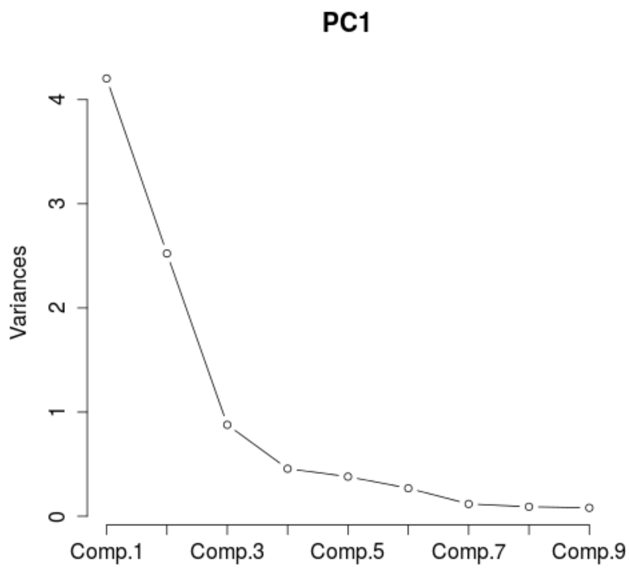
## 6.8 Qüestió 7

Determina amb quantes components et quedar'as usant tres mètodes diferents.

- Amb el primer mètode triem dues components principals ja que expliquen més del 70% de la variància total. (les dues primeres components principals expliquen 74.75% de la variància.

```
> summary(PC11)
Importance of components:
                          PC1    PC2     PC3     PC4     PC5     PC6
PC7      PC8     PC9
Standard deviation     2.0503 1.5885 0.93699 0.67538 0.61647 0.51840 0.34325
Proportion of Variance 0.4671 0.2804 0.09755 0.05068 0.04223 0.02986 0.01309
Cumulative Proportion  0.4671 0.7475 0.84501 0.89570 0.93792 0.96778 0.98087
```

- Amb el mètode del scree graph ens quedem també amb dues components principals




- Amb el darrer mètode seleccionarem les variàbles que tinguin una proporció de variància superior a la mitjana:

```
> var=PC11$sd^2
> var
[1] 4.20390772 2.52328456 0.87794165 0.45614191
[5] 0.38003864 0.26873670 0.11782266 0.09114400
[9] 0.08098216
> mean(var)
[1] 1
```

i per tant ens quedem amb les dues primeres components principals.

# 7 Pràctica 7

## 7.1 Introducció

L'objectiu d'aquesta pràctica és aplicar algunes tècniques de components principals, a les dades anomenades `wine.data`, es troben a

```
http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data
```

Les podem carregar a R amb

```
data <- read.csv("~/wine.data", header=FALSE)
```

```
> data
    V1    V2   V3   V4   V5  V6   V7   V8   V9  V10       V11   V12  V13  V14
1    1 14.23 1.71 2.43 15.6 127 2.80 3.06 0.28 2.29  5.640000 1.040 3.92 1065
2    1 13.20 1.78 2.14 11.2 100 2.65 2.76 0.26 1.28  4.380000 1.050 3.40 1050
..............................................................................
178  3 14.13 4.10 2.74 24.5  96 2.05 0.76 0.56 1.35  9.200000 0.610 1.60  560
```

Treballa a partir d'ara, amb les variables V3 a V10 de l'arxiu, i amb els individus que provenen de la priemra i segona zona vinícola.

```
newdata1 <- subset(data, V1==c("1"), select=3:10)
newdata2 <- subset(data, V1==c("2"), select=3:10)
newdata=rbind(newdata1,newdata2)
newdata
```

```
> newdata
      V3   V4   V5  V6   V7   V8   V9  V10
1   1.71 2.43 15.6 127 2.80 3.06 0.28 2.29
2   1.78 2.14 11.2 100 2.65 2.76 0.26 1.28
.........................................
130 4.30 2.38 22.0  80 2.10 1.75 0.42 1.35
```

A teoria hem donat com a $C$ una matriu que ens dóna les correlacions de les velles variables amb les components principals. Evidentment els seus elements depenen de si sortim de la matriu de covariàncies o correlacions.
L'element $c_{i,k}$ de $C$ és la correlació de la variable $X_i$, amb la component principal $Y_k$. Recorda també que els vectors fila de $C$ tenen mòdul 1.

## 7.2 Qüestió 1

Després d'executar el procediment de components principals sortint de les dades estandaritzades, troba amb el joc de matrius apropiat, la matriu $C$ per a les nostres dades. Quina serà la correlació de la primera variable amb la segona component principal?

```
PCAdades=prcomp(scale(newdata))
V=PCAdades$rotation #matriu de veps
VAPS=diag(c(PCAdades$sd^2))
sigma=sqrt(VAPS)

C=V%*%sigma #ja que les dades son escalades
C1=C
C
```

```
> round(C,2)
     [,1] [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]
V3   0.07 0.38  0.81  0.32 -0.27  0.11  0.02  0.01
V4   0.30 0.82 -0.29 -0.05 -0.15  0.08 -0.35 -0.07
V5  -0.31 0.73  0.04 -0.15  0.49  0.24  0.22  0.03
V6   0.51 0.16 -0.42  0.71  0.01  0.00  0.17  0.04
V7   0.88 0.02 -0.01 -0.27 -0.17  0.08  0.27 -0.20
V8   0.89 0.15  0.02 -0.32 -0.13 -0.07  0.04  0.25
V9  -0.59 0.49 -0.09 -0.07 -0.24 -0.55  0.17 -0.01
V10  0.69 0.02  0.29  0.09  0.48 -0.43 -0.12 -0.06
```

Així doncs la correlació de la priemra variable amb la segona component principal és:

```
apply(C^2,1,sum) #comprovem que les files de C^2 sumen 1
V[1,2]*PCAdades$sd[2]
C[1,2]
```

```
> apply(C^2,1,sum) #comprovem que les files de C^2 sumen 1
 V3  V4  V5  V6  V7  V8  V9 V10
  1   1   1   1   1   1   1   1
> V[1,2]*PCAdades$sd[2]
[1] 0.378899
> C[1,2]
      V3
0.378899
```

### 7.3 Qüestió 2

Fes el mateix sortint ara de les dades sense estandaritar, és a dir usant la matriu de covariàncies.

```
X=newdata
PCAdades=prcomp(newdata, scale=FALSE)
V=PCAdades$rotation #matriu de veps
VAPS=diag(c(PCAdades$sd^2))
sigma=sqrt(VAPS)

S=diag(diag(cov(X)))
S=solve(sqrt(S))
C=S%*%V%*%sigma #ja que les dades son escalades
C2=C
```

```
> round(C,2)
      [,1]  [,2]  [,3]  [,4]  [,5]  [,6]  [,7]  [,8]
[1,]  0.02 -0.15  0.67  0.72 -0.04 -0.01  0.00  0.00
[2,] -0.30 -0.41  0.29 -0.19 -0.31  0.54 -0.48  0.04
[3,]  0.13 -0.99  0.00 -0.01  0.00  0.00  0.00  0.00
[4,] -1.00 -0.01  0.00  0.00  0.00  0.00  0.00  0.00
[5,] -0.31  0.20  0.64 -0.52 -0.22 -0.33 -0.15 -0.01
[6,] -0.26  0.16  0.73 -0.57 -0.13  0.13  0.13  0.00
[7,]  0.22 -0.31 -0.23  0.22 -0.11  0.31 -0.10 -0.80
[8,] -0.28  0.07  0.55 -0.27  0.73  0.01 -0.07  0.00
```

```
> C[1,2]
[1] -0.1496345
```

### 7.4 Qüestió 3

En aquest segon cas explicita expressió que em dóna el valor de la primera component principal estandaritzada (les variàncies són els valors propis) per al segon individu.

```
PCdades=prcomp(newdata, scale=FALSE)
PC1=princomp ( newdata , cor=FALSE)
scale(PC1$scores)[2,1]
scale(PCdades$x)[2,1]]
```

```
> scale(PC1$scores)[2,1]
[1] -0.02059139
> scale(PCdades$x)[2,1]
[1] -0.02059139
```

$$Y[2,1]/(\lambda_1)$$

## 7.5 Qüestió 4

Comprova que totes les files, tant en un cas com en un altre, tenen mòdul unitat.

```
apply(C1^2,1,sum) #les files de C^2 sumen 1
apply(C2^2,1,sum) #les files de C^2 sumen 1
```

```
> apply(C1^2,1,sum) #les files de C^2 sumen 1
 V3  V4  V5  V6  V7  V8  V9 V10
  1   1   1   1   1   1   1   1
> apply(C2^2,1,sum) #les files de C^2 sumen 1
[1] 1 1 1 1 1 1 1 1
```

## 7.6 Qüestió 5

Quin serà el mòdul dels vectors columna de la matriu $C$ si surts de la matriu de correlacions mostral?

```
PCAdades=prcomp(scale(newdata))
(PCAdades$sd)^2

apply(C1^2,2,sum)
#si sortim de dades escalades, les columnes de C sumen els vaps lambda
```

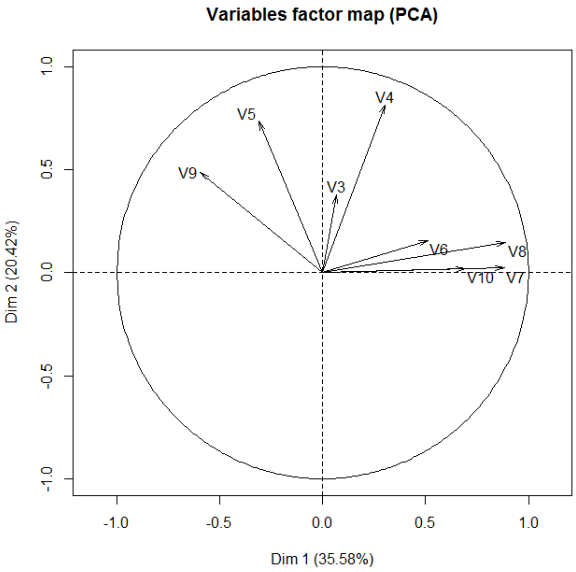
```
> (PCAdades$sd)^2
[1] 2.85 1.63 1.02 0.82 0.67 0.57 0.32 0.12
> apply(C1^2,2,sum)
[1] 2.85 1.63 1.02 0.82 0.67 0.57 0.32 0.12
```

## 7.7 Qüestió 6

Un gràfic interessant és el que a cada variable li fa correspondre el punt de coordenades les correlacions amb les dues primeres components principals. Per a dibuixar-lo, carregueu el paquet `FactoMineR`, apliqueu el mètode de components amb la instrucció `PCA()`, que també ens fa el procediment de components principals (veure informació a R), i executant la comanda

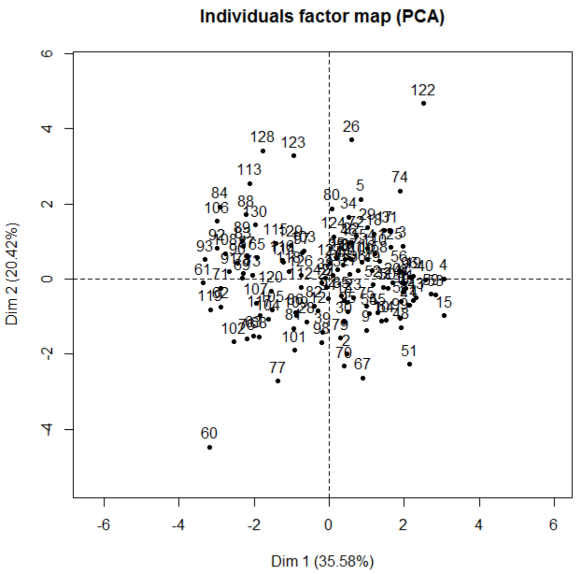
```
library(FactoMineR)
PCA1=PCA(newdata, scale.unit=TRUE)
plot.PCA(PCA1, choix="var", axes=1:2)
```

se'm mostra el dibuix.




Què passa si executes

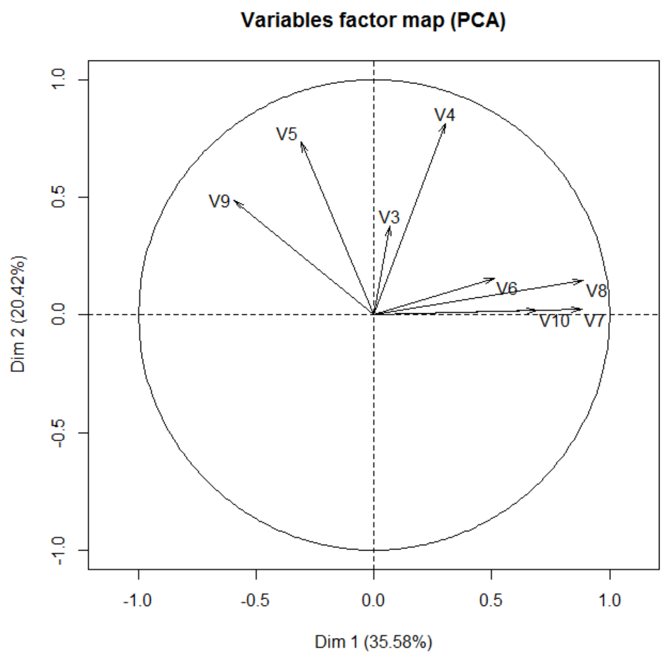
```
plot.PCA(PCA1, choix="ind", axes=1:2)
```



## 7.8 Qüestió 7

Dibuixa el gràfic de les variables en el pla de les dues primers components principals i analitza'l, comentant quina variable està més ben representada per les dues primeres components principals, i quines estan més correlades amb la primera component i quines amb la segona. Comenta també què significa la circumferència que apareix en el dibuix.

```
library(FactoMineR)
PCA1=PCA(newdata, scale.unit=TRUE)
plot.PCA(PCA1, choix="var", axes=1:2)
```




Les coordenades són les correlacions.
Hem dibuixat un vector per a cada variable, de components els primers 2 elements de la fila $i$ de la matriou $C$, és a dir, les correlacions de la varialbe $X_i$ amb les dues primeres components principals.
Recorda que si agafem tota la fila de la matriu $C$, aquest vector serà unitari, com només considerem 2 elements, el vector tindrà norma més petita o igual a 1, és a dir estarà dintre del cercle unitat.
Aquest gràfic és molt adequat per a veure qjines variables estan millor representades per les dures primeres components principals, i també per veure com estan correlades amb aquestes components.

Les variables més ben explicades en dimensió 2 (amb les 2 primeres CP) són les que s'aproximen més a la circunferència de radi 1, en aquest cas les variables 4, 7 i 8 són les més ben explicades per les dues primeres CP.

Les variables 7,8,9 i 10 són les més correladesamb la primera component principal (nota que 9 és inversament correlada, i les altres directament), en canvi les variables 4 i 5 són les més correlades amb la segona component principal.

El crecle mostra la correlació total, els vectors que toquin el crecle unitat són els que estan totalemnt correlacionats amb les dues primeres components pinricpals.

### 7.9 Qüestió 8

Defineix una funció que em faci el test d'esfericitat pels $k$ darrers valors propis i em retorni el $p$-valor i el quantil corresponent a 0.99.

```
testEsfericitat <- function(k,dades){ #torna matriu mb U i CHI2
  n=dim(dades)[1]
  p=dim(dades)[2]
  r=p-k

  PC=prcomp(scale(dades))
  lambdas=PC$sd^2
  lambdaMean=mean(lambdas)

  sumatori=sum(log(lambdas[(r+1):p]))
  U=(n-((2*p+11)/6))*(k*log(lambdaMean)-sumatori)

  v=(1/2)*(k-1)*(k+2)
  CHI=qchisq(.99, df=v)
  return(matrix(c( U,CHI),ncol=2))
}


decidirEsfericitat <-function(testesfericitat){
  U=testesfericitat[1]
  CHI=testesfericitat[2]
  if(U>=CHI) return(print("FALS, les lambda_{r+1} =! .. != lambda_{p}"))
  else return(print("CERT, lambda_{r+1}= .. = lambda_p"))
}


testEsfericitat(2,newdata)
decidirEsfericitat(testEsfericitat(2,newdata))
```

```
> testEsfericitat(2,newdata)
         [,1]    [,2]
[1,] 411.4032 9.21034
> decidirEsfericitat(testEsfericitat(2,newdata))
[1] "FALS, les lambda_{r+1} =! .. != lambda_{p}"
```

Els dos darrers vaps són diferents, per tant tots els vaps són diferents amb un 99% de confiança. Recorda que si volem veure les lambdas:

```
prcomp(scale(newdata))$sd^2
```

```
> round(prcomp(scale(newdata))$sd^2,4)
[1] 2.8468 1.6332 1.0152 0.8239 0.6698 0.5740 0.3188 0.118
```

# 8 Pràcica 8

## 8.1 Introducció

L'objectiu d'aquesta pràctica és utilitzar les components principals com a variables explicatives en una regressió, i a més treballar amb la regressió multivariant.
L'avantatge d'utilitzar components principals com a variables explicatives és que aquestes són no correlades, i per tant evitem possibles problemes de multicolinealitat. Utilitzar variables no correlades també implica que a l'afegir variables no ens canvien els valors de les estimacions de les $\beta$'s anteriors.

Per tant si emprem les components principals com a regressors, evitem multicolinealitats, i podem potser reduir el nombre de variables.

Utilitzarem les de `pottery` amb les quals ja havíem treballat en sessions anteriors. El nostre interès és explicar la primera variable `Al2O3` a partir de les restants (com sempre no has de considerar la variable categòrica `kiln`).

Examina la matriu de covariàncies de les variables explicatives i veuràs que tens valors propis molt propers a zero, per tant apareixeran problemes de multicolinealitat. Una manera d'evitar-la seria eliminant variables a partir de les seves relacions (donades, com sabeu, pels vectors propis corresponents). Altra mètode seria fer un procediment de components principals (per les variables explicatives), calcular els scores dels individus i utilitzarlos com a regressors (recorda que els valors de les components principals pels diferents individus és una sortida del procediment de components principals).

Tingues en compte que a l'emprar components principals podria passar que una component amb variància petita sigui un predictor significant de la variable resposta.

## 8.2 Qüestió 1

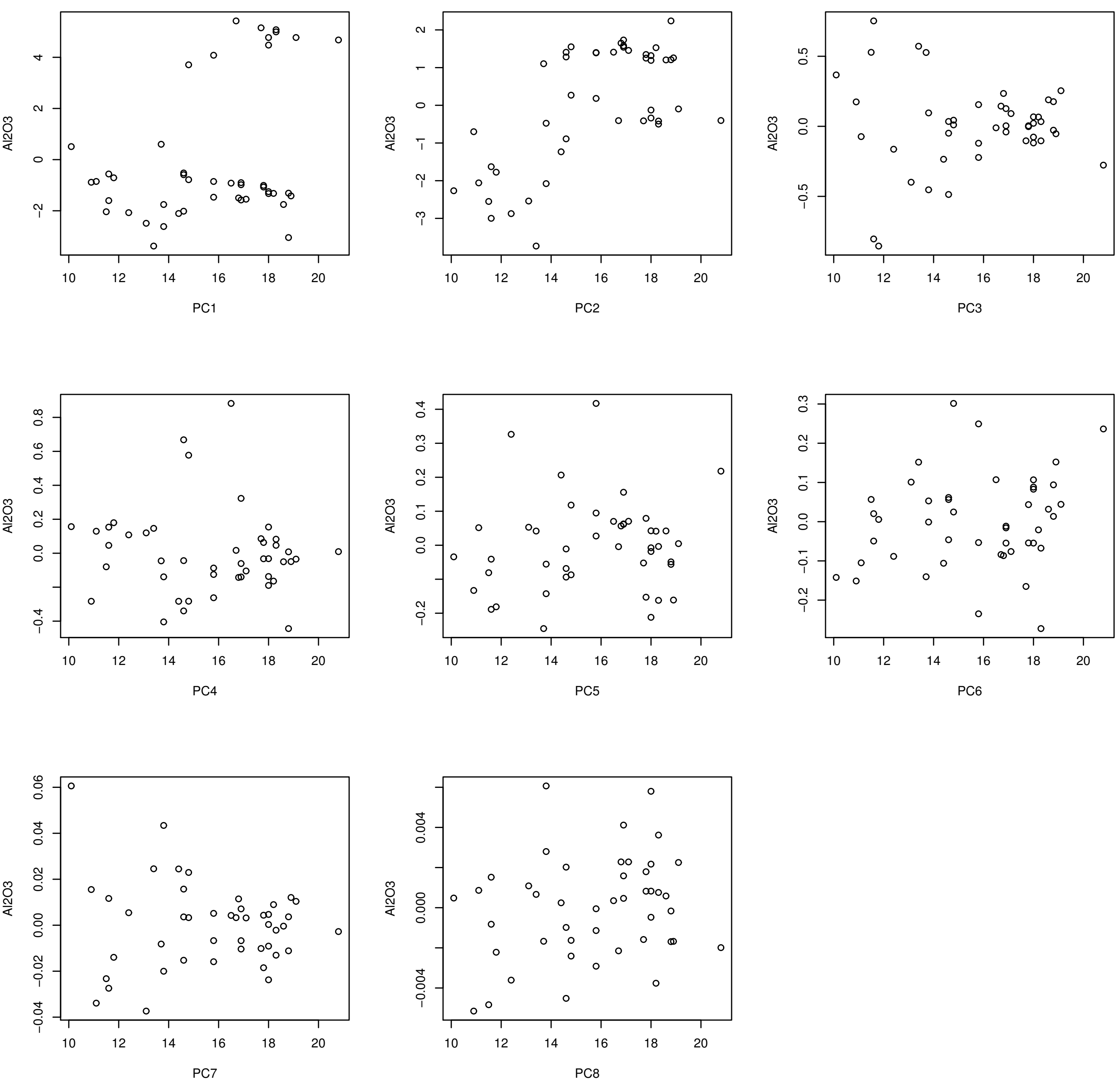
**Dibuixa el diagrama de barres de punts de Al2O3 contra cada una de les variables regressores.**
Primer carreguem els paquets, les dades i treiem les variables categòriques i la variable objectiu de la regressió:

```
library(MVA)
library(HSAUR2)

dades=pottery[c(-10,-1)] #cal treure 10 ja que es categorica
#treiem la 1a ja que es la que modelitzarem (regressio)
```

Ara calculem les components principals (que les farem servir de variables regresores), ho fem a partir de la matriu de covariàncies ja que totes les variables de les dades tenen la mateixa untiat de mesura.

```
dades.pca=prcomp(dades)
Y=dades.pca$x
p=dim(Y)[2]
```

```
dev.off()
graphics.off()
#png("AAA.png")

par(mfrow=c(3,3))
p=dim(Y)[2]
out<-sapply(1:p, function(i){ plot(pottery$Al2O3, Y[,i],
    xlab = paste("PC", i, sep = ""), ylab = "Al2O3")})
```



**Fes la regressió de Al2O3 contra cada una de les components principals (un sol regressor), i fes la regressió amb totes les components principals com a variables explicatives (nota que les estimacions dels paràmetres ja calculats es conserven tal com diu la teoria).**

```
reg1=lm(Al2O3 ~ Y[,1], data =pottery)
reg2=lm(Al2O3 ~ Y[,2] , data =pottery)
reg3=lm(Al2O3 ~ Y[,3], data =pottery)
reg4=lm(Al2O3 ~ Y[,4], data =pottery)
reg5=lm(Al2O3 ~ Y[,5] , data =pottery)
reg6=lm(Al2O3 ~ Y[,6] , data =pottery)
reg7=lm(Al2O3 ~ Y[,7] , data =pottery)
reg8=lm(Al2O3 ~ Y[,8] , data =pottery)
REG=lm(Al2O3 ~ Y, data =pottery)  #aqui fem la regressio amb totes les CP

summary(reg1)
summary(REG)
```

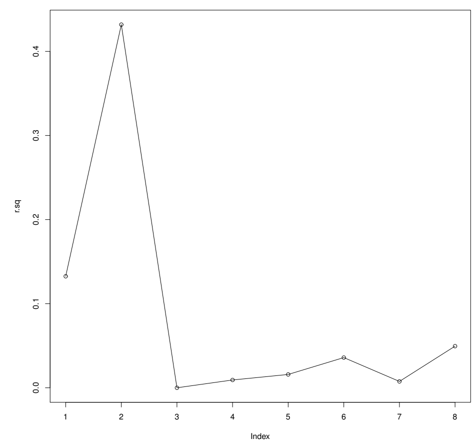
```
> summary(reg1)

Call:
lm(formula = Al2O3 ~ Y[, 1], data = pottery)

Residuals:
    Min      1Q  Median      3Q     Max
-5.7971 -1.4175  0.4073  1.7744  4.2176

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.7089     0.3796  41.381   <2e-16 ***
Y[, 1]        0.3698     0.1442   2.564   0.0139 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 2.547 on 43 degrees of freedom
Multiple R-squared:  0.1326,    Adjusted R-squared:  0.1124
F-statistic: 6.573 on 1 and 43 DF,  p-value: 0.01393

> summary(REG)

Call:
lm(formula = Al2O3 ~ Y, data = pottery)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8585 -0.9643 -0.1091  1.1241  2.6740

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.70889    0.25100  62.585  < 2e-16 ***
```

```
YPC1          0.36977     0.09537    3.877   0.00043 ***
YPC2          1.11300     0.15906    6.997  3.31e-08 ***
YPC3         -0.02334     0.82672   -0.028   0.97763
YPC4         -1.02533     0.99979   -1.026   0.31195
YPC5          2.54346     1.89710    1.341   0.18841
YPC6          4.30807     2.13462    2.018   0.05107 .
YPC7        -12.65821    13.66661   -0.926   0.36050
YPC8        235.07630    99.22810    2.369   0.02332 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.684 on 36 degrees of freedom
Multiple R-squared:  0.6825,    Adjusted R-squared:  0.612
F-statistic: 9.674 on 8 and 36 DF,  p-value: 4.891e-07
```

**Quina serà la component principal més predictiva?**

La component principal més predictiva serà la que tingui major coeficient de determinació $R^2$, fem el gràfic dels $R^2$ de cada regressió:

```
p=dim(Y)[2]
r.sq=matrix(nrow=p)
for(i in 1:p){
r.sq[i,1]=summary(lm(Al2O3~Y[,i],data=pottery))$r.squared
}
dev.off()
graphics.off()
#postscript("B.eps", horizontal=F,
#         width=10, height=10, paper="special", onefile=F)
plot(r.sq,type="l")
```




Es veu doncs que les CP amb coeficient de determinació $R^2$ més gran són la 2ª, 1ª i 8ª.
També ho podem veure mirant el summary del model lineal amb totes les components:

```
REG=lm(Al2O3 ~ Y, data =pottery)
summary(REG)
```

```
> summary(REG)

Call:
lm(formula = Al2O3 ~ Y, data = pottery)

Residuals:
    Min      1Q  Median      3Q     Max
-3.8585 -0.9643 -0.1091  1.1241  2.6740

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  15.70889    0.25100  62.585  < 2e-16 ***
YPC1          0.36977    0.09537   3.877  0.00043 ***
YPC2          1.11300    0.15906   6.997 3.31e-08 ***
YPC3         -0.02334    0.82672  -0.028  0.97763
YPC4         -1.02533    0.99979  -1.026  0.31195
YPC5          2.54346    1.89710   1.341  0.18841
YPC6          4.30807    2.13462   2.018  0.05107 .
YPC7        -12.65821   13.66661  -0.926  0.36050
YPC8        235.07630   99.22810   2.369  0.02332 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
Residual standard error: 1.684 on 36 degrees of freedom
Multiple R-squared:  0.6825,    Adjusted R-squared:  0.612
F-statistic: 9.674 on 8 and 36 DF,  p-value: 4.891e-07
```

**Quina serà la predicció de `Al2O3` per un cas en el qual les variables explicatives, escrites en l'ordre que presenta la base de dades, són** $(7.64, 1.82, 0.77, 0.40, 3.07, 0.98, 0.087, 0.014)$?

```
V=as.matrix(dades.pca$rotation)
CPY=t(V)%*%X #transformem les dades en CP
OY=c(1,CPY)
REG=lm(Al2O3 ~ Y, data =pottery)
REG$coefficients%*%OY #prediccio del model amb totes les CP

#fem ara la prediccio del model amb la 1a, 2a i 8a CP
OY2=c(1,X[1],X[2],X[8])
reg128=lm(Al2O3 ~ Y[,1]+Y[,2]+Y[,8], data =pottery)
reg128$coefficients%*%OY2 #pred. del model amb les CP 1,2 i 8
```

I tenim com a output les prediccions:

```
> REG$coefficients%*%OY #prediccio del model amb totes les CP
         [,1]
[1,] 21.01971
> reg128$coefficients%*%OY2 #pred. del model amb les CP 1,2 i 8
        [,1]
[1,] 23.8507
```

## 8.3 Qüestió 3

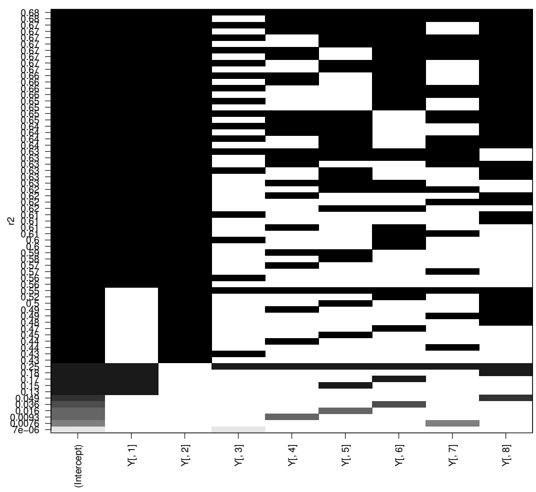
**Per a triar amb quantes variables explicatives et quedes consulta ajudes com a *Quick-R* (Multiple regression) :**

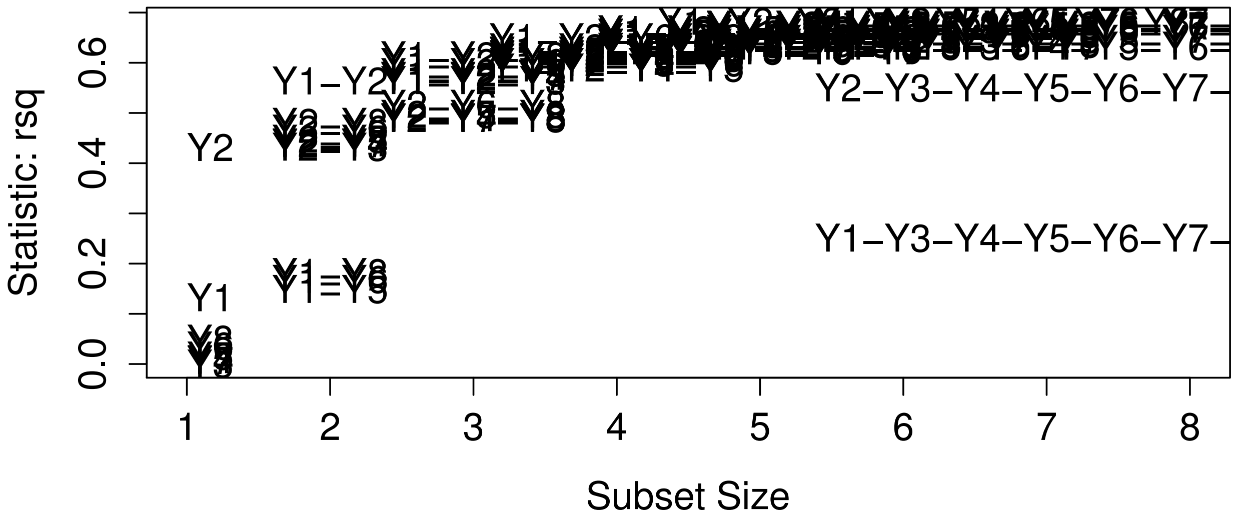
`http://www.statmethods.net/stats/regression.html`

**Allí trobaràs que en el paquet `leaps` hi ha una comanda molt agradable `regsubsets`, també podeu fer selecció *stepwise* amb la companda `stepAIC`.**
**Amb quin model et qudaràs finalment? Aprofitant les idees de *Quick-R* presenta els gràfics de validació adequats.**

```
###   leaps   ###
# All Subsets Regression

library(leaps)
attach(dades)
leaps<regsubsets(pottery$Al2O3~
Y[,1]+Y[,2]+Y[,3]+Y[,4]+Y[,5]+Y[,6]+Y[,7]+Y[,8],data=dades,nbest=10)

# view results
summary(leaps)
# plot a table of models showing variables in each model.
# models are ordered by the selection statistic.
plot(leaps,scale="r2")
# plot statistic by subset size
library(car)
subsets(leaps, statistic="rsq")
```

Ens torna els gràfics:






Aquests dos gràfics ordenen els models segons la predictivitat del model i el nombre de regresors que usin, en els dos gràfics com més alt és l'eix Y més alt és la predictivitat.

En el primer gràfic tenim a l'eix X els regressors i a l'eix Y la predictivitat del model. Cada model és una fila, té en negre les variables que usa, com més alta és la posició en l'eix Y millor descriu la variable Al2O3.

El segon gràfic ens descriu la mateixa informació que el primer, però de forma diferent, a l'eix X tenim el nombre de variables que usa, i a l'eix Y la predictivitat. Per tant a la columna *n* ens ordena per la predictivitat els models formats per *n* regresors.

Amb aquests gràfics és fàcil veure que amb 2 components el millor model és el que usa el primer i segon regressor (CP), i amb 3 components el format per la primera, segona i buitena component principal.

Fem ara selecció *stepwise*:

```
###    step aic    ###
# Stepwise Regression
library(MASS)
fitlm(pottery$Al2O3~
Y[,1]+Y[,2]+Y[,3]+Y[,4]+Y[,5]+Y[,6]+Y[,7]+Y[,8],data=dades)
step <- stepAIC(fit, direction="both")
step$anova # display results
```

té com a resultat el model format per les components princpals 1,2,5,6 i 8.

```
> step$anova # display results
Stepwise Model Path
Analysis of Deviance Table

Initial Model:
  pottery$Al2O3 ~ Y[, 1] + Y[, 2] + Y[, 3] + Y[, 4] + Y[, 5] +
  Y[, 6] + Y[, 7] + Y[, 8]

Final Model:
  pottery$Al2O3 ~ Y[, 1] + Y[, 2] + Y[, 5] + Y[, 6] + Y[, 8]


Step Df    Deviance Resid. Df Resid. Dev      AIC
1                             36   102.0637 54.85205
2 - Y[, 3]  1 0.002260081     37   102.0659 52.85304
3 - Y[, 7]  1 2.432159096     38   104.4981 51.91279
4 - Y[, 4]  1 2.981796242     39   107.4799 51.17886
```

## 8.4 Qüestió 4

**R també té, com sempre, comandes per fer la regressió directament amb components principals. A la llibreria `pls`, hi ha la comanda `pcr` que m'ho fa (mira la documentació).**

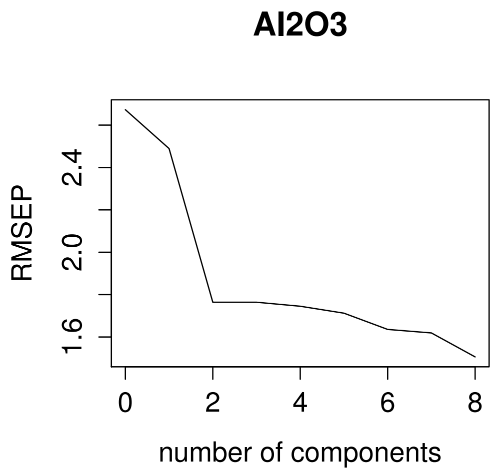
**Respon a la pregunta anterior utilitzant aquestes noves comandes.**
**Fes el gràfic de validació, emprant la comanda**

```
validationplot(, type="MSEP")
```

```
 library(pls)
mod<-pcr(Al2O3 ~ Fe2O3+MgO+CaO+Na2O+K2O+TiO2+MnO+BaO,
data=pottery,validation="none")
summary(mod)
validationplot(mod)
```

que ens torna

```
> mod<-pcr(Al2O3 ~ Fe2O3+MgO+CaO+Na2O+K2O+TiO2+MnO+BaO,data=pottery,validation="
> mod
Principal component regression , fitted with the singular value decomposition al
Call:
  pcr(formula = Al2O3 ~ Fe2O3 + MgO + CaO + Na2O + K2O + TiO2 +     MnO + BaO, d
> validationplot(mod)
> summary(mod)
Data:    X dimension: 45 8
Y dimension: 45 1
Fit method: svdpc
Number of components considered: 8
TRAINING: % variance explained
1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
X        72.13    98.05    99.01    99.67    99.85   100.00    100.0    100.00
Al2O3    13.26    56.44    56.44    57.37    58.95    62.55     63.3     68.25
```



Al gràfic tenim a l'eix X el nombre de components i a l'eix Y el RMSEP (un dindicador de predictivitat), com més baix és lel RMSEP més bona és la predictivitat, per tant, podem concloure que la predictivitat és prou bona en 2 regressors però és millor en 8 regressors.

## 8.5 Qüestió 5

**En la regressió multivariant la variablae dependent és un vector de variables. Pots fer-les amb la instrucció**

```
lm(formula = cbind() ~ variables dependents, data = )
```

**Fes la regressió de les dues primeres variables de les dades `pottery` contra les altres.**

```
lm(formula = cbind(Al2O3,Fe2O3) ~ MgO+CaO+Na2O+K2O+TiO2+MnO+BaO,
data =pottery )
```

té per output:

```
> lm(formula = cbind(Al2O3,Fe2O3) ~ MgO+CaO+Na2O+K2O+TiO2+MnO+BaO, data =pottery

Call:
lm(formula = cbind(Al2O3, Fe2O3) ~ MgO + CaO + Na2O + K2O + TiO2 +
    MnO + BaO, data = pottery)

Coefficients:
             Al2O3       Fe2O3
(Intercept)   10.06507   -1.97749
MgO           -0.61648    0.02902
CaO            0.32613    2.74632
Na2O           0.99360    2.02442
K2O            0.26330    1.16540
TiO2           4.09895    2.64948
MnO          -15.10625   14.08161
BaO          205.81580  -77.85608
```

# 9 Pràctica 9

## 9.1 Introducció

Les dades següents, introduïdes com una llista *(*ifxa't en l'estructura de la comanda), donen l'esperança de vida en 31 països. Les quatre primeres variables, anomenades `m0, m25, m50` i `m75`, indiquen la mitjana dels anys que resten de vida als homes que actualment tenen 0, 25, 50 i 75 anys, i les 4 darreres presenten l atemeixa informació per a les dones.

```
"life" <- structure(.Data = list(c(63., 34., 38., 59., 56., 62.,
50., 65., 56., 69., 65., 64., 56., 60., 61., 49., 59., 63., 59.,
65., 65., 64.,
64., 67., 61., 68., 67., 65., 59., 58., 57.)
, c(51., 29., 30., 42., 38., 44., 39., 44., 46., 47., 48., 50., 44.,
44., 45., 40., 42., 44., 44., 48., 48., 63.,
43., 45., 40., 46., 45., 46., 43., 44., 46.)
, c(30., 13., 17., 20., 18., 24., 20., 22., 24., 24., 26., 28., 25.,
22., 22., 22., 22., 23., 24., 28., 26., 21.,
21., 23., 21., 23., 23., 24., 23., 24., 28.)
, c(13., 5., 7., 6., 7., 7., 7., 7., 11., 8., 9., 11., 10., 6., 8.,
9., 6., 8., 8., 14., 9., 7., 6., 8., 10., 8.,
8., 9., 10., 9., 9.)
, c(67., 38., 38., 64., 62., 69., 55., 72., 63., 75., 68., 66., 61.,
65., 65., 51., 61., 67., 63., 68., 67., 68.,
68., 74., 67., 75., 74., 71., 66., 62., 60.)
, c(54., 32., 34., 46., 46., 50., 43., 50., 54., 53., 50., 51., 48.,
45., 49., 41., 43., 48., 46., 51., 49., 47.,
47., 51., 46., 52., 51., 51., 49., 47., 49.)
, c(34., 17., 20., 25., 25., 28., 23., 27., 33., 29., 27., 29., 27.,
25., 27., 23., 22., 26., 25., 29., 27., 25.,
24., 28., 25., 29., 28., 28., 27., 25., 28.)
, c(15., 6., 7., 8., 10., 14., 8., 9., 19., 10., 10., 11., 12., 9.,
10., 8., 7., 9., 8., 13., 10., 9., 8., 10., 11.,
10., 10., 10., 12., 10., 11.)
) , class = "data.frame" , names = c("m0", "m25", "m50", "m75",
"w0", "w25", "w50", "w75") , row.names = c("Algeria", "Cameroon",
"Madagascar", "Mauritius", "Reunion", "Seychelles", "South Africa
(C)", "South Africa (W)",
"Tunisia", "Canada", "Costa Rica", "Dominican Rep.", "El Salvador",
"Greenland", "Grenada", "Guatemala",
"Honduras", "Jamaica", "Mexico", "Nicaragua", "Panama",
"Trinidad (62)", "Trinidad (67)",
"United States (66)", "United States (NW66)", "United States (W66)",
"United States (67)", "Argentina",
"Chile", "Colombia", "Ecuador"))
```

Volem explicar aquestes variables per mitjà de factors. La comanda clau és `factanal`.

## 9.2 Qüestió 1

**Amb quants factors decidiràs treballar? Justifica la resposta.**
Per a decidir quants factors cal usar utilitzo la instrucció

```
sapply(1:3, function(f)
  factanal(life, factors = f, method ="mle")$PVAL)
```

que ens torna

```
> sapply(1:3, function(f)
+   factanal(life, factors = f, method ="mle")$PVAL)
   objective    objective    objective
1.879555e-24 1.911514e-05 4.578204e-01
```

Ens torna els $p$-valors del contrast explicat a classe, on la hipòtesi nul·la és que n'hi ha prou amb el número de factors indicats. Amb aquesta instrucció fem simultaniament contrastos corresponents a 1,2, i 3 factors.

Si considerem com a nivell de significació $\alpha = 0.05$, el cas per 1 i 2 factors tenim que el $p$ valor és més petit que $\alpha$ i per tant suggereix que les dades observades són inconsistents amb la hipòtesi nul·la i per tant, hem de rebutjar la hipòtesi nul·la, és a dir, necessitem usar més factors.

En canvi, el cas en 3 fectors obtenim un $p$-valor de $0.4578204 > \alpha$, per tant no tenim evidència per a rebutjar que 3 factors són suficients.

**Escriu explícitament l'estadístic de contrast per a dos factors, i el valor del quantil que utilitzaries si seguissis el mètode clàssic de contrast d'hipòtesi.**
Recordem la teoria: Sota condicions de normalitat (

$$H_0 : \Sigma = \Lambda\Lambda^t + \Psi$$

on $\Lambda$ és una matriu $p \times m$ amb $m$ (nombre de factors) fixat, mitjançant l'estadísitc de contrast

$$U = \left(n - \frac{2p + 4m + 11}{6}\right) \ln \frac{|\hat{\Lambda}\hat{\Lambda}^t + \hat{\Psi}|}{|S|}$$

on || significa el determinant de la matriu.
Sota la hipòtesi nul·la, l'estadístic $U$ té aproximadament una distribució $\chi^2$ amb $d = \frac{1}{2}|(p-m)^2-(p+m)|$ **graus de llibertat**, la diferència entre equacions i incògnites.
Si $U \geq \chi^2_{d,\alpha}$, aleshores rebutgem la $H_0$ i diem que 'per aquest valor de $m$ el model factorial no és vàlid'. Es pot provar d'augmentar el nombre de factors mentre $d$ és mantingui no-negatiu.

Fem-ho a R:

```
FA2=factanal(life , factors = 2, method ="mle")
lambda2<-FA2$loading
especificitats2 <-FA2$uniquenesses

p=dim(life)[2]
n=dim(life)[1]
m=2
d=((p-m)^2-(p+m))/2

den=det(cor(life))
num=det(lambda2%*%t(lambda2)+diag(especificitats2))
t1=(n-((2*p+4*m+11)/6))
U2=t1*log(num/den) #valor de l'estadistic

U2
qchisq(.95, df=d)
```

Que ens torna

```
> U2
[1] 45.22979
> qchisq(.95, df=d)
[1] 22.36203
```

Com que $U > \chi^2_{d,\alpha}$ rebutgem $H_0$ i per tant, per aquest nombre de factors (2) el model factorial no és vàlid, necessitem usar més factors.

He creat una funció anomenada `validesafactoritzacio` que ens torna un vector amb el primer element l'estadístic de contrast $U$ i per segon el valor de $\chi^2_{d,\alpha}$:

```
validesafactoritzacio <- function(data,nfactors){
  p=dim(data)[2]
  n=dim(data)[1]
  m=nfactors
  d=((p-m)^2-(p+m))/2

  FA=factanal(life , factors = nfactors , method ="mle")
  lambda<-FA$loading
  especificitats <-FA$uniquenesses

  den=det(cor(life))
  num=det(lambda%*%t(lambda)+diag(especificitats))
  t1=(n-((2*p+4*m+11)/6))
  U=t1*log(num/den)

  CH2=qchisq(0.95, df=d)
  return(c(U,CH2))
}
```

Que ens permet fer els testos explítiament amb rapidesa:

```
validesafactoritzacio(life ,1)
validesafactoritzacio(life ,2)
validesafactoritzacio(life ,3)
```

que ens torna

```
> validesafactoritzacio(life ,1)
[1] 164.17256  31.41043
> validesafactoritzacio(life ,2)
[1] 45.22979 22.36203
> validesafactoritzacio(life ,3)
[1]  8.058792 14.067140
```

On es veu que en el cas de un factor i de dos, $U > \chi^2_{d,\alpha}$ i per tant rebutgem la hipòtesi nul·la, en canvi si usem 3 factors $U < \chi^2_{d,\alpha}$ i per tant no tenim evidència per afirmar que necessitem més factors, podem pensar que són suficients.

**Troba l'estimació de la matriu de pesos o coeficients factorials (factor loadings) $\Lambda$ (loadings) i de les especificitats $E$ (uniquenesses)**.

Recorda que un model factorial s'expresa de forma matricial

$$X = \Lambda F + E$$

```
FA2=factanal(life , factors = 2, method ="mle")
lambda2<-FA2$loading
especificitats2 <-FA2$uniquenesses
```

té com a output:

```
> lambda3
Loadings:
    Factor1 Factor2 Factor3
m0  0.964   0.122   0.226
m25 0.646   0.169   0.438
m50 0.430   0.354   0.790
m75         0.525   0.656
w0  0.970   0.217
w25 0.764   0.556   0.310
w50 0.536   0.729   0.401
w75 0.156   0.867   0.280
               Factor1 Factor2 Factor3
SS loadings      3.375   2.082   1.640
Proportion Var   0.422   0.260   0.205
Cumulative Var   0.422   0.682   0.887
```

```
> especificitats3
        m0         m25        m50        m75         w0        w25        w50
w75
0.00500000 0.36167392 0.06627724 0.28779358 0.00500000 0.01106701 0.02012006 0.14
```

## 9.3 Qüestió 2

**Quina és l'estimació de la covariància de la variable `m25` amb el segon factor? I amb el tercer?**
En el model factorial, els pesos són les covariàncies (correlacions, en cas de dades tipificades) entre les variables i els factors. Es a dir, per a cada parella $(i,k)$, $i = 1,\ldots,p$, $k = 1,\ldots,m$.

$$\mathrm{Cov}(X_i, F_k) = \lambda_{ik} \qquad \mathrm{Cov}(X_i, F_k) = \lambda_{ik} \text{si } X_i \text{ tipificades}$$

Per tant

```
###     Q2    ###
#Cov(X_i,F_k)=lambda_ik
FA3=factanal(life, factors = 3, method ="mle")
lambda3<-FA3$loading

lambda3[2,2] #Cov variable m25 (var 2) amb el 2n factor
lambda3[2,3] #Cov variable m25 (var 3) amb el 3r factor
```

```
> lambda3[2,2] #Cov variable m25 (var 2) amb el 2n factor
[1] 0.1689417
> lambda3[2,3] #Cov variable m25 (var 2) amb el 3r factor
[1] 0.4382963
```

**Quant val l'especificitat per la segona variable?**

```
##     Q3     ###
#Quant val l'especificitat per la segona variable?
FA3=factanal(life, factors = 3, method ="mle")
 especificitats3 <-FA3$uniquenesses
 especificitats3[2] #especificitat per la 2a variable
```

```
> especificitats3[2] #especificitat per la 2a variable
      m25
0.3616739
```

**Quin és el valor de la comunalitat per a la segona variable?**
Les comunalitats $h_i^2$ són els elements de la diagonal de $\Lambda\Lambda^t$

```
###     Q4     ###
#Quin es el valor de la comunalitat per a la segona variable?
comunalitats3<-diag(lambda3%*%t(lambda3))
comunalitats3[2] #valor comunalitat per la 2a variable
```

```
> comunalitats3[2] #valor comunalitat per la 2a variable
      m25
0.6383053
```

**Escriu l'expressió de la segona variable en funció dels factors**

```
###    Q5    ###
#Escriu l'expressi? de la segona variable en funci? dels factors
# X_i=lambda_{i,1}*F_1+\dots + \lambda_{i,m}*F_m + E_i
# X_2=lambda_{2,1}*F_1+lambda_{2,2}*F_2+lambda_{3,2}*F_3+E_2

L21=lambda3[2,1]
L22=lambda3[2,2]
L32=lambda3[2,3]
E2=especificitats3[2]

L21; L22; L32; E2;
```

Ens torna

```
> L21; L22; L32; E2;
[1] 0.6462665
[1] 0.1689417
[1] 0.4382963
      m25
0.3616739
```

Per tant l'expressió de la segona variable en funció dels factors és

$$X_2 = 0.6462665F1 + 0.1689417F2 + 0.4382963F3 + 0.3616739$$

**Fes els contrastos anteriors sortint de la matriu de covariàncies i comprova que surt el mateix resultat qeu abans**

```
###    Q6    ###
#Fes els contrastos anteriors sortint de la matriu de covari?ncies i comprova qu
FA33=factanal(covmat=cov(life),n.obs=n, factors=3)
n=dim(life)[1]
sapply(1:3, function(f)
  factanal(covmat=cov(life), factors = f, method ="mle",n.obs=n)$PVAL)

#comprovem que ?s igual que FA3:
lambda33<-FA33$loading
especificitats33 <-FA33$uniquenesses
lambda3
lambda33
especificitats3
especificitats33
```

que ens torna

```
> sapply(1:3, function(f)
+ factanal(covmat=cov(life), factors = f, method ="mle",n.obs=n)$PVAL)
   objective     objective     objective
1.879555e-24 1.911514e-05 4.578204e-01
```

i la mateixa matriu de pesos i especificitats.

**Comprova si $\hat{\Lambda}\hat{\Lambda}' + \hat{\Psi}$ és proper a $S$.** Per a fer-ho calculem la component més gran en valor absolut de la diferència:

```
###   Q7   ###

FA3=factanal(life ,factors=3,method="mle")
lamb3=FA3$loadings
S.estimada3=lambda3%*%t(lambda3)+diag(FA3$uniquenesses)
#recorda que cal posar diag, imprescindicble
CRM=cor(life)
max(abs(S.estimada3-CRM))
```

Que ens torna

```
> max(abs(S.estimada3-CRM))
[1] 0.05936939
```

El model és bo ja que les matrius són similars.

**Defineix una funció que em doni la matriu diferència entre $S$ i $\hat{\Lambda}\hat{\Lambda}' + \hat{\Psi}$.**

```
funcio.MatError<-function(dades,nf){
  options(digits=10)
  n=dim(dades)[1]
  FA=factanal(dades,factors=nf,method="mle")
  lamb=FA$loadings
  S.estimada=lamb%*%t(lamb)+diag(FA$uniquenesses)
  CRM=cor(dades)
  return(S.estimada-CRM)
}
```

**Adapta la funció anterior per a que em doni el màxim dels valors absoluts dels elements de la matriu diferència.**

```
funcio.MaxError<-function(dades,nf){
  M=funcio.MatError(dades,nf)
  return(max(abs(M)))
}
```

Podem executar la funció

```
round(funcio.MatError(life ,3),3)
funcio.MaxError(life ,3)
```

que ens torna

```
> round(funcio.MatError(life,3),3)
        m0     m25     m50     m75     w0    w25    w50    w75
m0   0.000  -0.006   0.000  -0.001  0.000  0.000 -0.001  0.002
m25 -0.006   0.000   0.017   0.037  0.006 -0.001 -0.002 -0.023
m50  0.000   0.017   0.000  -0.014  0.000 -0.002  0.003  0.002
m75 -0.001   0.037  -0.014   0.000 -0.003  0.009  0.002 -0.059
w0   0.000   0.006   0.000  -0.003  0.000  0.000  0.001 -0.003
w25  0.000  -0.001  -0.002   0.009  0.000  0.000 -0.001  0.004
w50 -0.001  -0.002   0.003   0.002  0.001 -0.001  0.000  0.000
w75  0.002  -0.023   0.002  -0.059 -0.003  0.004  0.000  0.000
> funcio.MaxError(life,3)
[1] 0.0593693875
```

## 9.4 Qüestió 4

**Calcula la matriu de pesos sense rotació i també aplicant la rotació donada pel mètode varimax.**

Calculem $\Lambda$, avans de rotar i dreprés:

```
#sense rotar
FA3=factanal(life, factors=3,method="mle")
lambda=FA3$loadings
#rotant
lambda.nou=varimax(loadings(FA3), normalize = FALSE)$loadings
```

```
lambda.nou
lambda
```

les pots veure a la pàgina següent:

```
> lambda.nou

 Loadings:
    Factor1 Factor2 Factor3
m0  0.960   0.125   0.240
m25 0.639   0.188   0.442
m50 0.417   0.396   0.777
m75         0.562   0.626
w0  0.971   0.212
w25 0.764   0.566   0.293
w50 0.535   0.747   0.369
w75 0.160   0.880   0.232

               Factor1 Factor2 Factor3
SS loadings      3.346   2.220   1.531
Proportion Var   0.418   0.278   0.191
Cumulative Var   0.418   0.696   0.887
> lambda

 Loadings:
    Factor1 Factor2 Factor3
m0  0.964   0.122   0.226
m25 0.646   0.169   0.438
m50 0.430   0.354   0.790
m75         0.525   0.656
w0  0.970   0.217
w25 0.764   0.556   0.310
w50 0.536   0.729   0.401
w75 0.156   0.867   0.280

               Factor1 Factor2 Factor3
SS loadings      3.375   2.082   1.640
Proportion Var   0.422   0.260   0.205
Cumulative Var   0.422   0.682   0.887
```

**Comprara les comunalitats i especificitats corresponents (abans de rotar i després)**
Les comunalitats i especificitats són les mateixes sense rotar i rotant (és així per a qualsevol rotació).

```
comunalitats.noves=diag(lambda.nou%*%t(lambda.nou))
comunalitats.noves #son iguals
comunalitats3 #son iguals

Identity<-diag(matrix(rep(1,8),nrow=8,ncol=8))
especificitats.noves=Identity-comunalitats.noves

round(especificitats3,3) #son iguals
round(especificitats.noves,3) #son iguals
```

## 9.5 Codi complert

```
"life" <- structure(.Data = list(c(63., 34., 38., 59., 56., 62.,
                                  50., 65., 56., 69., 65., 64., 56., 60., 61.,
                                  65., 65., 64.,
                                  64., 67., 61., 68., 67., 65., 59., 58., 57.)
                                , c(51., 29., 30., 42., 38., 44., 39., 44., 46.
                                    44., 45., 40., 42., 44., 44., 48., 48., 63.
                                    43., 45., 40., 46., 45., 46., 43., 44., 46.
                                , c(30., 13., 17., 20., 18., 24., 20., 22., 24.
                                    22., 22., 22., 22., 23., 24., 28., 26., 21.
                                    21., 23., 21., 23., 23., 24., 23., 24., 28.
                                , c(13., 5., 7., 6., 7., 7., 7., 7., 11., 8., 9
                                    9., 6., 8., 8., 14., 9., 7., 6., 8., 10., 8
                                    8., 9., 10., 9., 9.)
                                , c(67., 38., 38., 64., 62., 69., 55., 72., 63.
                                    65., 65., 51., 61., 67., 63., 68., 67., 68.
                                    68., 74., 67., 75., 74., 71., 66., 62., 60.
                                , c(54., 32., 34., 46., 46., 50., 43., 50., 54.
                                    45., 49., 41., 43., 48., 46., 51., 49., 47.
                                    47., 51., 46., 52., 51., 51., 49., 47., 49.
                                , c(34., 17., 20., 25., 25., 28., 23., 27., 33.
                                    25., 27., 23., 22., 26., 25., 29., 27., 25.
                                    24., 28., 25., 29., 28., 28., 27., 25., 28.
                                , c(15., 6., 7., 8., 10., 14., 8., 9., 19., 10.
                                    10., 8., 7., 9., 8., 13., 10., 9., 8., 10.,
                                    10., 10., 10., 12., 10., 11.)
) , class = "data.frame" , names = c("m0", "m25", "m50", "m75",
                                    "w0", "w25", "w50", "w75") , row.names = c(









### Q1 ###

sapply(1:3, function(f)
  factanal(life , factors = f, method ="mle")$PVAL)
help(factanal)
FA2=factanal(life , factors = 2, method ="mle")
```

```
lambda2<-FA2$loading
especificitats2 <-FA2$uniquenesses
lambda2
especificitats2

p=dim(life)[2]
n=dim(life)[1]
m=2
d=((p-m)^2-(p+m))/2

den=det(cor(life))
num=det(lambda2%*%t(lambda2)+diag(especificitats2))
t1=(n-((2*p+4*m+11)/6))
U2=t1*log(num/den) #valor de l'estadistic

U2
qchisq(.95, df=d)

validesafactoritzacio <- function(data,nfactors){
  p=dim(data)[2]
  n=dim(data)[1]
  m=nfactors
  d=((p-m)^2-(p+m))/2

  FA=factanal(life, factors = nfactors, method ="mle")
  lambda<-FA$loading
  especificitats <-FA$uniquenesses

  den=det(cor(life))
  num=det(lambda%*%t(lambda)+diag(especificitats))
  t1=(n-((2*p+4*m+11)/6))
  U=t1*log(num/den)

  CH2=qchisq(0.95, df=d)
  return(c(U,CH2))
}
validesafactoritzacio(life,1)
validesafactoritzacio(life,2)
validesafactoritzacio(life,3)
factanal(life, factors = 3, method ="mle")


FA3=factanal(life, factors = 3, method ="mle")
lambda3<-FA3$loading
especificitats3 <-FA3$uniquenesses
lambda3
especificitats3
```

```
###     Q2    ###

#Cov(X_i,F_k)=lambda_ik
FA3=factanal(life, factors = 3, method ="mle")
lambda3<-FA3$loading

lambda3[2,2] #Cov variable m25 (var 2) amb el 2n factor
lambda3[2,3] #Cov variable m25 (var 2) amb el 3r factor

###     Q3     ###
#Quant val l'especificitat per la segona variable?
FA3=factanal(life, factors = 3, method ="mle")
especificitats3<-FA3$uniquenesses
especificitats3[2] #especificitat per la 2a variable

###     Q4     ###
#Quin ?s el valor de la comunalitat per a la segona variable?
comunalitats3<-diag(lambda3%*%t(lambda3))
comunalitats3[2] #valor comunalitat per la 2a variable


#compovaci?: (la suma de comunalitats i especificitats sumen 1)
round(comunalitats3+especificitats3,3)

###     Q5     ###
#Escriu l'expressi? de la segona variable en funci? dels factors
# X_i=lambda_{i,1}*F_1+\dots + \lambda_{i,m}*F_m + E_i
# X_2=lambda_{2,1}*F_1+lambda_{2,2}*F_2+lambda_{3,2}*F_3+E_2

L21=lambda3[2,1]
L22=lambda3[2,2]
L32=lambda3[2,3]
E2=especificitats3[2]

L21; L22; L32; E2;

###     Q6     ###
#Fes els contrastos anteriors sortint de la matriu de covari?ncies i comprova qu
FA33=factanal(covmat=cov(life),n.obs=n,factors=3)
n=dim(life)[1]
sapply(1:3, function(f)
  factanal(covmat=cov(life), factors = f, method ="mle",n.obs=n)$PVAL)

#comprovem que ?s igual que FA3:
```

```
lambda33<-FA33$loading
especificitats33<-FA33$uniquenesses
lambda3
lambda33
especificitats3
especificitats33



###   Q7   ###

FA3=factanal(life, factors=3,method="mle")
lamb3=FA3$loadings
S.estimada3=lambda3%*%t(lambda3)+diag(FA3$uniquenesses)
#recorda que cal posar diag, imprescindicble
CRM=cor(life)
max(abs(S.estimada3-CRM))

#el model factorial ser? bo si les dues matrius s?n similars

funcio.MatError<-function(dades,nf){
   options(digits=10)
   n=dim(dades)[1]
   FA=factanal(dades, factors=nf,method="mle")
   lamb=FA$loadings
   S.estimada=lamb%*%t(lamb)+diag(FA$uniquenesses)
   CRM=cor(dades)
   return(S.estimada-CRM)
}
funcio.MaxError<-function(dades,nf){
  M=funcio.MatError(dades,nf)
   return(max(abs(M)))
}

round(funcio.MatError(life,3),3)
funcio.MaxError(life,3)



###   Q8   ###
#Primer calculem sense rotar:
FA3=factanal(life, factors=3,method="mle")
lambda=FA3$loadings
comunalitats3<-diag(lambda3%*%t(lambda3))
especificitats3<-FA3$uniquenesses

round(comunalitats3+especificitats3,3) #cuadra (sumen 1)
#ara rotem
```

```
lambda.nou=varimax(loadings(FA3), normalize = FALSE)$loadings
lambda.nou
lambda
#Amb el metode varimax el que fem es maximitzar les variancies
#dels quadrats dels pesos per columnes.
#Si la variancia de la columna j es gran, el corresponent factor Fj
#pesos grans en valor absolut en unes variables i petits en altres, que ?s el qu

comunalitats.noves=diag(lambda.nou%*%t(lambda.nou))
comunalitats.noves #son iguals
comunalitats3 #son iguals

Identity<-diag(matrix(rep(1,8),nrow=8,ncol=8))
especificitats.noves=Identity-comunalitats.noves

round(especificitats3,3) #son iguals
round(especificitats.noves,3) #son igualss
```

# 10 Pràctica 10

## 10.1 Introducció

Usem les mateixes dades que a la pràctica anterior:

```
"life" <- structure(.Data = list(c(63., 34., 38., 59., 56., 62.,
50., 65., 56., 69., 65., 64., 56., 60., 61., 49., 59., 63., 59.,
65., 65., 64.,
64., 67., 61., 68., 67., 65., 59., 58., 57.)
, c(51., 29., 30., 42., 38., 44., 39., 44., 46., 47., 48., 50., 44.,
44., 45., 40., 42., 44., 44., 48., 48., 63.,
43., 45., 40., 46., 45., 46., 43., 44., 46.)
, c(30., 13., 17., 20., 18., 24., 20., 22., 24., 24., 26., 28., 25.,
22., 22., 22., 22., 23., 24., 28., 26., 21.,
21., 23., 21., 23., 23., 24., 23., 24., 28.)
, c(13., 5., 7., 6., 7., 7., 7., 7., 11., 8., 9., 11., 10., 6., 8.,
9., 6., 8., 8., 14., 9., 7., 6., 8., 10., 8.,
8., 9., 10., 9., 9.)
, c(67., 38., 38., 64., 62., 69., 55., 72., 63., 75., 68., 66., 61.,
65., 65., 51., 61., 67., 63., 68., 67., 68.,
68., 74., 67., 75., 74., 71., 66., 62., 60.)
, c(54., 32., 34., 46., 46., 50., 43., 50., 54., 53., 50., 51., 48.,
45., 49., 41., 43., 48., 46., 51., 49., 47.,
47., 51., 46., 52., 51., 51., 49., 47., 49.)
, c(34., 17., 20., 25., 25., 28., 23., 27., 33., 29., 27., 29., 27.,
25., 27., 23., 22., 26., 25., 29., 27., 25.,
24., 28., 25., 29., 28., 28., 27., 25., 28.)
, c(15., 6., 7., 8., 10., 14., 8., 9., 19., 10., 10., 11., 12., 9.,
10., 8., 7., 9., 8., 13., 10., 9., 8., 10., 11.,
10., 10., 10., 12., 10., 11.)
) , class = "data.frame" , names = c("m0", "m25", "m50", "m75",
"w0", "w25", "w50", "w75") , row.names = c("Algeria", "Cameroon",
"Madagascar", "Mauritius", "Reunion", "Seychelles", "South Africa
(C)", "South Africa (W)",
"Tunisia", "Canada", "Costa Rica", "Dominican Rep.", "El Salvador",
"Greenland", "Grenada", "Guatemala",
"Honduras", "Jamaica", "Mexico", "Nicaragua", "Panama",
"Trinidad (62)", "Trinidad (67)",
"United States (66)", "United States (NW66)", "United States (W66)",
"United States (67)", "Argentina",
"Chile", "Colombia", "Ecuador"))
```

Utilitzàvem la comanda `factanal` per a fer anàlisi factorial. Si mireu l'ajuda de R veureu que aquesta comanda tan sols utilitza le mètode de màxima versemblança. Si voleu emprar el mètode de components principals, una possiblitat que hem trobat a la web `Quick R` és executar les següents instruccions (observeu que hem de carregar la llibreria `psych`).

```
pc<-principal(dades o una matriu de covariancies,
```

```
nombre de factors , rotate=metode de rotacio)
```

la qual encara que li donem una matriu de covariàncies sortirà de la matriu de correlacions. Per a sortir de la matriu de covariàncies hem d'escriure `covar=TRUE`. Pel que fa al mètode de rotació, si li posem `none`, no s'aplicara cap.

Si volem emprar el mètode de factors principals, una possibilitat que hem trobat a la web `Quick R` són les següents comandes (les quals també estan a la llibreria `psych`).

```
library(psych)
fit<-fa(nom de les dades, nfactors=, rotation="metode de rotacio")
fit #print results
```

Observa que a l'anàlisi factorial podem efectuar rotacions de la matriu $\hat{\Lambda}$ aplicant diferents procediments. També a la instrucció `factanal` podíem triar el mètode de determinació de la rotació adecuada.

Mira el help d'aquestes comandes de *R*.

## 10.2 Qüestió 1

**Trobeu l'estimació de la matriu $\Lambda$ i de les especificitats pel mètode de les components principals per a les dades anteriors (no apliquis rotacions).**
Primer recordem la forma matricial del model factorial:

$$X = \Lambda F + E$$

on $X$ és el vector aleatori columna $p \times 1$ inicial, $F$ és un fector aleatori columna $m \times 1$ que conté els $F_1, \dots, F_m$ anomenats **factors comuns**, que explicaran les correlacions entre les varialbes, $\Lambda$ és una matriu $p \times m$ de coeficients escalars, anomenats **pesos o coeficients factorials** (factor loadings), que denotem $\lambda_{i,j}$, $i = 1, \dots, m$, $j = 1, \dots, p$. $E$ és un vector aleatori columna $p \times 1$ que conté $E_1, \dots, E_p$ variables anomenades **factors específics** que explicaran la part residual de variabilitat de les variables.
NOTA: No hem de confondre la notació dels pesos amb els $'\lambda_i'$ que deontaven els valors propis de l'ACP.
El model factorial descrit per cada una de les variables és

$$X_i = \lambda_{i1}F_1 + \lambda_{i2}F_2 + \dots + \lambda_{im}F_m + E_i, \ i = 1 \dots p.$$

Fem-ho a *R*, nota però que a la pràctica anterior vem veure que usar 3 compornents prinipals és suficient per aquestes dades.

```
library(psych)

pc=principal(life,3,rotate="none")
lambda = pc$loadings
especificitat=pc$uniquenesses

lambda
especificitats
```

**Fes el mateix programant tots els passos a partir de les instruccions que coneixes del mètode de components principals.**
Per a trobar $\hat{\Lambda}$ usant el mètode de components principals, usem $\hat{\Lambda} = VD$, on $V$ és la matriu ortogonal de vectors propis unitaris columna $v_j$ i $D^2$ és la matriu diagonal dels valors propis ordenats, que denotem $\theta_1 \dots \theta_p$, que surten de la diagonalització de la matriu de correlacions mostrals: $R = VD^2V^t$

nota que escollim les $m$ primeres columnes d'aquesta matriu, en el nostre cas $m = 3$.

```
life.stand<-scale(life)
veps=prcomp(life.stand)$rotation
vaps=(prcomp(life.stand)$sdev^2)

lambda.pc=veps[,1:3]%*%diag(sqrt(vaps[1:3]),3)
veps[,1:3]
diag(sqrt(vaps[1:3]),3)
#factanal sempre surt de correlacions

lambda.pc
lambda
```

Nota que $\Lambda$ calculada directament `lambda` i la calculada amb tots els passos `lambda.pc` són la mateixa, encara que les seves columnes poden tenir signes diferents, degut al signe arbitrari d'elecció dels veps.

Falta calcular l'estimació de les especificitats. La matriu producte $\hat{\Lambda}\hat{\Lambda}^t$ s'anomena matriu de correlacions **reproduida**, les **comunalitats** s'estimen com les diagnoals de la matriu producte de pesos factorials :

$$\hat{h_i^2} = [\hat{\Lambda}\hat{\Lambda}^t]_{ii}$$

L'estimació de les **especificitats** es defineix de forma que a la diagnoal de $R$ hi hagi igualtat, és a dir

$$\hat{\psi}_i = 1 - \hat{h_i}^2$$

```
especificiatas.pc=1-diag(lambda.pc%*%t(lambda.pc))
especificitats
especificiatas.pc
```

Nota que aquí les estimacions de les especificitats forçosament han de ser iguals.

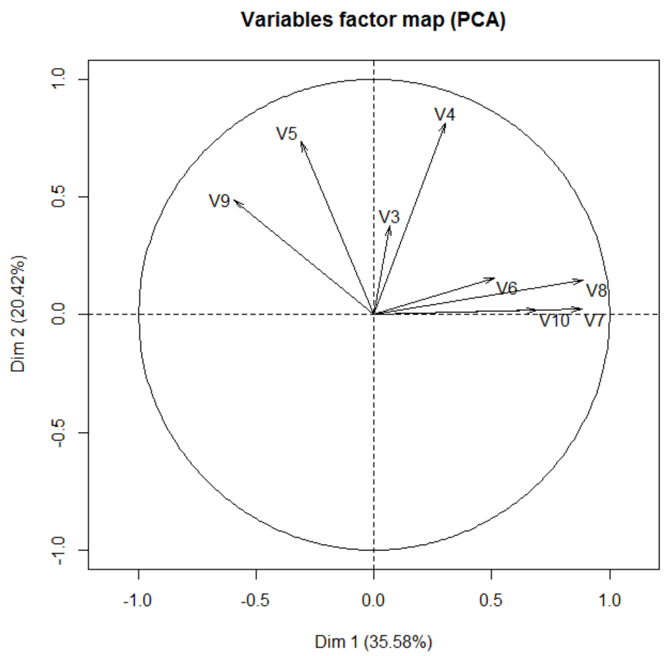
**Aplica ara una rotació escollida pel mètode varimax (recorda que a la pràctica anterior utilitzàrem la comanda `varimax`.**

```
pc.rot=principal(life ,nfactors=3,rotate="varimax",residuals=FALSE)
lambda.rot=pc.rot$loadings
respecificitat.rot=pc.rot$uniquenesses
```

### 10.3 Qüestió 2

**Calcula, a partir de les estiamcions per màxima versemblança, la puntuació factorial dels individus de la matriu de dades i fes el plot del primer factor contra el segon, i del segon contra el tercer, indicant en el gràfic la inicial del país.**

```
pc.rot=principal(life ,nfactors=3,rotate="varimax",residuals=FALSE)
lambda.rot=pc.rot$loadings
respecificitat.rot=pc.rot$uniquenesses

PROC=factanal(life ,factors=3,scores="regression")
scores=as.matrix(PROC$scores)

par(mfrow=c(1,2))
plot(scores[,1],scores[,2])
text(x=scores[,1],y=scores[,2],substr(rownames(life),1,1),
cex=0.8,pos=1,col=c("red","black"))

plot(scores[,2],scores[,3])
text(x=scores[,2],y=scores[,3],substr(rownames(life),1,1),
cex=0.8,pos=1,col=c("red","black"))
```

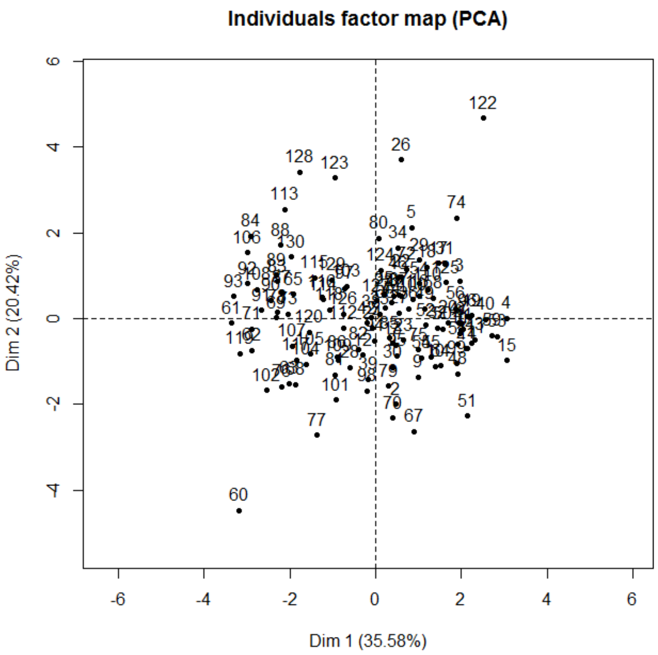


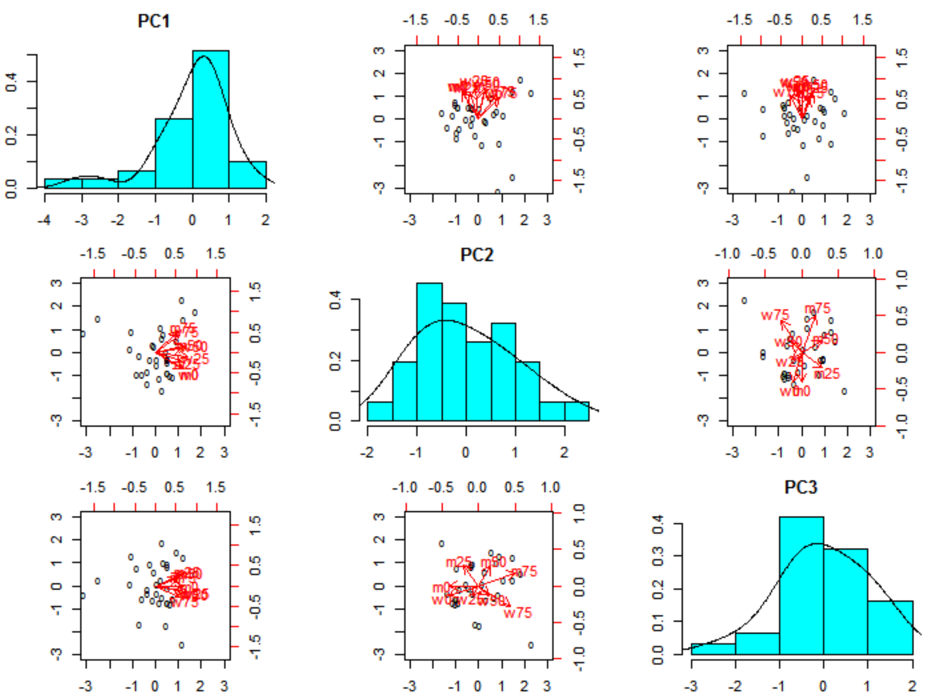

Recorda que si fem servir totes les coordenades, la distància euclídea entre els punts dels individus està lligada amb la de Mahalanobis entre les dades originals, i els angles entre variables a les seves correlacions. Però això no serà en el biplot ja que ens reduïm a dimensió 2. (Observa que a la comanda li has de posar el nom del procediment de components principals fet abans, no el nom de l'arxiu de dades.).

### 10.3.1 Qüestió 3

**Fes el biplot corresponent a la matriu de dades inicial, interpretant els punts associats als individus i els associats a les variables.**

```
fit <- princomp(life , cor=TRUE)
biplot(fit)

biplot(pc,3,rotate="none")
```



# 11 Pràctica 11

## 11.1 Introducció

Pels 50 estats d'USA tenim el `ratio` per 100000 habitants per a 7 tipus d'actes delictius,

```
crime <- structure(c(2, 2.2, 2, 3.6, 3.5, 4.6, 10.7, 5.2, 5.5,
                      5.5, 6, 8.9, 11.3, 3.1, 2.5, 1.8, 9.2, 1, 4, 3.1, 4.4, 4.
                      7.1, 5.9, 8.1, 8.6, 11.2, 11.7, 6.7, 10.4, 10.1, 11.2, 8.
                      8.1, 13.5, 2.9, 3.2, 5.3, 7, 11.5, 9.3, 3.2, 12.6, 5, 6.6
                      8.6, 4.8, 14.8, 21.5, 21.8, 29.7, 21.4, 23.8, 30.5, 33.2,
                      38.6, 25.9, 32.4, 67.4, 20.1, 31.8, 12.5, 29.2, 11.6, 17.
                      32.9, 56.9, 43.6, 52.4, 26.5, 18.9, 26.4, 41.3, 43.9, 52.
                      47, 28.4, 25.8, 28.9, 40.1, 36.4, 51.6, 17.3, 20, 21.9, 4
                      43, 25.3, 64.9, 53.4, 51.1, 44.9, 72.7, 31, 28, 24, 22, 1
                      192, 514, 269, 152, 142, 90, 325, 301, 73, 102, 42, 170,
                      80, 124, 304, 754, 106, 41, 88, 99, 214, 367, 83, 208, 11
                      224, 107, 240, 20, 21, 22, 145, 130, 169, 59, 287, 135, 2
                      88, 106, 102, 92, 103, 331, 192, 205, 431, 265, 176, 235,
                      424, 162, 148, 179, 370, 32, 87, 184, 252, 241, 476, 668,
                      354, 525, 319, 605, 222, 274, 408, 172, 278, 482, 285, 35
                      178, 243, 329, 538, 437, 180, 354, 244, 286, 521, 401, 10
                      755, 949, 1071, 1294, 1198, 1221, 1071, 735, 988, 887, 11
                      783, 1004, 956, 1136, 385, 554, 748, 1188, 1042, 1296, 17
                      625, 1225, 1340, 1453, 2221, 824, 1325, 1159, 1076, 1030,
                      1787, 2049, 783, 1003, 817, 1792, 1845, 1908, 915, 1604,
                      1696, 1162, 1339, 2347, 2208, 2697, 2189, 2568, 2758, 292
                      1654, 2574, 2333, 2938, 3378, 2802, 2785, 2801, 2500, 204
                      2677, 3008, 3090, 2978, 4131, 2522, 1358, 2423, 2846, 298
                      1740, 2126, 2304, 1845, 2305, 3417, 3142, 3987, 3314, 280
                      4231, 3712, 4337, 4074, 3489, 4267, 4163, 3384, 3910, 375
                      228, 181, 906, 705, 447, 637, 776, 354, 376, 328, 628, 80
                      288, 158, 439, 120, 99, 168, 258, 272, 545, 975, 219, 169
                      430, 598, 193, 544, 267, 150, 195, 442, 649, 714, 215, 18
                      486, 343, 419, 223, 478, 315, 402, 762, 604, 328), .Dim =
                      ), .Dimnames = list(c("ME", "NH", "VT", "MA", "RI", "CT",
                                            "NJ", "PA", "OH", "IN", "IL", "MI",
                                            "SD", "NE", "KS", "DE", "MD", "DC",
                                            "FL", "KY", "TN", "AL", "MS", "AR",
                                            "WY", "CO", "NM", "AZ", "UT", "NV",
                                          c("Murder", "Rape", "Robbery", "Assau
                                            "Vehicle")))
```

Fixem-nos que el que obtens no és un data.frame (pots veure-ho amb la comanda `is.data.frame(nom)`). Aquestes dades les podem convertir en un `data frame` amb)

```
crime<-as.data.frame(nom)
```

## 11.2 Qüestió 1

**Considera els 6 primers estats i troba la matriu de distàncies euclídees amb la comanda `dist`.**

```
crime; crime[1:6,]; #considerem els 6 primers estats
(d=dist(crime[1:6,]));
```

Tenim per output:

```
         ME       NH       VT       MA       RI
NH 160.8786
VT 379.7249 528.2908
MA 852.7852 803.5249 938.0101
RI 773.9702 816.5027 653.8917 508.5636
CT 665.3028 766.6251 419.2765 752.6512 342.6149
```

**Quina és la distància entre NH i VT?** La distància entre NH i VT és 528.2908

**Fes a ma el dendograma corresponent a aquests 6 individus utilitzan la distància al veí més llunya (complet linkage).**
PAS 1:

```
(d=dist(crime[1:6,]));
range(d)[1] #La distancia minima es 160.8786
D=as.matrix(d)
D[2,1]  #la distancia minima es correspon a les components 2,1
        #es a dir, els individus mes propers son el ME i el NH
        #per tant ajuntem els individus ME i NH
```

La matriu de distàncies ens queda: (agafem la distància màxima)

```
> D2
         [,1]     [,2]     [,3]     [,4]     [,5]
[1,]   0.0000 528.2908 852.7852 816.5027 766.6251
[2,] 528.2908   0.0000 938.0101 653.8917 419.2765
[3,] 852.7852 938.0101   0.0000 508.5636 752.6512
[4,] 816.5027 653.8917 508.5636   0.0000 342.6149
[5,] 766.6251 419.2765 752.6512 342.6149   0.0000
```

PAS 2:

```
min(D2[D2>0]) #la distancia minima es 342.6149
D2[4,5]   #la distancia minima es correspon a les components 5,6
          #es a dir, els individus mes propers son el RI i CT
```

```
  La nova matriu de distancies ens queda
  > D3
         [,1]     [,2]     [,3]     [,4]
[1,]   0.0000 528.2908 852.7852 816.5027
[2,] 528.2908   0.0000 938.0101 653.8917
[3,] 852.7852 938.0101   0.0000 752.6512
[4,] 816.5027 653.8917 752.6512   0.0000
```

PAS 3:

```
  min(D3[D3>0]) #la distancia minima es 528.2908
D3[2,1]   #la distancia minima es correspon a les components 2,1
#es a dir, els individus mes propers son el VT i el primer cluster
```

La matriu de distàncies ens queda

```
  > D4
         [,1]     [,2]     [,3]
[1,]   0.0000 938.0101 816.5027
[2,] 938.0101   0.0000 752.6512
[3,] 816.5027 752.6512   0.0000
```

PAS 4:

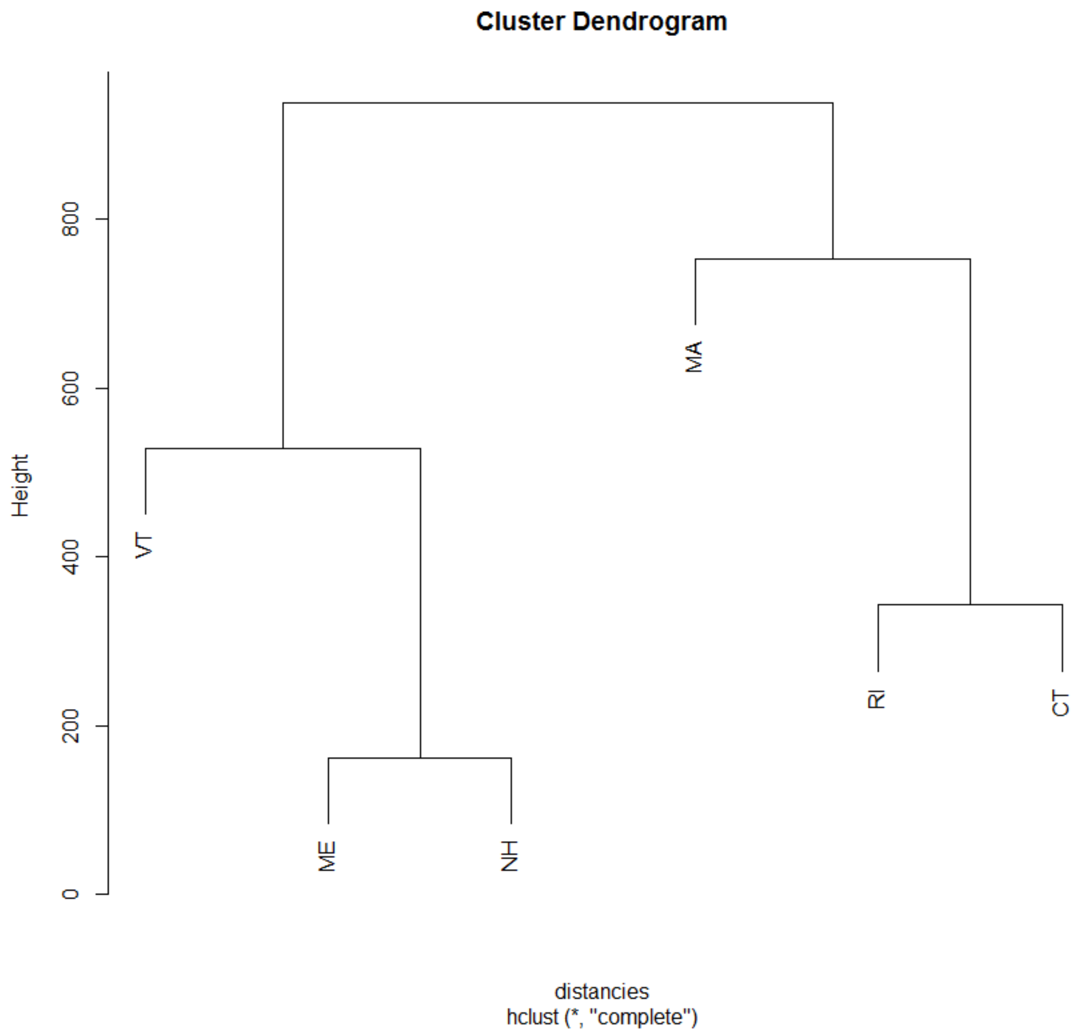
```
  min(D4[D4>0]) #la distancia minima es 752.6512
              #la distancia minima es correspon a les components 2,3
              #es a dir, les components mes properes son la 2 i 3
```

La matriu de distàncies ens queda

```
   D5
         [,1]     [,2]
[1,]   0.0000 938.0101
[2,] 938.0101   0.0000
```

Si volem fer el dendograma atomàticament ho podem fer així:

```
dades=crime[1:6,]
distancies<-dist(dades)
cs<-hclust(distancies,method="complete")
plot(cs)
```




recorda que hem usat *complete linkage* (distàncies al veí més llunyà).

**A quina distància es fa la tercera associació?** La tercera associació es fa a distància 528.2908

## 11.3 Qüestió 2

**Aplica les comandes:**

```
nom2 <- sapply(nom1, function(x) diff(range(x)))
nom3 <- sweep(nom1,2, nom2, FUN = "/")
```

**a les dades `crime` i explica què és el que obtens.**

Per aplicar-les cal fer:

```
crime2<-as.data.frame(crime)
crime3<-sapply(crime2, function(x) diff(range(x)))
crime4<-sweep(crime,2,crime3, FUN = "/")
```

L'objectiu d'aquestes comanes es crear un nou dataframe a partir del original, tranformant-lo amb l'objectiu de tenir variables amb variància similar.

La primera comanda (crime3), calcula per a cada columna la diferència entre el valor màxim i el mínim (el rang de cada columna).
La segona comanda (crime4) divideix cada columna del dataframe per el seu rang.

**Quines seran les variàncies de les dades transformades?**

```
#Calculem ara les variancies de les dades transformades
variancies<-apply(crime4,2,var) #2 in dica columna
variancies
```

Que ens torna

```
> variancies
    Murder       Rape    Robbery    Assault   Burglary      Theft    Vehicle
0.02578017 0.05687124 0.03403775 0.05439933 0.05277909 0.06411424 0.06516672
```

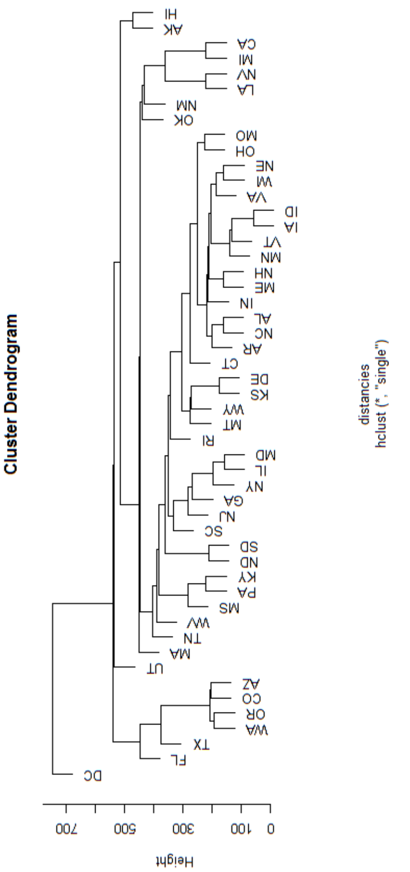
## 11.4 Qüestió 3

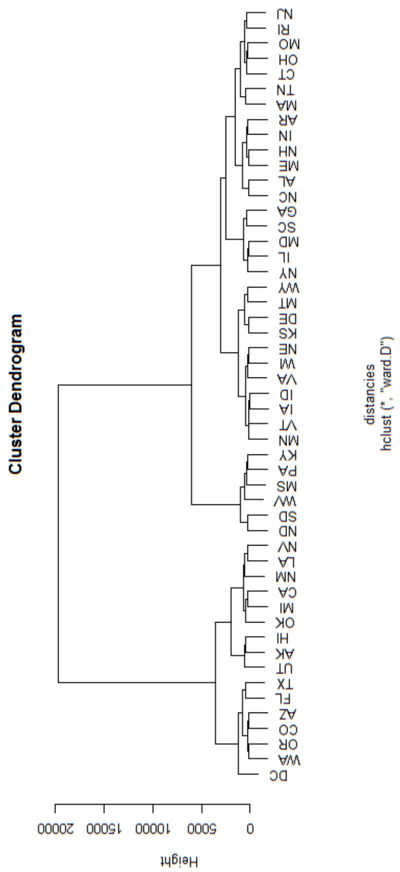
Construeix el dendograma de les dades originals amb la comanda

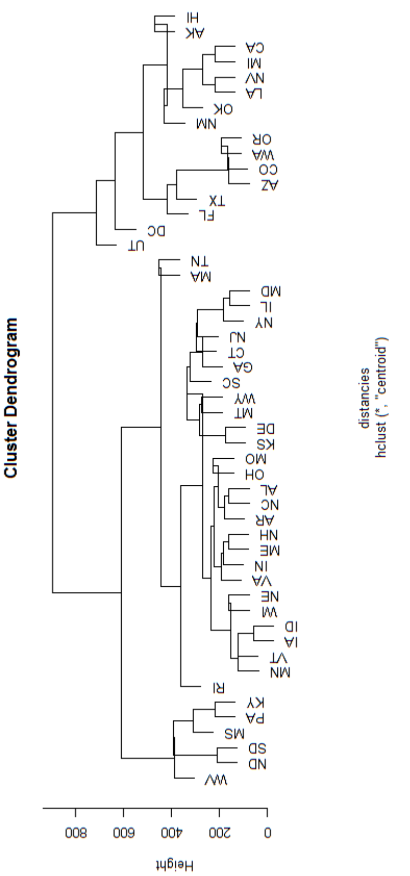
```
cs <- hclust(nom, method = )
plot(cs )
```

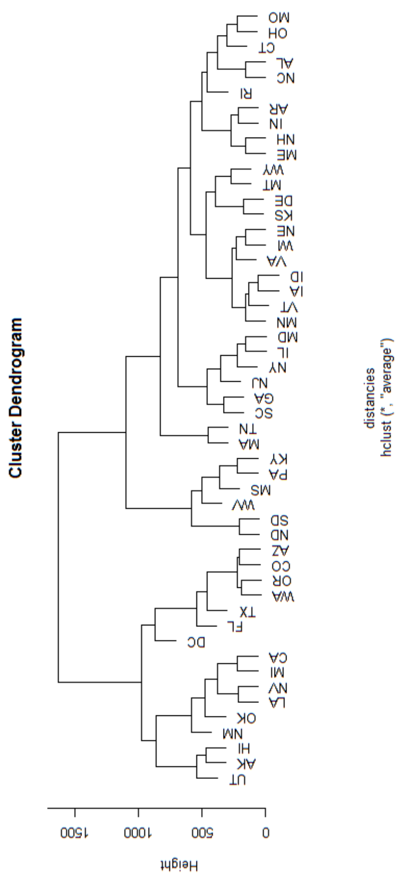
on `method` indica quina distància entre els clusters fem servir: `single`, `complete`, `average`, `mètode de Ward`.

```
### Questio 3 ###
#Volem consturir el dendograma de les dades originals.
#-> primer hem de calcular la matriu de dissimilaritats ,
#en aquest cas, utilitzarem la distancia euciliana:
#podem usar molts metodes
distancies<-dist(crime)
par(mfrow=c(2,2))
cs<-hclust(distancies,method="centroid"); plot(cs)
cs<-hclust(distancies,method="single"); plot(cs)
cs<-hclust(distancies,method="average"); plot(cs)
cs<-hclust(distancies,method="ward"); plot(cs)
```


Cluster Dendrogram
Height
distancies
hclust (*, "single")


Cluster Dendrogram
Height
distancies
hclust (*, "ward.D")

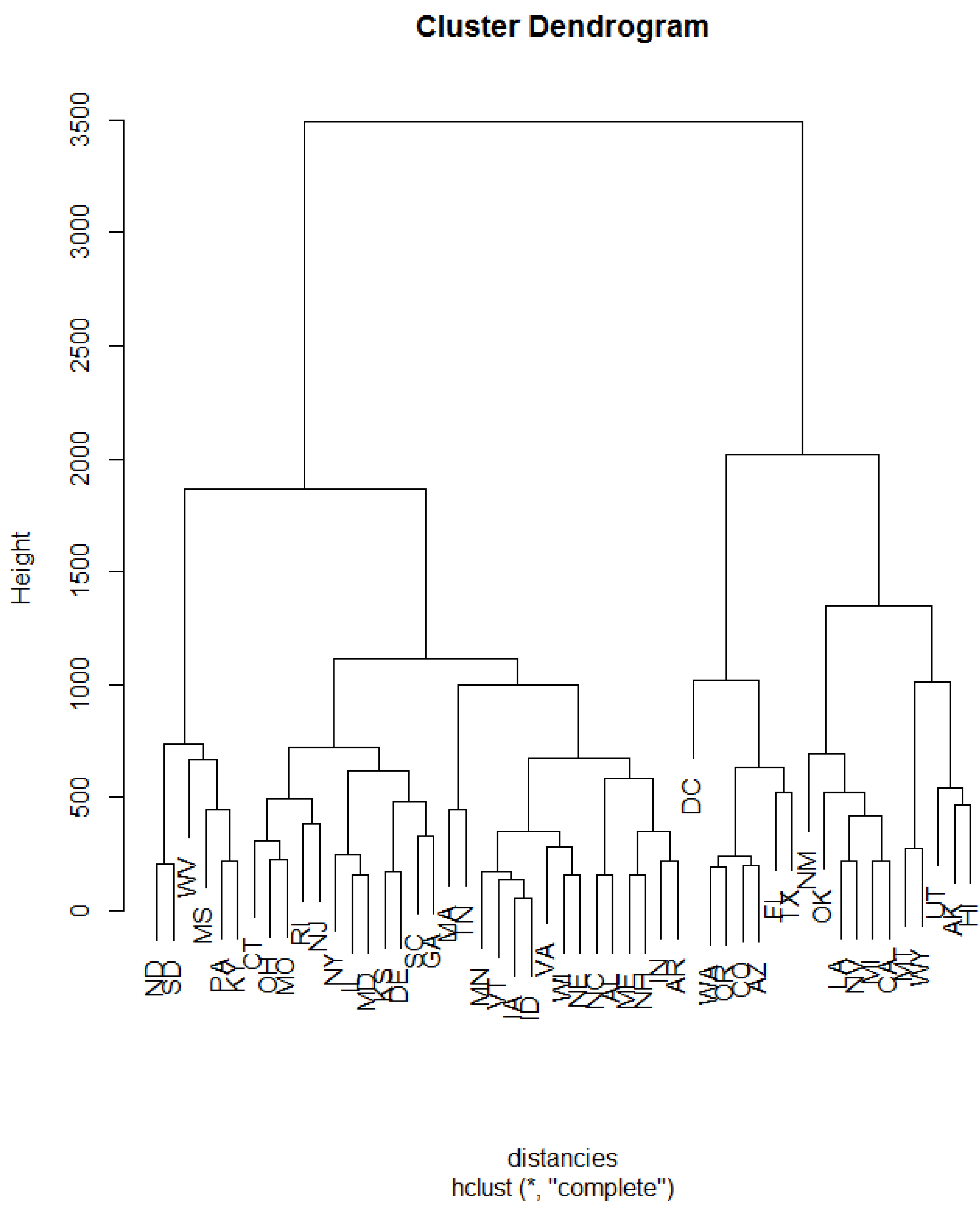

Cluster Dendrogram
Height
distancies
hclust (*, "centroid")


Cluster Dendrogram
Height
distancies
hclust (*, "average")

**Usant el mètode *complete linkage*:**

**Si decideixes quedar-te amb 4 clusters, quins serien?** Primer calculem i mirem l'arbre:

```
cs<-hclust(distancies,method="complete"); plot(cs)
```




Si volem tallar l'arbre en 4, els grups ens queden de la següent forma:

```
cutree(cs,k=4)
```

Que ens torna a quin grup correspon cada individu.

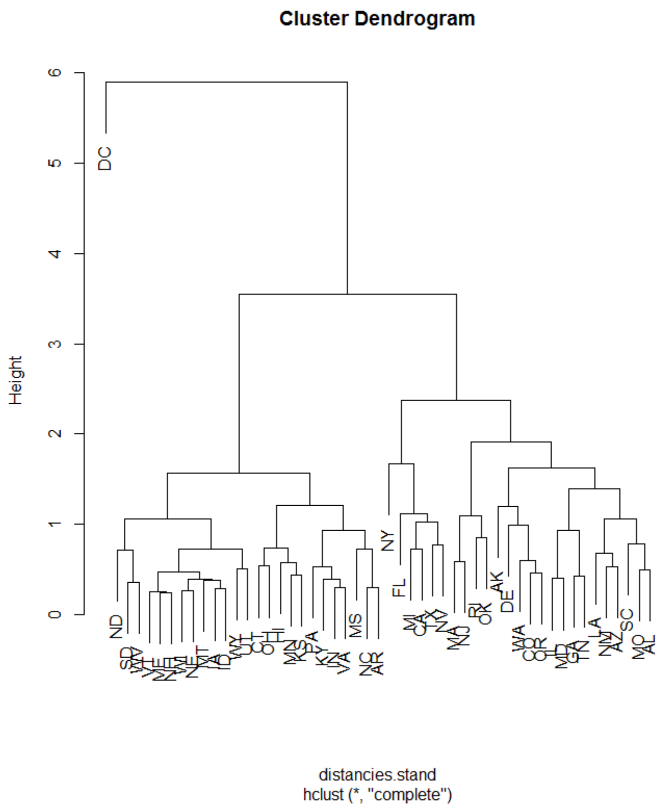
```
> cutree(cs,k=4)
ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD NE KS DE MD DC VA WV NC
 1  1  1  1  1  1  1  1  2  1  1  1  3  1  1  1  1  2  2  1  1  1  1  4  1
2  1  1  1  4  2  1  1  2  1  3  3  4  3  1  3  4  3  4  3  3  4  4  3
AK HI
 3  3
```

## 11.5 Qüestió 4

**Estandaritza, sense centrar, les variables amb el clàssic `scale`. (cal escriure `center=FALSE`). Centrar et pot dificular formar els clusters (ja que redueix les distàncies).**

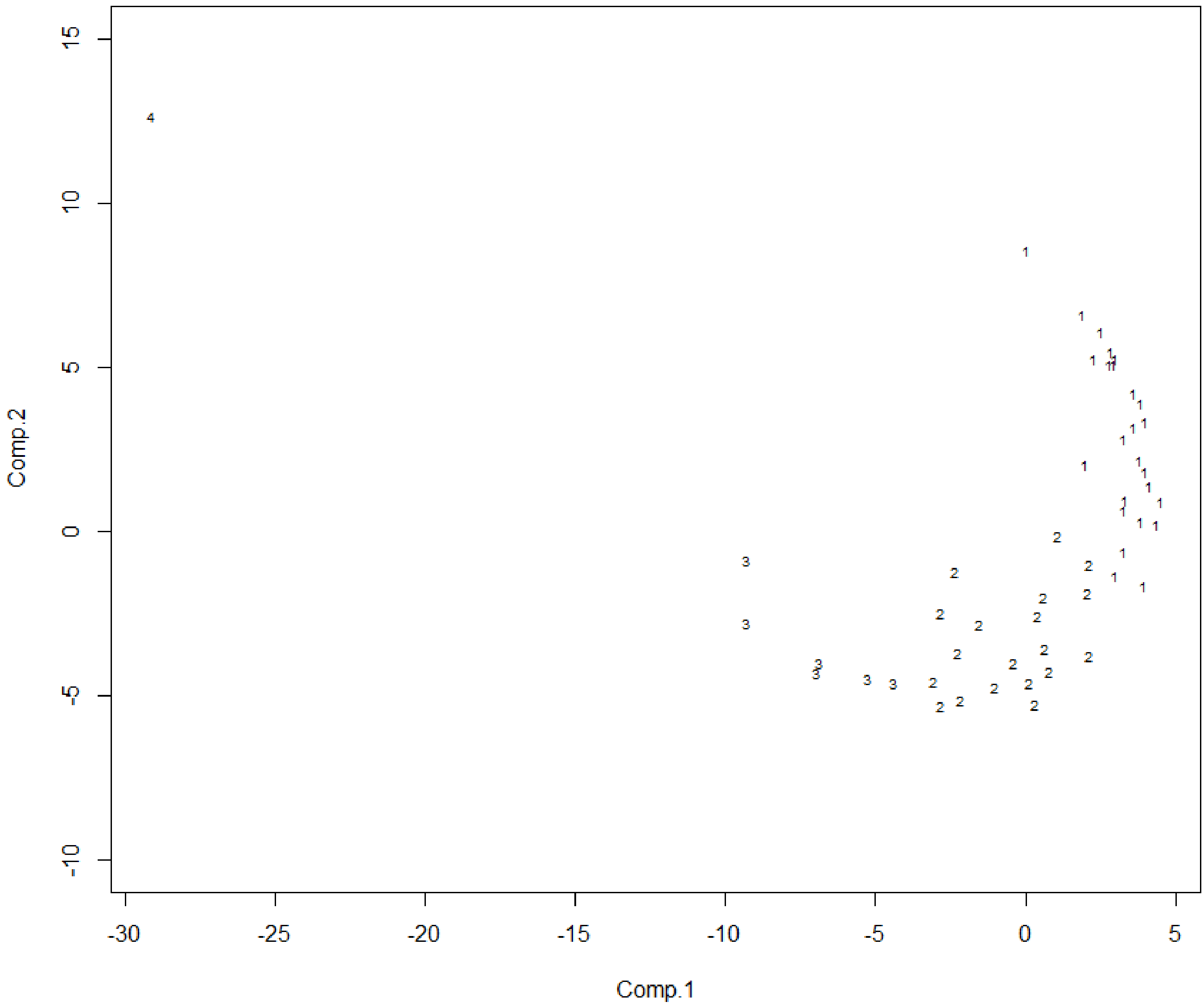
**Fes el dendograma per a aqeustes variables escalades.**

```
#estandaritza sense centrar
crime.stand=scale(crime,center=FALSE)
distancies.stand=dist(crime.stand)
cs.stand=hclust(distancies.stand,method="complete"); plot(cs.stand)
```



**Calcula les ocmponents principals de les variables estandaritzades (per tant usant la matriu de correlacions) i dibuixa el plot de les dues primeres components prinicpals, indicant en cada punt el conglomerat en el què l'has classificat.**

```
crime.pc <- princomp(distancies.stand, cor = TRUE)
xlim <- range(crime.pc$scores[,1])
plot(crime.pc$scores[,1:2], type = "n", xlim = xlim, ylim = c(-10, 15))
lab <-cutree(cs.stand, k=4, h = 4)
text(crime.pc$scores[,1:2], labels = lab, cex = 0.6)
```



**Amb les dades estandaritzades anteriors, ara utilitzarem el mètode de les $K$-means. Si agafem $k = 5$ quins conglomerats faries?**

```
clusters<-kmeans(crime.stand, centers = 5, iter.max = 10, nstart = 1,
                    algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"))$c
clusters
```

```
> clusters
ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD
 5  5  5  2  2  3  1  2  3  3  3  1  1  5  5  5  3  5  5
NE KS DE MD DC VA WV NC SC GA FL KY TN AL MS AR LA OK TX
 5  3  3  1  4  3  5  3  3  1  1  3  1  3  3  3  1  2  1
MT ID WY CO NM AZ UT NV WA OR CA AK HI
 5  5  5  1  1  1  5  1  3  1  1  1  3
> clusters
ME NH VT MA RI CT NY NJ PA OH IN IL MI WI MN IA MO ND SD
 5  5  5  2  2  3  1  2  3  3  3  1  1  5  5  5  3  5  5
NE KS DE MD DC VA WV NC SC GA FL KY TN AL MS AR LA OK TX
 5  3  3  1  4  3  5  3  3  1  1  3  1  3  3  3  1  2  1
MT ID WY CO NM AZ UT NV WA OR CA AK HI
 5  5  5  1  1  1  5  1  3  1  1  1  3
```

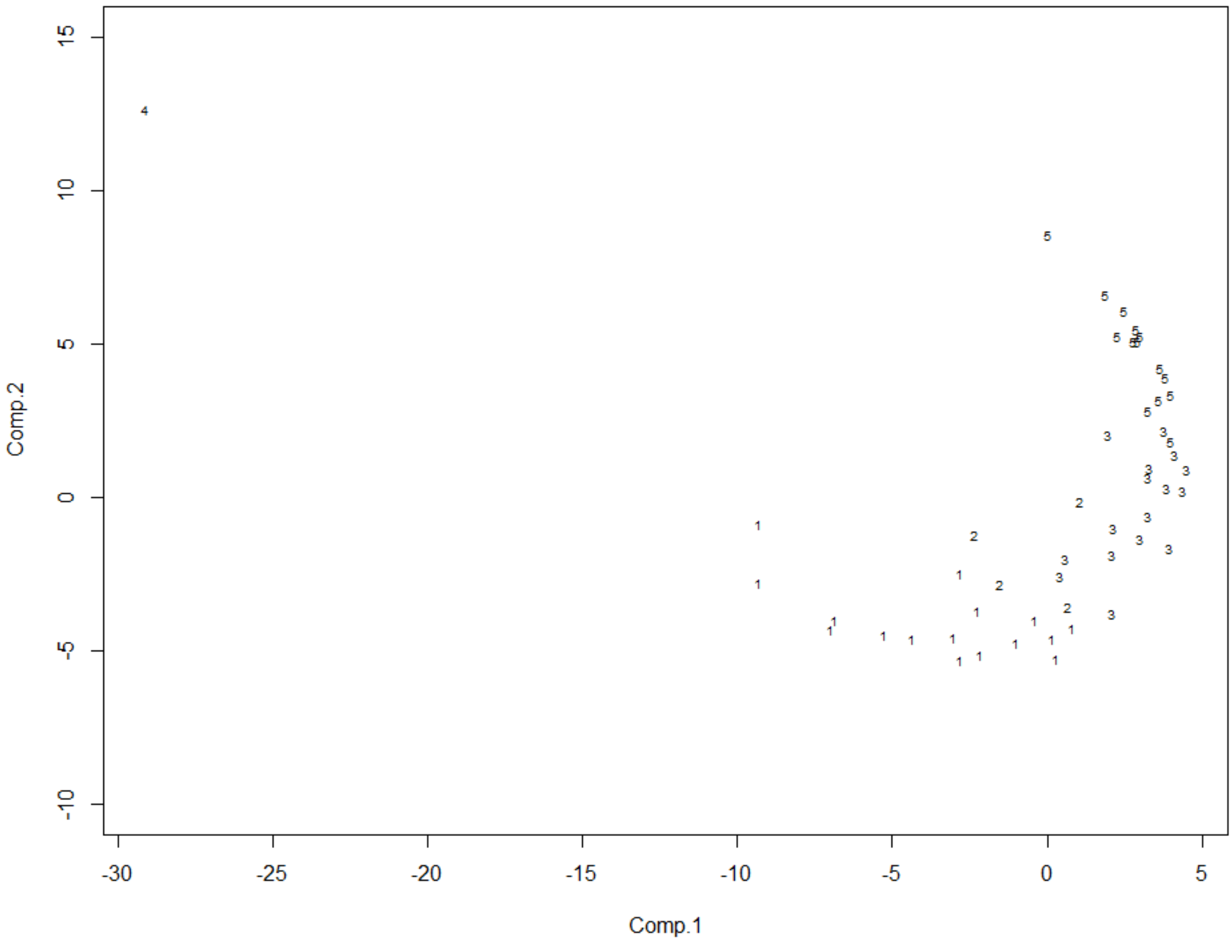
**Quins són els centroides dels 5 grups?**

```
 centroides<-kmeans(crime.stand, centers = 5, iter.max = 10, nstart = 1,
                      algorithm = c("Hartigan-Wong", "Lloyd", "Forgy", "MacQueen"))
centroides
```

```
 > centroides
     Murder      Rape   Robbery   Assault  Burglary     Theft   Vehicle
1 0.3438887 0.5328905 0.1814225 0.4225879 0.6135120 0.8598457 0.4096494
2 0.7582935 0.7890639 0.5480700 0.7365039 0.8234645 0.8123887 0.6923602
3 1.0987387 1.3015925 0.8333171 1.1521745 1.2852744 1.1687619 1.0288135
4 1.0224984 1.1145372 1.5730155 1.3522479 1.0915724 1.0180937 1.5482776
5 3.5366750 1.3971711 3.6270755 2.0723868 1.3395381 1.3465925 2.1366888
```

**Dibuixa el gràfic de les dues primeres components principals, acompanyant cada punt de l'etiqueta del cluster on l'has classificat.**

```
 xlim <- range(crime.pc$scores[,1])
plot(crime.pc$scores[,1:2], type = "n", xlim = xlim , ylim = c(-10, 15))
lab <-clusters
text(crime.pc$scores[,1:2], labels = lab, cex = 0.6)
```



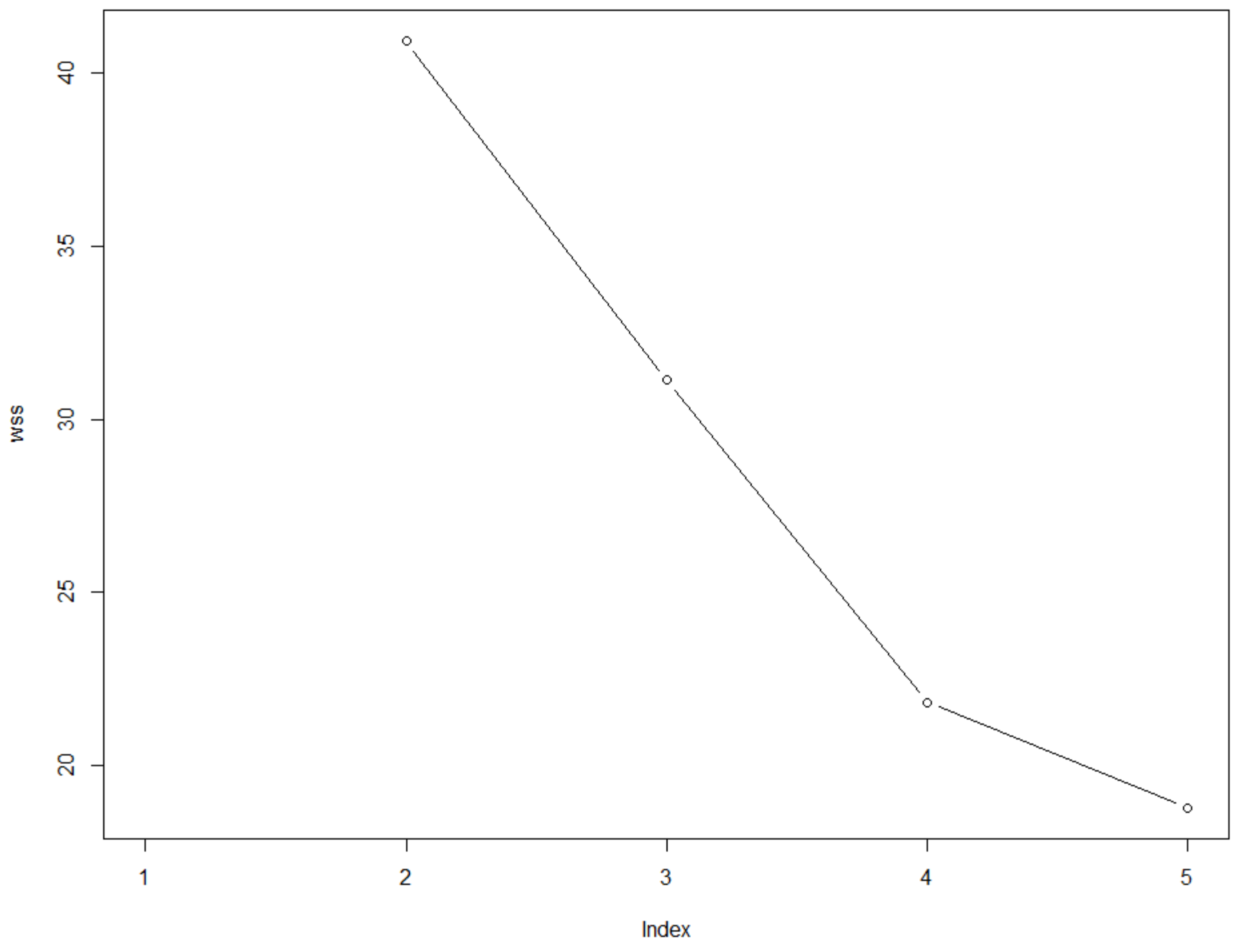

**Troba la variabilitat `whithin` dels conglomerats que has format. (és una sortida de la comanda kmeans)**

```
within<-kmeans(crime.stand, centers = 5, iter.max = 10, nstart = 1, algorithm =
within
```

```
> within
[1] 0.000000 2.139227 5.113720 5.277730 6.232814
```

**Fes un gràfic de les variabilitats `within` si fem entre 2 i 5 conglomerats amb el mètode k-means.**

```
n <- nrow(crime.stand)
wss <- rep(0, 5)
wss[1] <- (n - 1) * sum(sapply(crime.stand, var))
for (i in 2:5)
   {
      wss[i] <- sum(kmeans(crime.stand, centers = i)$withinss)
}
plot(wss,type="b")
```



**Crea una nova variable categòrica que m'indiqui si l'estat és segur (S), normal (N) o perillós (P), en funció de si la ratio d'assasinats és més baixa que 7, està entre 7 i 12 o és superior a 12.**

```
crime=as.data.frame(crime)
Segur<-subset(crime, Murder<7)
Normal<-subset(crime, Murder>=7 & Murder<=12)
Perillos<-subset(crime, Murder>12)
# Juntem totes les categories en un sol data frame
crime<-rbind(Segur,Normal,Perillos)
crime$Perillossitat<-c(rep("S", dim(Segur)[1]), rep("N", dim(Normal)[1]),
                        rep("P", dim(Perillos)[1]))
 crime$Perillossitat<-as.factor(crime$Perillossitat)
 crime<-crime[order(rownames(crime)),]
 crime<-as.data.frame(crime)
crime
```

```
 #Podem veure la taula de contingencia de perillositat usant:
xtabs(~Perillossitat,data=crime)
```

```
 Perillossitat
 N  P  S
20  4 27
```

# 12 Pràctica 12

## 12.1 Introducció

Usem les dades `pottery` de la biblioteca `HSAUR2`:

```
library(HSAUR2)
pottery
```

## 12.2 Qüestió 1

**Escriu explícitament les funcions discriminants en aquest cas. Per què surten 4 funcions discriminants?** Primer fem l'anàlisi lineal discriminant

```
# Primer realitzem l'analisi discriminant lineal:
library(MASS)
pottery.lda=lda(kiln ~ Al2O3 + Fe2O3 + MgO + CaO
                                          + Na2O + K2O + TiO2 + MnO + BaO ,
               data=pottery,
               na.action="na.omit", CV=FALSE)
   #equivalentment podem usar
   #pottery.lda=lda(kiln~.,data=pottery)
pottery.lda

pottery.lda$scaling[,]  #vectors propis matriu W-1B
#components dels vectors amb els que formarem les funcions discriminants
#es scalings (no loadings)

#els vaps de la matriu els obtinc:
pottery.lda$svd #indiquen la variabilitat explicada per cada funcio discriminant
```

Així doncs els loadings de les funcions descriminants són:

```
> round(pottery.lda$scaling[,],2)  #estem arrodonint a 2 xifres
        LD1    LD2    LD3    LD4
Al2O3 -0.66  -0.20   0.08  -0.26
Fe2O3  1.11  -1.51   0.25  -0.03
MgO    0.19   0.24   1.51  -0.21
CaO   -1.16  -3.52  -0.84   0.66
Na2O  -3.65  -3.01   2.49   0.06
K2O    2.35   2.07  -3.29  -1.46
TiO2  -5.97   2.56   1.94  -2.78
MnO   22.87  13.57   5.05  15.06
BaO   38.43  77.19  80.12 -54.40
```

Per tant, la **primera** funció discriminant és

$$z_1 = -0.66 \times Al2O3 + 1.11 \times Fe2O3 + 0.19 \times MgO - 1.16 \times CaO - 3.65 \times Na2O$$
$$+2.35 \times K2O - 5.97 \times TiO2 + 22.87 \times MnO + 38.43 \times BaO$$

La **segona** funció discriminant és

$$z_2 = -0.20 \times Al2O3 - 1.51 \times Fe2O3 + 0.24 \times MgO - 3.52 \times CaO - 3.01 \times Na2O + 2.07 \times K2O \\ +2.56 \times TiO2 + 13.57 \times MnO + 77.19 \times BaO$$

La **tercera** funció descriminant és

$$z_3 = 0.08Al2O3 + 0.25Fe2O3 + 1.51MgO - 0.84CaO + 2.49Na2O - 3.29K2O \\ +1.94TiO2 + 5.05MnO + 80.12BaO$$

I la **quarta** funció discriminant és

$$z_4 = -0.26Al2O3 - 0.03Fe2O3 - 0.21MgO + 0.66CaO + 0.06Na2O - 1.46K2O \\ -2.78TiO215.06MnO - 54.40BaO$$

Surten 4 funciosn descriminants, ja que el nombre de màxim defuncions discriminnants és el mínim entre $g-1$ i $p$, en el nostre gas $g=5$ ja que tenim 5 grups i $p=9$ ja que tenim 9 variables.
Recorda que rang$(W)=p$ i rang$(B)=g-1$.

**Quins són els valors propis de la matriu $W^{-1}B$. Amb quantes funcions discriminants et quedaries? Per què?** Els vaps els calculem així:

```
pottery.lda$svd
```

Em quedo amb 4 funcions discriminants, ja que tenim 4 valors propis, i el nombre de valors propis de la matriu $W^{-1}B$ correspón al nombre màxim de funcions discriminats.

**Quins són els vectors de mitjanes de les dades corresponent a la població 3 i a la 5?**

```
pottery.lda
pottery.lda$means[3,]
pottery.lda$means[5,]
```

```
> pottery.lda$means
    Al2O3    Fe2O3      MgO       CaO       Na2O      K2O       TiO2
MnO        BaO
1 16.91905 7.428571 1.842381 0.9390476 0.3457143 3.102857 0.9376190 0.07114286 0
2 12.55833 6.340000 4.931667 0.2008333 0.2550000 4.123333 0.7008333 0.12100000 0
3 11.70000 5.415000 3.855000 0.2950000 0.0500000 4.575000 0.5750000 0.09750000 0
4 18.18000 1.712000 0.674000 0.0260000 0.0540000 2.076000 1.0460000 0.00220000 0
5 17.32000 1.512000 0.606000 0.0520000 0.0480000 1.966000 0.9940000 0.00420000 0
> pottery.lda$means[3,]
  Al2O3   Fe2O3     MgO     CaO    Na2O     K2O    TiO2     MnO     BaO
11.7000  5.4150  3.8550  0.2950  0.0500  4.5750  0.5750  0.0975  0.0140
> pottery.lda$means[5,]
  Al2O3   Fe2O3     MgO     CaO    Na2O     K2O    TiO2     MnO     BaO
17.3200  1.5120  0.6060  0.0520  0.0480  1.9660  0.9940  0.0042  0.0156
```

**Troba les puntuacions discriminants de la gerra 3 i de la 45.**

```
pottery.lda.values=predict(pottery.lda,pottery[1:9])
#pottery.lda.values$x[,i] conte el valor de la i-essima funcio discriminant
pottery.lda.values$x[3,]
pottery.lda.values$x[45,]
```

que ens torna

```
> pottery.lda.values$x[3,]
       LD1          LD2          LD3          LD4
-1.1919304   -4.8577957   0.3428354   -0.1062783
> pottery.lda.values$x[45,]
        LD1           LD2           LD3           LD4
-11.65419926    3.86864925    1.08193448   -0.01493934
```

és a dir

**Calcula la mitjana de la primera puntuació discriminant de tots els individus de la base de dades i comprova que és 0.**

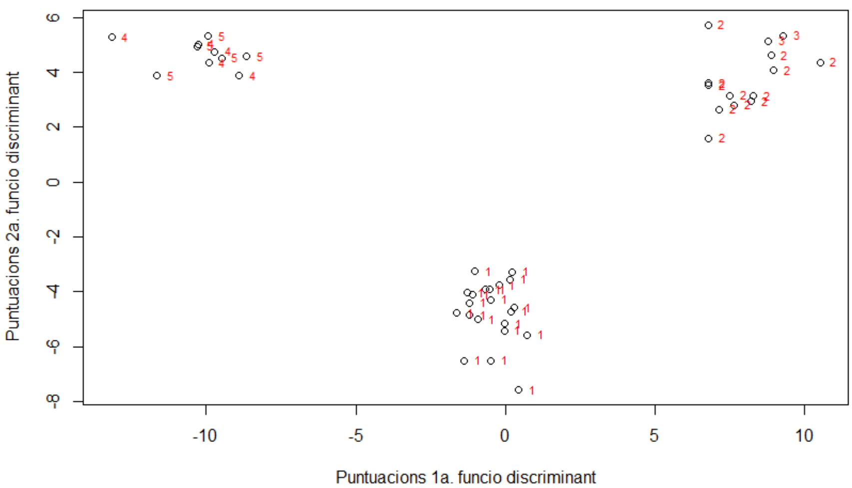
```
mean(pottery.lda.values$x[,1])
```

que ens torna un valor que podem considerar com a zero:

```
> mean(pottery.lda.values$x[,1])
[1] -7.459311e-16
```

**Fes el diagrama de punts de les dues primeres puntuacions discriminants, indicant cada punt de quin forn procedeix.**

```
plot(pottery.lda.values$x[,1],pottery.lda.values$x[,2],xlab = "Puntuacions 1a. f
    ylab = "Puntuacions 2a. funcio discriminant")
 text(pottery.lda.values$x[,1], pottery.lda.values$x[,2], pottery$kiln , cex=0.7,
```



**Una nova gerra trobada en unes excavacions té per valors de les 9 variables d'interès el següent vector**

$$(17.1, 8.1, 2.01, 0.70, 0.50, 3.12, 0.90, 0.087, 0.019)$$

**En quin forn la classificaries?**
R pot fer la classificació directa usant

```
predict(pottery.lda,
        newdata=data.frame(Al2O3=17.1, Fe2O3=8.1,  MgO=2.01,  CaO=0.7,
                 Na2O=0.5,K2O=3.12,TiO2=0.9   ,MnO=0.087   ,BaO=0.019))
```

que ens torna

```
    $class
[1] 1
Levels: 1 2 3 4 5

$posterior
  1            2              3             4             5
1 1 6.813554e-28 8.133097e-45 1.122862e-47 7.706815e-46

$x
         LD1        LD2     LD3          LD4
1 0.5786901 -5.057714 1.0474 -0.03539159
```

Per tant, classifiquem aquest individu al grup 1.

També podem fer els càlculs a mà, usant el criteri de màxima versemblança, recorda que

$$q_j = -\frac{1}{2}\left[\ln|\Sigma_j| + (x-\mu_j)^t\Sigma_j^{-1}(x-\mu_j)\right]$$

i classifiquem el nou individu al $q_j$ més gran.
Calculem els $q_j$ a mà usant R:

```
 x=c(17.1, 8.1, 2.01, 0.70, 0.50, 3.12, 0.90, 0.087, 0.019)

m1=pottery.lda$means[1,]
m2=pottery.lda$means[2,]
m3=pottery.lda$means[3,]
m4=pottery.lda$means[4,]
m5=pottery.lda$means[5,]

pottery1=pottery[which(pottery$kiln=='1'),]
pottery2=pottery[which(pottery$kiln=='2'),]
pottery3=pottery[which(pottery$kiln=='3'),]
pottery4=pottery[which(pottery$kiln=='4'),]
pottery5=pottery[which(pottery$kiln=='5'),]


CV1=cov(pottery1[1:9])
```

```
CV2=cov(pottery2[1:9])
CV3=cov(pottery3[1:9])
CV4=cov(pottery4[1:9])
CV5=cov(pottery5[1:9])
(q1=-(1/2)*(log(det(CV1))+t(x-m1)%*%solve(CV1)%*%(x-m1)))
(q2=-(1/2)*(log(det(CV2))+t(x-m2)%*%solve(CV1)%*%(x-m2)))
(q3=-(1/2)*(log(det(CV3))+t(x-m3)%*%solve(CV1)%*%(x-m3)))
(q4=-(1/2)*(log(det(CV4))+t(x-m4)%*%solve(CV1)%*%(x-m4)))
(q5=-(1/2)*(log(det(CV5))+t(x-m5)%*%solve(CV1)%*%(x-m5)))
```

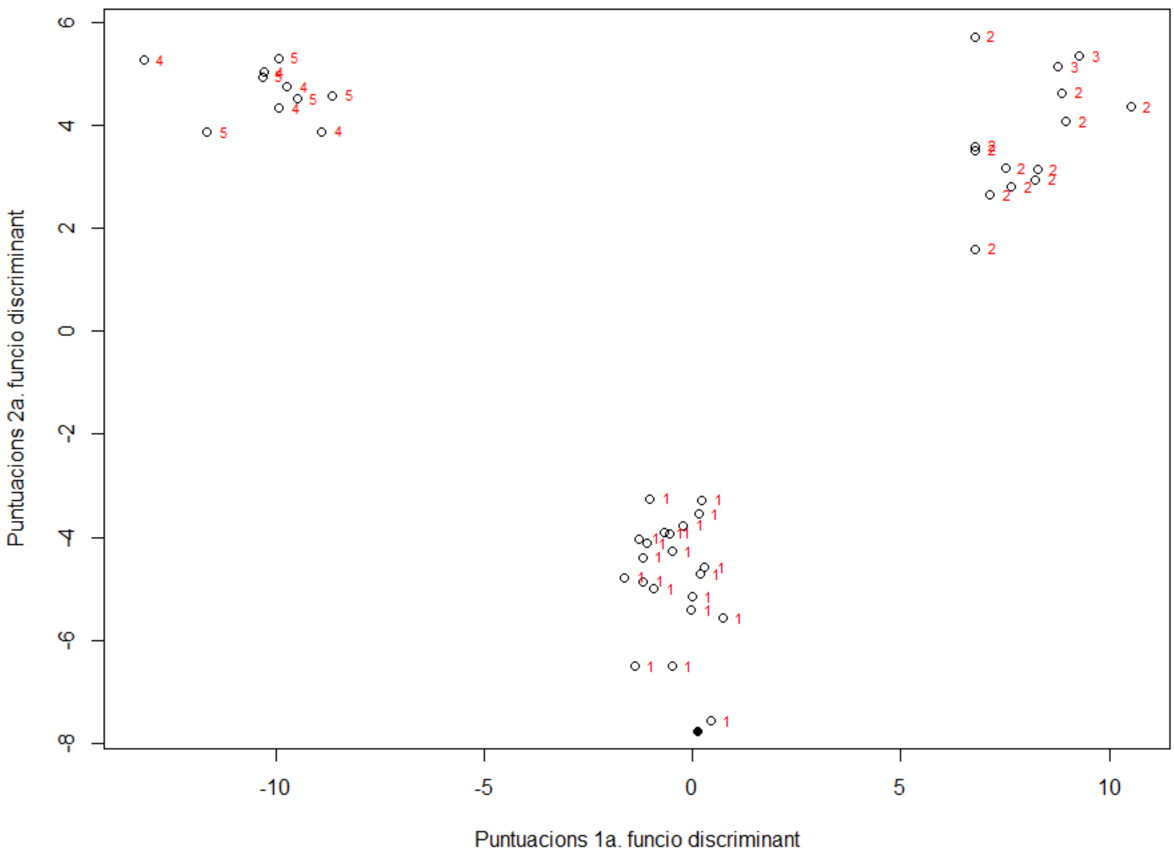
que ens torna

```
> (q1=-(1/2)*(log(det(CV1))+t(x-m1)%*%solve(CV1)%*%(x-m1)))
         [,1]
[1,] 17.61745
> (q2=-(1/2)*(log(det(CV2))+t(x-m2)%*%solve(CV1)%*%(x-m2)))
          [,1]
[1,] -449.0331
> (q3=-(1/2)*(log(det(CV3))+t(x-m3)%*%solve(CV1)%*%(x-m3)))
          [,1]
[1,] -111.7203
> (q4=-(1/2)*(log(det(CV4))+t(x-m4)%*%solve(CV1)%*%(x-m4)))
          [,1]
[1,] -91.39545
> (q5=-(1/2)*(log(det(CV5))+t(x-m5)%*%solve(CV1)%*%(x-m5)))
          [,1]
[1,] -82.39823
```

per tant classifiquem l'individu al gurp 1.

Pot ser interessant fer el scatter plot de les dues primeres puntuacions discriminants d'aquest punt, i ajuntar-lo al scater plot anterior, per comprovar gràficament que té sentit classificar-lo al grup 1.

```
graphics.off()
plot(pottery.lda.values$x[,1],pottery.lda.values$x[,2],xlab = "Puntuacions 1a. f
    ylab = "Puntuacions 2a. funcio discriminant")
text(pottery.lda.values$x[,1], pottery.lda.values$x[,2], pottery$kiln , cex=0.7,

puntuacions.x=sapply(1:4,function(i){t(x)%*%pottery.lda$scaling[,i]})#puntuacion
points(puntuacions.x[1],puntuacions.x[2],pch=19)
```




La nova gerra està representada per el punt negre, i efectivament hem comprovat que té sentit gràficament classificar-lo al grup 1.

## 12.3 Qüestió 2

**Amb les dades `pottery`, Quines dimensions tindran les matrius `within` i `between`?**
Com que tenim $p = 9$ variables, seran matirus $9 \times 9$.

**Quins són els rangs de les dues matrius anteriors? i quin serà el rang de la matriu $W^{-1}B$?** El rang de la matriu $W$ és $p = 9$, el rang de $B$ és $g - 1 = 4$, on $g = 5$ (nombre de grups), i el rang de la matriu $W^{-1}B$ és el mínim entre aquests dos rangs, per tant 4.

**Crea una funció que donades unes dades em calculi les matrius $B$ i $W$ i el producte $W^{-1}B$.**

```
calcWithinGroupsVariance <- function(variable,groupvariable)
{
  # find out how many values the group variable can take
  groupvariable2 <- as.factor(groupvariable[[1]])
  levels <- levels(groupvariable2)
  numlevels <- length(levels)
  # get the mean and standard deviation for each group:
  numtotal <- 0
  denomtotal <- 0
  for (i in 1:numlevels)
  {
    leveli <- levels[i]
    levelidata <- variable[groupvariable==leveli,]
    levelilength <- length(levelidata)
    # get the standard deviation for group i:
    sdi <- sd(levelidata)
    numi <- (levelilength - 1)*(sdi * sdi)
    denomi <- levelilength
    numtotal <- numtotal + numi
    denomtotal <- denomtotal + denomi
  }
  # calculate the within-groups variance
  Vw <- numtotal / (denomtotal - numlevels)
  return(Vw)
}
```

```
calcBetweenGroupsVariance <- function(variable,groupvariable)
  {
    # find out how many values the group variable can take
      groupvariable2 <- as.factor(groupvariable[[1]])
      levels <- levels(groupvariable2)
      numlevels <- length(levels)
     # calculate the overall grand mean:
       grandmean <- colMeans(variable)
       # get the mean and standard deviation for each group:
         numtotal <- 0
         denomtotal <- 0
         for (i in 1:numlevels)
           {
             leveli <- levels[i]
             levelidata <- variable[groupvariable==leveli,]
             levelilength <- length(levelidata)
             # get the mean and standard deviation for group i:
               meani <- mean(levelidata)
               sdi <- sd(levelidata)
               numi <- levelilength * ((meani - grandmean)^2)
               denomi <- levelilength
               numtotal <- numtotal + numi
               denomtotal <- denomtotal + denomi
               }
         # calculate the between-groups variance
          Vb <- numtotal / (numlevels - 1)
          Vb <- Vb[[1]]
          return(Vb)
          }
```

```
calcWithinGroupsCovariance <- function(variable1,variable2,groupvariable)
{
  # find out how many values the group variable can take
  groupvariable2 <- as.factor(groupvariable[[1]])
  levels <- levels(groupvariable2)
  numlevels <- length(levels)
  # get the covariance of variable 1 and variable 2 for each group:
  Covw <- 0
  for (i in 1:numlevels)
  {
    leveli <- levels[i]
    levelidata1 <- variable1[groupvariable==leveli,]
    levelidata2 <- variable2[groupvariable==leveli,]
    mean1 <- mean(levelidata1)
    mean2 <- mean(levelidata2)
    levelilength <- length(levelidata1)
    # get the covariance for this group:
    term1 <- 0
    for (j in 1:levelilength)
    {
      term1 <- term1 + ((levelidata1[j] - mean1)*(levelidata2[j] - mean2))
    }
    Cov_groupi <- term1 # covariance for this group
    Covw <- Covw + Cov_groupi
  }
  totallength <- nrow(variable1)
  Covw <- Covw / (totallength - numlevels)
  return(Covw)
}
```

```
calcBetweenGroupsCovariance <- function(variable1,variable2,groupvariable)
  {
    # find out how many values the group variable can take
      groupvariable2 <- as.factor(groupvariable[[1]])
      levels <- levels(groupvariable2)
      numlevels <- length(levels)
     # calculate the grand means
       # Amb la funcio "mean" no funciona!!!
       #variable1mean <- mean(variable1)
       #variable2mean <- mean(variable2)
       variable1mean <- colMeans(variable1)
       variable2mean <- colMeans(variable2)

         # calculate the between-groups covariance
         Covb <- 0
         for (i in 1:numlevels)
           {
             leveli <- levels[i]
             levelidata1 <- variable1[groupvariable==leveli,]
             levelidata2 <- variable2[groupvariable==leveli,]
             mean1 <- mean(levelidata1)
             mean2 <- mean(levelidata2)
             levelilength <- length(levelidata1)
             term1 <- (mean1 - variable1mean)*
             (mean2 - variable2mean)*(levelilength)
             Covb <- Covb + term1
             }
         Covb <- Covb / (numlevels - 1)
         Covb <- Covb[[1]]
         return(Covb)
         }
```

Un cop definides aquestes funcions, en defnim unes que ens donguin les matrius directament usant-les:

```
MatrixW=function(mydataset){ #assumeix que la ultima variable es la de classe
  p=dim(mydataset)[2]-1
  W=matrix(rep(0,p^2),ncol=p)
  for(i in 1:p){
    for(j in 1:p){
      if(i==j){W[i,j]=calcWithinGroupsVariance(pottery[i],pottery[p+1])}
      else {W[i,j]=calcWithinGroupsCovariance(pottery[i],pottery[j],pottery[p+1]
  }}
 return(W)
}
MatrixB=function(mydataset){ #assumeix que la ultima variable es la de classe
  p=dim(mydataset)[2]-1
  B=matrix(rep(0,p^2),ncol=p)
  for(i in 1:p){
    for(j in 1:p){
      if(i==j){B[i,j]=calcBetweenGroupsVariance(pottery[i],pottery[p+1])}
    else {B[i,j]=calcBetweenGroupsCovariance(pottery[i],pottery[j],pottery[p+1])
  }}
  return(B)
}
####
B=MatrixB(pottery)
W=MatrixW(pottery)

qr(B)$rank
qr(W)$rank
qr(solve(W)%*%B)$rank
```

que ens torna

```
round(MatrixB(pottery),3)
          [,1]    [,2]    [,3]    [,4]   [,5]    [,6]   [,7]   [,8]   [,9]
 [1,]   56.380 -15.248 -39.332   3.662 -0.050 -18.057  3.413 -0.872  0.009
 [2,]  -15.248  58.725  17.533   8.139  2.946  12.758 -1.200  0.786  0.011
 [3,]  -39.332  17.533  29.578  -1.693  0.495  13.490 -2.362  0.697 -0.003
 [4,]    3.662   8.139  -1.693   1.831  0.467   0.163  0.163  0.031  0.003
 [5,]   -0.050   2.946   0.495   0.467  0.167   0.415 -0.013  0.032  0.001
 [6,]  -18.057  12.758  13.490   0.163  0.415   7.035 -1.144  0.356 -0.001
 [7,]    3.413  -1.200  -2.362   0.163 -0.013  -1.144  0.211 -0.055  0.001
 [8,]   -0.872   0.786   0.697   0.031  0.032   0.356 -0.055  0.019  0.000
 [9,]    0.009   0.011  -0.003   0.003  0.001  -0.001  0.001  0.000  0.000
> round(MatrixW(pottery),3)
         [,1]   [,2]  [,3]   [,4]   [,5]  [,6]   [,7]  [,8]  [,9]
 [1,]   2.399  0.526 0.139 -0.053  0.014 0.256  0.035 0.008 0.002
 [2,]   0.526  0.494 0.060 -0.017  0.023 0.114  0.051 0.004 0.001
 [3,]   0.139  0.060 0.381  0.008  0.002 0.079  0.000 0.001 0.000
 [4,]  -0.053 -0.017 0.008  0.044 -0.001 0.008 -0.002 0.000 0.000
 [5,]   0.014  0.023 0.002 -0.001  0.018 0.013  0.003 0.002 0.000
 [6,]   0.256  0.114 0.079  0.008  0.013 0.096  0.013 0.002 0.000
 [7,]   0.035  0.051 0.000 -0.002  0.003 0.013  0.014 0.000 0.000
 [8,]   0.008  0.004 0.001  0.000  0.002 0.002  0.000 0.000 0.000
 [9,]   0.002  0.001 0.000  0.000  0.000 0.000  0.000 0.000 0.000

 > qr(B)$rank
[1] 4
> qr(W)$rank
[1] 9
> qr(solve(W)%*%B)$rank
[1] 4
```

## 12.4 Exercici Final

*(pag 94-101 Everitt)* **Troba la combinació lineal de les 2 primeres variables de la variable pottery i la combinació lineal de les 2 següents que tenen correlació més gran**
La combinació lineal de les 2 primeres varibles de la variable pottery i al combinació lienal de les 2 següents que tenen correlació més gran és $u_1$ i $v_1$ (mira el càlcul de $u_1$ i $v_1$ a continuació).
És tan senzill com trobar $u_2$ i $v_2$, ja que $u_i$ són mutuament incorrelacionades i $v_i$ també.

Fem els càlculs de les $v_i$ i $u_i$ a R:

```
r11=as.matrix(pottery[1:2,1:2])
r22=as.matrix(pottery[3:4,3:4])
r12=as.matrix(pottery[1:2,3:4])
r21=as.matrix(pottery[3:4,1:2])

(E1=solve(r11) %*% r12 %*% solve(r22) %*%r21)
(E2 <- solve(r22) %*% r21 %*% solve(r11) %*%r12)

(e1 <- eigen(E1))
e1$values
round(e1$vectors,2)
(e2 <- eigen(E2))
e2$values
round(e2$vectors,2)

#D'aqui treiem les relacions:
#u1=0.9V1-0.43V2
#v1=-0.39V3--0.92V4

#u2=-0.4V1+0.92V2
#v2=-0.45V3+0.89V4
```

$$
\begin{aligned}
u_1 &= 0.9V1 - 0.43V2 \\
v_1 &= -0.39V3 - 0.92V4 \\
u_2 &= -0.4V1 + 0.92V2 \\
v_2 &= -0.45V3 + 0.89V4
\end{aligned}
$$