UNIVERSITAT DE BARCELONA

Universitat Oberta de Catalunya

Escalado Multidimensional (MDS)
Soluciones

Francesc Carmona y Josep Gregori*

29 de octubre de 2021

Ejercicio 1

La matriz adjunta indica las similitudes[1] *entre siete animales según un experto:*

```
> D <- matrix(c(0,7,5,9,5,7,9,
+               7,0,4,6,4,6,7,
+               5,4,0,3,4,5,6,
+               9,6,3,0,3,2,2,
+               5,4,4,3,0,5,4,
+               7,6,5,2,5,0,4,
+               9,7,6,2,4,4,0), ncol=7, byrow=T)
> colnames(D) <- rownames(D) <- LETTERS[1:7]
```

a) *Construir la matriz* $\mathbf{B} = -\frac{1}{2}\mathbf{H}\mathbf{D}^{(2)}\mathbf{H}$, *donde* $\mathbf{D}^{(2)}$ *es la matriz de distancias al cuadrado y* $\mathbf{H}$ *es la matriz de centrado, y calcular sus valores propios. Observar si la matriz de distancias es euclídea.*[2]

La matriz de centrado $\mathbf{H}$ es

```
> n <- nrow(D)
> unos <- rep(1,n)
> I <- diag(unos)
> H <- I - 1/n*(unos %*% t(unos))
```

y la matriz $\mathbf{B}$ es

```
> B <- -(1/2)*(H %*% D^2 %*% H)
> rownames(B) <- colnames(B) <- rownames(D)
> round(B,4)

         A       B       C        D       E       F        G
A  31.5714 -0.6429  6.0000 -20.8571  4.5714 -4.0000 -16.6429
B  -0.6429 16.1429  2.7857  -6.0714  1.3571 -5.2143  -8.3571
C   6.0000  2.7857  5.4286   2.0714 -4.0000 -5.0714  -7.2143
D -20.8571 -6.0714  2.0714   7.7143  0.6429  6.5714   9.9286
E   4.5714  1.3571 -4.0000   0.6429  2.5714 -6.5000   1.3571
F  -4.0000 -5.2143 -5.0714   6.5714 -6.5000  9.4286   4.7857
G -16.6429 -8.3571 -7.2143   9.9286  1.3571  4.7857  16.1429
```

*Alumno del curso 2009-10

[1] Aquí la palabra "similitudes" se utiliza coloquialmente y no es correcta según la definición estadística. Se ha utilizado porque estaba en el ejercicio original. En realidad es una matriz de distancias.

[2] El paquete `ade4` dispone de una función `is.euclid()`.

Veamos sus valores propios:

```
> eigen(B)$values

[1]  5.791850e+01  2.142208e+01  1.219826e+01  8.714724e+00  4.137911e-01
[6] -1.243450e-14 -1.166735e+01
```

La matriz de distancias **D** no es euclídea ya que la matriz **B** tiene valores propios negativos. Lo mismo se puede ver con la función `is.euclid()` del paquete `ade4`.

```
> library(ade4)
> is.euclid(as.dist(D))

[1] FALSE
```

b) *Obtener la representación con las dos primeras coordenadas principales e indicar el grado de bondad de esta representación.*

Se puede hacer a partir de la descomposición de la matriz **B** *o con la función*

`cmdscale(`*dist*`, eig=T)`.

Con la diagonalización de la matriz **B**, las dos primeras coordenadas principales son

```
> B.eigen <- eigen(B)
> # Coordenadas principales
> X <- B.eigen$vect[,1:2] %*% diag(sqrt(B.eigen$val[1:2]))
> rownames(X) <- rownames(B)
> colnames(X) <- c("PC1","PC2"); X

         PC1         PC2
A  5.3653128  2.10386005
B  1.3884389 -3.58622610
C  1.1638928 -0.81833285
D -3.2507539 -0.24452185
E  0.4520411 -0.02609855
F -1.5654748  1.50797637
G -3.5534569  1.06334294
```

cuya bondad de ajuste en dimensión 2 es

```
> sum(B.eigen$val[1:2])/sum(B.eigen$val[B.eigen$val>0])

[1] 0.7881461
```

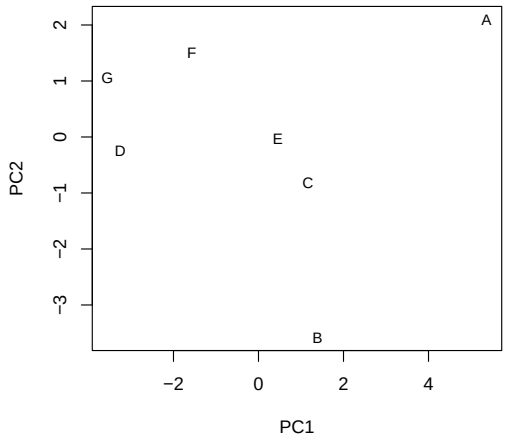
Repetimos ahora con la función `cmdscale()`

```
> mds <- cmdscale(as.dist(D),eig=T); mds

$points
         [,1]        [,2]
A  5.3653128  2.10386005
```

```
B  1.3884389 -3.58622610
C  1.1638928 -0.81833285
D -3.2507539 -0.24452185
E  0.4520411 -0.02609855
F -1.5654748  1.50797637
G -3.5534569  1.06334294

$eig
[1]  5.791850e+01  2.142208e+01  1.219826e+01  8.714724e+00  4.137911e-01
[6] -5.329071e-15 -1.166735e+01

$x
NULL

$ac
[1] 0

$GOF
[1] 0.7062874 0.7881461
```

La representación en coordenadas principales de los animales es

```
> plot(X,type="n")
> text(X[,1],X[,2],labels=rownames(D),cex=0.8)
```



Ejercicio 2 (∗)

Poner un ejemplo para comprobar que el escalado multidimensional clásico aplicado a las distancias euclídeas calculadas sobre una matriz de datos multivariantes $\mathbf{X}$ *es equivalente a la solución que se obtiene por el análisis de componentes principales de la matriz de covarianzas de* $\mathbf{X}$.

Simulamos una matriz de datos 20×5

```
> set.seed(123)
> X <- matrix(runif(100,0,10), ncol=5)
> X <- scale(X,scale=FALSE) # la centramos
```

y obtenemos las componentes principales $\mathbf{Y}$

```
> S <- cov(X)
> S.eigen <- eigen(S)
> Y <- X %*% S.eigen$vectors
> round(Y,4)

         [,1]    [,2]    [,3]    [,4]    [,5]
 [1,] -0.8794 -1.1888 -5.0618 -1.9851 -0.2231
 [2,] -2.4525  0.6449  3.8935 -1.7528 -1.1032
 [3,]  0.7306 -0.4688 -0.3203 -1.6444 -0.7127
 [4,] -5.3452 -0.2007  2.5303 -0.4962 -2.6332
 [5,] -2.5723  4.5190 -3.8677  1.5102  1.5748
 [6,]  0.8863 -3.7084 -2.7830 -3.3245  0.7931
 [7,] -1.9889 -4.0437  0.3165  3.8643  1.5294
 [8,] -2.8228 -0.5935  1.4180  5.3597 -0.4058
 [9,] -0.1369 -2.7669  0.9378  4.1986  2.9777
[10,]  6.3831  2.9080  1.1169  0.6076 -0.6943
[11,] -5.1090  3.8714 -4.6616  0.1154  0.3030
[12,] -1.6315 -2.3300 -1.2788  0.6935 -2.5187
[13,]  1.1560  2.1458 -1.0449  2.5262 -3.6384
[14,] -3.3238 -1.3108  2.5712 -4.3894  0.2700
[15,]  7.1029 -0.8629  0.1807 -0.3679  2.2206
[16,] -1.3342  4.5551  0.9982 -2.2706  2.0026
[17,] -1.6488 -4.7122 -0.0177 -2.0243  0.5640
[18,]  7.9835  0.1595 -2.2639 -0.2826 -0.4210
[19,]  5.3488 -0.0821  2.3683  0.2535 -2.2127
[20,] -0.3459  3.4651  4.9682 -0.5912  2.3279
```

Ahora calculamos las distancias euclídeas entre los puntos (filas)

```
> D <- as.matrix(dist(X))
```

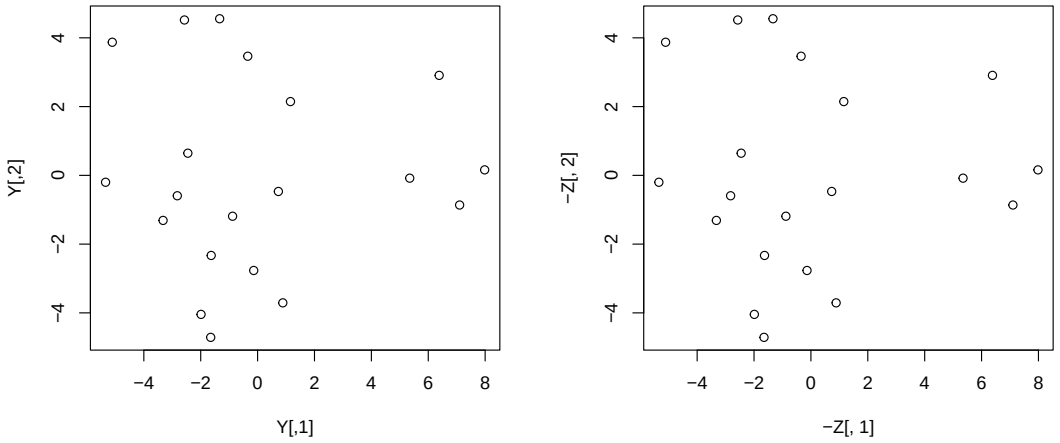
y obtenemos las coordenadas principales $\mathbf{Z}$

```
> n <- nrow(D)
> unos <- rep(1,n)
> H <- diag(unos) - 1/n*(unos %*% t(unos))
> B <- -(1/2)*(H %*% D^2 %*% H)
> B.eigen <- eigen(B)
> B.values <- ifelse(B.eigen$values>0,B.eigen$values,0)
> Z <- B.eigen$vectors %*% diag(sqrt(B.values))
> round(Z[,1:5],4)

         [,1]    [,2]    [,3]    [,4]    [,5]
 [1,]  0.8794  1.1888  5.0618 -1.9851  0.2231
 [2,]  2.4525 -0.6449 -3.8935 -1.7528  1.1032
 [3,] -0.7306  0.4688  0.3203 -1.6444  0.7127
 [4,]  5.3452  0.2007 -2.5303 -0.4962  2.6332
 [5,]  2.5723 -4.5190  3.8677  1.5102 -1.5748
 [6,] -0.8863  3.7084  2.7830 -3.3245 -0.7931
 [7,]  1.9889  4.0437 -0.3165  3.8643 -1.5294
 [8,]  2.8228  0.5935 -1.4180  5.3597  0.4058
 [9,]  0.1369  2.7669 -0.9378  4.1986 -2.9777
```

```
[10,] -6.3831 -2.9080 -1.1169  0.6076  0.6943
[11,]  5.1090 -3.8714  4.6616  0.1154 -0.3030
[12,]  1.6315  2.3300  1.2788  0.6935  2.5187
[13,] -1.1560 -2.1458  1.0449  2.5262  3.6384
[14,]  3.3238  1.3108 -2.5712 -4.3894 -0.2700
[15,] -7.1029  0.8629 -0.1807 -0.3679 -2.2206
[16,]  1.3342 -4.5551 -0.9982 -2.2706 -2.0026
[17,]  1.6488  4.7122  0.0177 -2.0243 -0.5640
[18,] -7.9835 -0.1595  2.2639 -0.2826  0.4210
[19,] -5.3488  0.0821 -2.3683  0.2535  2.2127
[20,]  0.3459 -3.4651 -4.9682 -0.5912 -2.3279
```

Así pues, dada una matriz de datos $\mathbf{X}$, se ve que las componentes principales $\mathbf{Y}$ son equivalentes a las coordenadas principales $\mathbf{Z}$, habida cuenta de la indeterminación de los signos (factor de ± 1 en cada vector).
La representación de las observaciones es la misma:

```
> plot(Y)             # componentes principales en dos dimensiones
> plot(-Z[,1],-Z[,2]) # coordenadas principales en dos dimensiones
```



Ejercicio 3 (∗)

Frecuentemente en las aplicaciones nos encontramos con una variable categórica nominal con k estados excluyentes medida sobre una muestra de $n = n_1 + \cdots + n_g$ individuos provenientes de g poblaciones. En estas condiciones, el vector de frecuencias de la i-ésima población $\mathbf{n}_i = (n_{i1}, \ldots, n_{ik})$ tiene una distribución conjunta multinomial con parámetros $(n_i, \mathbf{p}_i)$, donde $n_i = n_{i1} + \cdots + n_{ik}$ y $\mathbf{p}_i = (p_{i1}, \ldots, p_{ik})$ son las probabilidades de un individuo de la población i tales que $\sum_{j=1}^{k} p_{ij} = 1$.
En genética, una medida de disimilaridad, entre otras muchas en estas circunstancias, es la distancia de Bhattacharyya cuya expresión es

$$d_{ij}^2 = \arccos\left(\sum_{l=1}^{k} \sqrt{p_{il}p_{jl}}\right) \tag{1}$$

Si definimos el coeficiente de Bhattacharyya entre dos poblaciones como

$$\cos\varphi = \sum_{l=1}^{k} \sqrt{p_{il}p_{jl}}$$

esta distancia está relacionada en genética con las distancias de Cavalli-Sforza

$$d_{\text{C-Sf}} = \sqrt{2(1-\cos\varphi)}$$

y la distancia de Nei[3]

$$d_{\text{Nei}} = -\ln\cos\varphi$$

La siguiente tabla contiene las proporciones génicas observadas de los grupos sanguíneos correspondientes a 10 poblaciones distintas y excluyentes.

```
> pg <- matrix(c(0.21,0.06,0.06,0.67,
+                0.25,0.04,0.14,0.57,
+                0.22,0.06,0.08,0.64,
+                0.19,0.04,0.02,0.75,
+                0.18,0.00,0.15,0.67,
+                0.23,0.00,0.28,0.49,
+                0.30,0.00,0.06,0.64,
+                0.10,0.06,0.13,0.71,
+                0.27,0.04,0.06,0.63,
+                0.21,0.05,0.20,0.54), ncol=4, byrow=T)
> colnames(pg) <- c("gr.A","gr.B","gr.AB","gr.0")
> rownames(pg) <- c("francesa","checa","germánica","vasca","china",
+                   "ainu","esquimal","negra_USA","española","egipcia")
```

a) *Obtener las distancias de Bhattacharyya según la fórmula 1.*

```
> P <- pg
> Q <- sqrt(P)
> BC <- Q %*% t(Q)
> round(BC,3)

          francesa checa germánica vasca china  ainu esquimal negra_USA
francesa     1.000 0.988     0.999 0.992 0.959 0.922    0.966     0.983
checa        0.988 1.000     0.993 0.965 0.975 0.966    0.969     0.978
germánica    0.999 0.993     1.000 0.986 0.963 0.935    0.966     0.984
vasca        0.992 0.965     0.986 1.000 0.949 0.890    0.966     0.968
china        0.959 0.975     0.963 0.949 1.000 0.981    0.982     0.964
ainu         0.922 0.966     0.935 0.890 0.981 1.000    0.952     0.932
esquimal     0.966 0.969     0.966 0.966 0.982 0.952    1.000     0.936
negra_USA    0.983 0.978     0.984 0.968 0.964 0.932    0.936     1.000
española     0.997 0.991     0.997 0.989 0.965 0.934    0.980     0.970
egipcia      0.976 0.996     0.984 0.944 0.969 0.971    0.948     0.980
          española egipcia
francesa     0.997   0.976
checa        0.991   0.996
germánica    0.997   0.984
vasca        0.989   0.944
china        0.965   0.969
ainu         0.934   0.971
esquimal     0.980   0.948
negra_USA    0.970   0.980
española     1.000   0.976
egipcia      0.976   1.000
```

[3] Fuera de la genética, ésta es la que se conoce como distancia de Bhattacharyya.

```
> # Distancia de Bhattacharyya (al cuadrado)
> BC <- ifelse(BC>1., 1., BC)
> D2B <- acos(BC)
> round(D2B,3)

          francesa checa germánica vasca china  ainu esquimal negra_USA
francesa     0.000 0.157     0.044 0.125 0.286 0.397    0.262     0.185
checa        0.157 0.000     0.116 0.267 0.224 0.260    0.248     0.209
germánica    0.044 0.116     0.000 0.166 0.271 0.364    0.261     0.177
vasca        0.125 0.267     0.166 0.000 0.322 0.473    0.261     0.255
china        0.286 0.224     0.271 0.322 0.000 0.193    0.190     0.271
ainu         0.397 0.260     0.364 0.473 0.193 0.000    0.310     0.370
esquimal     0.262 0.248     0.261 0.261 0.190 0.310    0.000     0.361
negra_USA    0.185 0.209     0.177 0.255 0.271 0.370    0.361     0.000
española     0.080 0.136     0.078 0.152 0.265 0.364    0.202     0.244
egipcia      0.220 0.090     0.179 0.336 0.249 0.242    0.323     0.200
          española egipcia
francesa     0.080   0.220
checa        0.136   0.090
germánica    0.078   0.179
vasca        0.152   0.336
china        0.265   0.249
ainu         0.364   0.242
esquimal     0.202   0.323
negra_USA    0.244   0.200
española     0.000   0.221
egipcia      0.221   0.000
```

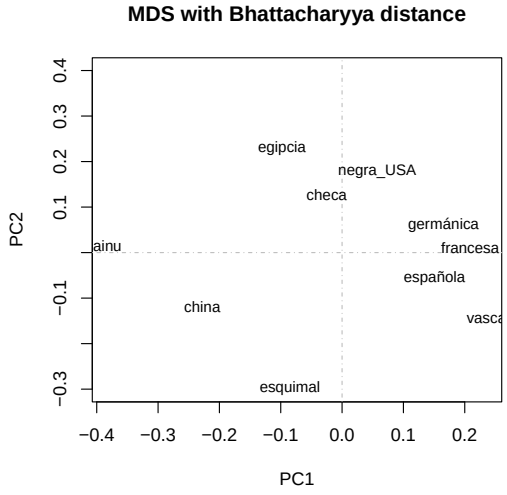
b) *Representar estas poblaciones con las dos primeras coordenadas principales. ¿Se observa algún tipo de agrupación?*

El escalado multidimensional es

```
> mds <- cmdscale(as.dist(sqrt(D2B)),eig=T)
```

y su representación en coordenadas principales es

```
> plot(mds$points,type="n",xlab="PC1",ylab="PC2",ylim=c(-0.3,0.4))
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,1], mds$points[,2],labels=rownames(D2B),cex=0.8)
> title("MDS with Bhattacharyya distance")
```



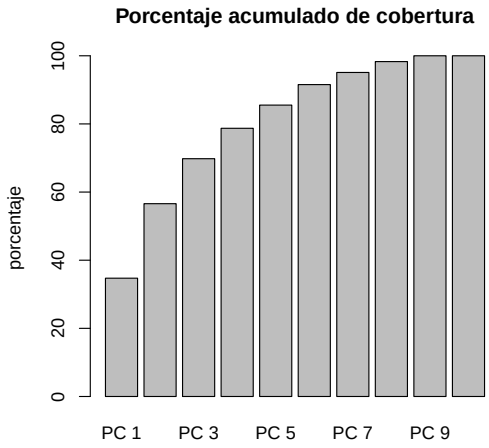
c) *¿Es ésta una distancia euclídea? ¿Cuál es la dimensión de la representación euclídea? Determinar el porcentaje de variabilidad explicado por las dos primeras coordenadas principales.*

Es una distancia euclídea porque no presenta valores propios negativos, el último debe considerarse 0.

```
> is.euclid(as.dist(sqrt(D2B)),print=T)

 [1]  3.682035e-01  2.318954e-01  1.400283e-01  9.465435e-02  7.216623e-02
 [6]  6.386087e-02  3.774957e-02  3.385219e-02  1.804391e-02 -5.436766e-17
[1] TRUE
```

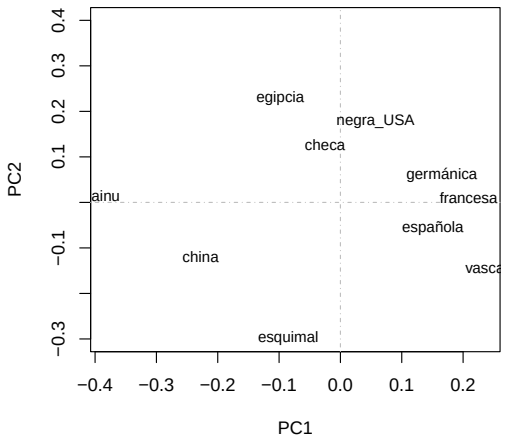
Porcentaje de variabilidad según dimensión.

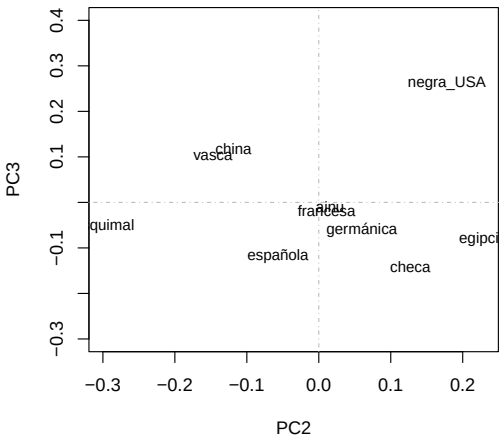
```
> n <- nrow(D2B)
> unos <- rep(1,n)
> I <- diag(unos)
> H <- I - 1/n*(unos %*% t(unos))
> B <- -1/2*(H %*% D2B %*% H)
> rownames(B) <- rownames(D2B)
> colnames(B) <- colnames(D2B)
> evv <- eigen(B)$values; evv

 [1]  3.682035e-01  2.318954e-01  1.400283e-01  9.465434e-02  7.216622e-02
 [6]  6.386086e-02  3.774957e-02  3.385219e-02  1.804391e-02 -8.944549e-18

> pb <- cumsum(evv)/sum(evv)*100; round(pb,2)

 [1]  34.72  56.59  69.79  78.72  85.52  91.55  95.11  98.30 100.00 100.00

> names(pb) <- paste("PC",1:10)
> barplot(pb,ylab="porcentaje")
> title(main="Porcentaje acumulado de cobertura",line=1.5)
```

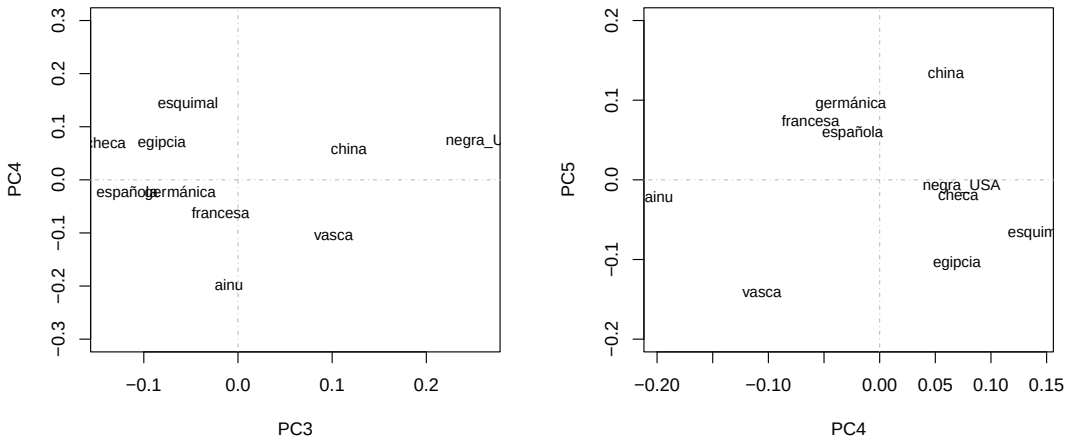


Representación dos a dos de algunas coordenadas principales:

```
> mds <- cmdscale(as.dist(sqrt(D2B)),k=5,eig=T)
> plot(mds$points,type="n",xlab="PC1",ylab="PC2",ylim=c(-0.3,0.4))
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,1], mds$points[,2],labels=rownames(D2B),cex=0.8)
> plot(mds$points[,2:3],type="n",xlab="PC2",ylab="PC3",ylim=c(-0.3,0.4))
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,2], mds$points[,3],labels=rownames(D2B),cex=0.8)
> plot(mds$points[,3:4],type="n",xlab="PC3",ylab="PC4",ylim=c(-0.3,0.3))
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,3], mds$points[,4],labels=rownames(D2B),cex=0.8)
> plot(mds$points[,4:5],type="n",xlab="PC4",ylab="PC5",ylim=c(-0.2,0.2))
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,4], mds$points[,5],labels=rownames(D2B),cex=0.8)
```







La distancia es euclídea puesto que la diagonalización de la matriz de distancias no presenta valores propios negativos. El último es prácticamente nulo.

No se llega a cubrir el 80 % de variabilidad con menos de 5 coordenadas principales (PC). Ver el gráfico de barras.

El grupo española/francesa/germánica se conserva bastante bien en la representación dos a dos de todas las dimensiones, lo que no es de extrañar dadas las similitudes en la distribución de grupos sanguíneos.

El resto no muestra similitudes que se conserven, pues en unas dimensiones aparecen opuestas y en otras pueden acercarse hasta casi superponerse. Los gráficos muestran el grado de conservación de características propias, entre poblaciones que han interactuado poco en su evolución.

Ejercicio 4

Si las variables observadas sobre un conjunto de individuos son del tipo binario, es necesario disponer de una matriz de distancias basadas en los coeficientes de similaridad que se obtienen de las tablas del tipo

	1	0
1	a	b
0	c	d

donde a es el número de variables con respuesta 1 en ambos individuos, d es el número de variables con respuesta 0 en ambos individuos, etc.
Entre los muchos coeficientes de similaridad destacan dos:

$$\text{Sokal y Michener: } s_{ij} = \frac{a+d}{p} \qquad \text{Jaccard: } s_{ij} = \frac{a}{p-d}$$

donde $p = a + b + c + d$ es el número de variables binarias observadas.
El coeficiente de Sokal y Michener[4] *se utiliza con variables binarias simétricas, es decir, aquellas en las que los valores 1 y 0 son intercambiables. Uno no es más importante que el otro, como el sexo. El coeficiente de Jaccard, en cambio, se utiliza con variables binarias asimétricas, es decir, aquellas en las que la presencia de la característica (el 1) es importante.*
Aplicando uno de estos coeficientes a un conjunto de individuos se obtiene una matriz de similaridades (s_{ij}) que se puede transformar en una distancia

$$d_{ij}^2 = s_{ii} + s_{jj} - 2s_{ij} = 2(1 - s_{ij})$$

[4]También llamado coeficiente de acoplamiento simple.

Dada la siguiente matriz de datos

```
> X <- matrix(c(1,1,0,0,1,1,
+               1,1,1,0,0,1,
+               1,0,0,1,0,1,
+               0,0,0,0,1,0), nrow=4, byrow=T)
> rownames(X) <- LETTERS[1:4]
> colnames(X) <- paste("X",1:6,sep="")
```

con las observaciones de 6 variables binarias sobre 4 individuos,

a) *Calcular los coeficientes de Sokal y Michener para cada par de individuos y obtener la matriz de distancias asociada.*

La similaridad de Sokal y Michener es

```
> SM.simil <- function(x,y){
+     if (length(x) != length(y)){
+            stop("los vectores han de tener la misma longitud")} else {
+     a <- sum(x==1 & y==1)
+     d <- sum(x==0 & y==0)
+     (a+d)/length(x)
+     }
+ }
```

Vamos a calcular la matriz de similaridades entre las filas de una matriz binaria $\mathbf{X}$ y, a partir de ella, la matriz de distancias (al cuadrado).

```
> bin.simil <- function(X,simil){
+   n <- nrow(X)
+   S <- matrix(0,nrow=n,ncol=n)
+   for(i in 1:(n-1)) for(j in (i+1):n) S[i,j] <- simil(X[i,],X[j,])
+   S <- S + diag(1,n) + t(S)
+   rownames(S) <- colnames(S) <- rownames(X)
+   as.dist(2*(1-S))
+ }
```

Ahora aplicamos estas funciones para calcular la matriz de distancias (al cuadrado) entre los 4 individuos:

```
> bin.simil(X,SM.simil)

          A         B         C
B 0.6666667
C 1.0000000 1.0000000
D 1.0000000 1.6666667 1.3333333
```

b) *Lo mismo que en el apartado anterior pero con el coeficiente de Jaccard.*

Similaridad de Jaccard

```
> J.simil <- function(x,y){
+     if (length(x) != length(y)) {
+           stop("los vectores han de tener la misma longitud")} else {
+     a <- sum(x==1 & y==1)
+     d <- sum(x==0 & y==0)
+     a /(length(x)-d)
+     }
+ }
> bin.simil(X,J.simil)

    A   B   C
B 0.8
C 1.2 1.2
D 1.5 2.0 2.0
```

Ejercicio 5

En muchos análisis multivariantes se dispone de un conjunto de variables mixto, es decir, unas variables son del tipo cuantitativo y otras cualitativo (nominales, ordinales o incluso binarias). En estos casos es necesario disponer de una distancia como la de Gower que tiene en cuenta el tipo de variable.
El cuadrado de la distancia de Gower entre dos filas de datos se define como $d_{ij}^2 = 1 - s_{ij}$, *donde* s_{ij} *es el coeficiente de similaridad*

$$s_{ij} = \frac{\sum_{h=1}^{p_1}(1 - |x_{ih} - x_{jh}|/G_h) + a + \alpha}{p_1 + (p_2 - d) + p_3}$$

p_1 *es el número de variables cuantitativas,* p_2 *es el número de variables binarias,* p_3 *el número de variables cualitativas (no binarias),* a *el número de coincidencias* $(1,1)$ *en las variables binarias,* d *el número de coincidencias* $(0,0)$ *en las variables binarias,* α *el número de coincidencias en las variables cualitativas (no binarias) y* G_h *es el rango (o recorrido) de la* h*-ésima variable cuantitativa.*

a) *Calcular la distancia de Gower entre los datos de 18 tipos de flores de la base de datos* ***flower*** *del paquete* ***cluster*** *de* ***R****.*

Probar la función `daisy()`[5] *del paquete* ***cluster****.*

Los datos.

```
> require(cluster)
> data(flower)
> str(flower)

'data.frame': 18 obs. of  8 variables:
 $ V1: Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 2 2 ...
 $ V2: Factor w/ 2 levels "0","1": 2 1 2 1 2 2 1 1 2 2 ...
 $ V3: Factor w/ 2 levels "0","1": 2 1 1 2 1 1 1 2 1 1 ...
 $ V4: Factor w/ 5 levels "1","2","3","4",..: 4 2 3 4 5 4 4 2 3 5 ...
 $ V5: Ord.factor w/ 3 levels "1"<"2"<"3": 3 1 3 2 2 3 3 2 1 2 ...
 $ V6: Ord.factor w/ 18 levels "1"<"2"<"3"<"4"<..: 15 3 1 16 2 12 13 7 4 14 ...
 $ V7: num  25 150 150 125 20 50 40 100 25 100 ...
 $ V8: num  15 50 50 50 15 40 20 15 15 60 ...

> colnames(flower) <- c("winters","shadow","tubers","color","soil","pref",
+                       "height","distance")
```

[5] Considerar atentamente si las variables binarias son simétricas o asimétricas.

La distancia de Gower:

```
> G.dist <- daisy(flower,type=list(symm=c(1,2),asymm=3))
> attr(G.dist,"Types")

[1] "S" "S" "A" "N" "O" "O" "I" "I"
```

En el cálculo de la distancia se han considerado los siguientos tipos de variables para las respectivas columnas:

simétrica, simétrica, asimétrica, nominal, ordinal, ordinal, intervalo, intervalo

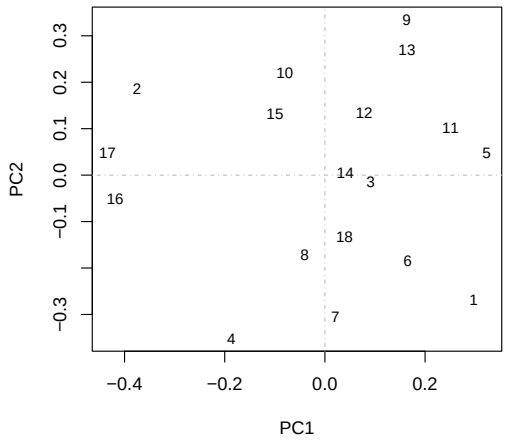
b) *Realizar un escalado multidimensional con esta distancia y representar las flores en un gráfico de dispersión de dos dimensiones.*

El escalado multidimensional con esta distancia es

```
> mds <- cmdscale(G.dist,eig=T); mds

$points
             [,1]         [,2]
 [1,]  0.29679786 -0.267818251
 [2,] -0.37498350  0.185603174
 [3,]  0.09175369 -0.014312588
 [4,] -0.18695632 -0.351656935
 [5,]  0.32275116  0.046753019
 [6,]  0.16540081 -0.185220673
 [7,]  0.02095674 -0.305706225
 [8,] -0.04058749 -0.172534548
 [9,]  0.16345140  0.334351964
[10,] -0.08006258  0.220997949
[11,]  0.25101547  0.102385809
[12,]  0.07838220  0.134277648
[13,]  0.16398347  0.269159304
[14,]  0.04093219  0.006265813
[15,] -0.09951482  0.132503360
[16,] -0.41848238 -0.052096511
[17,] -0.43414959  0.049266822
[18,]  0.03931167 -0.132219131

$eig
 [1]  9.115922e-01  6.915706e-01  5.317533e-01  3.574191e-01  2.190966e-01
 [6]  1.089910e-01  5.540941e-02  4.734512e-02  2.111761e-02 -5.551115e-17
[11] -2.387905e-03 -1.890756e-02 -2.356828e-02 -4.479860e-02 -6.770476e-02
[16] -1.065829e-01 -1.313834e-01 -1.566752e-01

$x
NULL

$ac
[1] 0

$GOF
[1] 0.4585308 0.5444980
```

Ahora podemos hacer la representación en coordenadas principales.

```
> require(MASS)
> plot(mds$points,type="n",xlab="PC1",ylab="PC2")
> abline(h=0,v=0,lty=4,col="gray")
> text(mds$points[,1],mds$points[,2],labels=1:18,cex=0.8)
```



con una bondad de ajuste

```
> evv <- cmdscale(G.dist,k=17,eig=T)$eig
> sum(evv[1:2])/sum(evv[evv>0])

[1] 0.544498
```

Ejercicio 6

Consideremos los datos sobre cráneos de varones egipcios de cinco épocas históricas que se pueden obtener en el siguiente enlace:

`http://lib.stat.cmu.edu/DASL/Datafiles/EgyptianSkulls.html`

También podemos bajarlos desde la página del libro de Everitt (2005) y así los podremos cargar directamente en R con la siguientes instrucciones:

```
> skulls <- source("chap5skulls.dat")$value
> str(skulls)

'data.frame': 150 obs. of  5 variables:
 $ EPOCH: Factor w/ 5 levels "c4000BC","c3300BC",..: 1 1 1 1 1 1 1 1 1 1 ...
 $ MB   : num  131 125 131 119 136 138 139 125 131 134 ...
 $ BH   : num  138 131 132 132 143 137 130 136 134 134 ...
 $ BL   : num  89 92 99 96 100 89 108 93 102 99 ...
 $ NH   : num  49 48 50 44 54 56 48 48 51 51 ...
```

Donde el path debe ser la dirección a la carpeta donde hemos dejado el archivo una vez descomprimido. Así tendremos la base de datos ***skulls*** *con cinco variables. La primera variable es el factor EPOCH y las otras cuatro son las medidas biométricas estudiadas del cráneo.*

a) *En primer lugar podemos realizar un MANOVA para contrastar la diferencia de medias entre los niveles del factor o poblaciones. No entraremos aquí en la comprobación de las hipótesis de normalidad y de igualdad de las matrices de covarianzas.*

```
> skulls.manova <- manova(cbind(MB,BH,BL,NH) ~ EPOCH, data=skulls)
> summary(skulls.manova, test="Wilks") # test="Pillai" or "Hotelling" or "Roy"

           Df   Wilks approx F num Df den Df   Pr(>F)
EPOCH       4 0.66359   3.9009     16 434.45 7.01e-07 ***
Residuals 145
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El p-valor obtenido muestra que hay diferencias entre las medias de las poblaciones.

b) (**) *Comprobaremos que el test anterior rechaza la igualdad de medias y, por lo tanto, justifica un* ***análisis canónico de poblaciones****.*

Utilizar la función `candisc()` *del paquete* `candisc` *o consultar*

`http://erre-que-erre-paco.blogspot.com/2010/03/analisis-canonico-de-poblaciones.html`

Habiéndose comprobado que las poblaciones son distintas puede realizarse un análisis canónico de poblaciones para representar en un espacio reducido los puntos de las medias poblacionales, y así visualizar los resultados del test anterior.

Calculamos en primer lugar la matriz centrada de medias.

```
> attach(skulls)
> nT <- nrow(skulls)
> p <- ncol(skulls)-1
> g <- nlevels(EPOCH)
> n.i <- tapply(EPOCH,EPOCH,length); n.i

c4000BC c3300BC c1850BC  c200BC  cAD150
     30      30      30      30      30

> # Vector de medias global
> gmean <- apply(skulls[,2:5],2,mean); gmean

       MB        BH        BL        NH
133.97333 132.54667  96.46000  50.93333

> # Medias de grupo
> means <- aggregate(skulls[,2:5],by=list(EPOCH),mean)
> means <- as.matrix(means[,2:5])
> rownames(means) <- levels(EPOCH); round(means,2)

            MB     BH    BL    NH
c4000BC 131.37 133.60 99.17 50.53
c3300BC 132.37 132.70 99.07 50.23
c1850BC 134.47 133.80 96.03 50.57
c200BC  135.50 132.30 94.53 51.97
cAD150  136.17 130.33 93.50 51.37
```

```
> # Matriz centrada de medias
> X <- means - matrix(gmean,nrow=g,ncol=p,byrow=TRUE); X

                MB         BH         BL         NH
c4000BC -2.6066667  1.0533333  2.7066667 -0.4000000
c3300BC -1.6066667  0.1533333  2.6066667 -0.7000000
c1850BC  0.4933333  1.2533333 -0.4266667 -0.3666667
c200BC   1.5266667 -0.2466667 -1.9266667  1.0333333
cAD150   2.1933333 -2.2133333 -2.9600000  0.4333333
```

y la matriz de covarianzas común

```
> # Matrices de covarianzas de grupo (denominador n-1)
> S.list <- by(skulls[,2:5],skulls$EPOCH,FUN=cov); S.list

skulls$EPOCH: c4000BC
          MB         BH         BL         NH
MB 26.309195  4.1517241  0.4540230  7.2459770
BH  4.151724 19.9724138 -0.7931034  0.3931034
BL  0.454023 -0.7931034 34.6264368 -1.9195402
NH  7.245977  0.3931034 -1.9195402  7.6367816
------------------------------------------------------------
skulls$EPOCH: c3300BC
          MB        BH         BL        NH
MB 23.136782  1.010345  4.7678161 1.8425287
BH  1.010345 21.596552  3.3655172 5.6241379
BL  4.767816  3.365517 18.8919540 0.1908046
NH  1.842529  5.624138  0.1908046 8.7367816
------------------------------------------------------------
skulls$EPOCH: c1850BC
           MB          BH         BL          NH
MB 12.1195402  0.78620690 -0.7747126  0.89885057
BH  0.7862069 24.78620690  3.5931034 -0.08965517
BL -0.7747126  3.59310345 20.7229885  1.67011494
NH  0.8988506 -0.08965517  1.6701149 12.59885057
------------------------------------------------------------
skulls$EPOCH: c200BC
          MB        BH        BL       NH
MB 15.362069 -5.534483 -2.172414 2.051724
BH -5.534483 26.355172  8.110345 6.148276
BL -2.172414  8.110345 21.085057 5.328736
NH  2.051724  6.148276  5.328736 7.964368
------------------------------------------------------------
skulls$EPOCH: cAD150
           MB         BH         BL         NH
MB 28.6264368 -0.2298851 -1.8793103 -1.9942529
BH -0.2298851 24.7126437 11.7241379  2.1494253
BL -1.8793103 11.7241379 25.5689655  0.3965517
NH -1.9942529  2.1494253  0.3965517 13.8264368

> # Matriz de covarianzas común (W)
> S <- matrix(0,nrow=4,ncol=4)
> for(i in 1:g) S <- S + n.i[i]*S.list[[i]]
> S <- S/nT; round(S,3)
```

```
       MB     BH     BL     NH
MB 21.111  0.037  0.079  2.009
BH  0.037 23.485  5.200  2.845
BL  0.079  5.200 24.179  1.133
NH  2.009  2.845  1.133 10.153
```

Ahora hacemos la diagonalización canónica por Cholesky.

```
> A <- crossprod(X)
> T <- chol(S)
> T.inv <- solve(T)
> evr <- eigen( t(T.inv) %*% A %*% T.inv ); evr

eigen() decomposition
$values
[1] 2.054627647 0.188494673 0.075904888 0.009765007

$vectors
           [,1]       [,2]       [,3]      [,4]
[1,]  0.6154648  0.1447008 -0.2974889 0.7153776
[2,] -0.2865995  0.8997428  0.2759714 0.1793414
[3,] -0.6878362 -0.3348596 -0.0402855 0.6427500
[4,]  0.2567980 -0.2395695  0.9130812 0.2072295
```

Valores propios, porcentajes de cobertura, y porcentajes acumulados.

```
> l <- evr$values
> pct <- round(l/sum(l)*100,2); pct

[1] 88.23  8.09  3.26  0.42

> pct.ac <- cumsum(pct); pct.ac

[1]  88.23  96.32  99.58 100.00
```

Vectores propios de $\mathbf{T'}^{-1}\mathbf{A}\mathbf{T}^{-1}$: $\mathbf{TV}$

```
> TV <- evr$vectors
```

Vectores propios $\mathbf{V}$: $\mathbf{T}^{-1}(\mathbf{TV})$

```
> V2 <- T.inv %*% TV; V2

          [,1]        [,2]        [,3]          [,4]
MB  0.12667629  0.03873784 -0.09276835  0.1488398644
BH -0.03703209  0.21009773  0.02456846 -0.0004200843
BL -0.14512512 -0.06811443 -0.01474860  0.1325007670
NH  0.08285128 -0.07729281  0.29458931  0.0668588797
```

Ortogonales (incorrelacionados) respecto a la métrica de $\mathbf{A}$

```
> t(V2) %*% A %*% V2

              [,1]          [,2]          [,3]         [,4]
[1,]  2.054628e+00 -2.220446e-16 -1.665335e-16 2.359224e-16
[2,] -3.079134e-16  1.884947e-01 -2.012279e-16 1.769418e-16
[3,] -1.040834e-17 -1.526557e-16  7.590489e-02 1.353084e-16
[4,] -2.417771e-17  2.065405e-16  1.474515e-16 9.765007e-03
```

Ortonormales (incorrelacionados) respecto a la métrica de **S**

```
> t(V2) %*% S %*% V2

              [,1]         [,2]          [,3]          [,4]
[1,]  1.000000e+00 1.387779e-17  2.775558e-17 -9.020562e-17
[2,] -3.989864e-17 1.000000e+00  0.000000e+00  1.162265e-16
[3,] -2.775558e-17 2.775558e-17  1.000000e+00  2.775558e-17
[4,] -8.326673e-17 9.714451e-17 -5.551115e-17  1.000000e+00
```

Coordenadas canónicas para las medias centradas

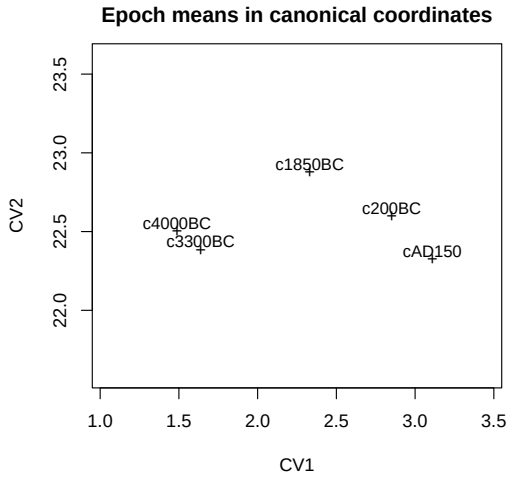
```
> round(X %*% V2,3)

          [,1]   [,2]   [,3]   [,4]
c4000BC -0.795 -0.033  0.110 -0.057
c3300BC -0.645 -0.153 -0.092  0.059
c1850BC  0.048  0.340 -0.117 -0.008
c200BC   0.568  0.059  0.185  0.041
cAD150   0.825 -0.212 -0.087 -0.036
```

Coordenadas canónicas para las medias (no centradas)

```
> Y <- means %*% V2; round(Y,3)

          [,1]   [,2]  [,3]   [,4]
c4000BC 1.489 22.497 4.520 36.015
c3300BC 1.638 22.377 4.318 36.131
c1850BC 2.332 22.870 4.293 36.063
c200BC  2.852 22.589 4.595 36.112
cAD150  3.109 22.319 4.323 36.035
```

Representación canónica (ambos ejes en la misma escala)

```
> require(MASS)
> eqscplot(Y[,1:2],xlim=c(1,3.5),ylim=c(22.2,23),pch="+",lwd=4,
+ xlab="CV1",ylab="CV2")
> title(main="Epoch means in canonical coordinates",line=1)
> text(Y[,1],Y[,2]+0.06,labels=rownames(means),cex=0.9)
```

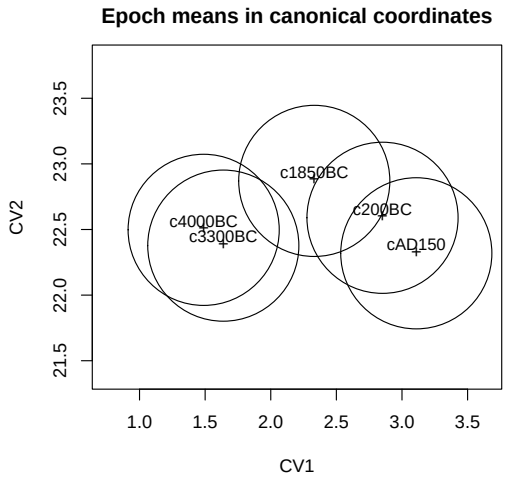


Radio de confianza al 95 %

```
> alpha <- 0.05
> R2 <- (nT-g)*p/(nT-g-p+1)*qf(1-alpha,p,nT-g-p+1)
> R <- sqrt(R2/n.i); round(R,2)

c4000BC c3300BC c1850BC  c200BC  cAD150
   0.58    0.58    0.58    0.58    0.58
```

Representación de los círculos de confianza con los dos ejes en la misma escala.

```
> eqscplot(Y[,1:2],pch="+",lwd=4,xlim=c(0.7,3.7),xlab="CV1",ylab="CV2")
> title(main="Epoch means in canonical coordinates",line=1)
> text(Y[,1],Y[,2]+0.07,labels=rownames(means),cex=0.9)
> symbols(Y[,1],Y[,2],circles=R,inches=FALSE,add=TRUE)
```



Análisis canónico discriminante

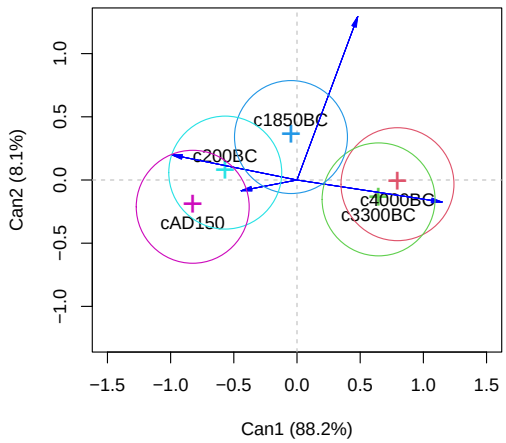
```
> library(candisc)
> skulls.can <- candisc(skulls.manova); skulls.can


Canonical Discriminant Analysis for EPOCH:

     CanRsq Eigenvalue Difference  Percent Cumulative
1 0.2982926  0.4250954     0.3861 88.22718     88.227
2 0.0375351  0.0389989     0.3861  8.09410     96.321
3 0.0154616  0.0157045     0.3861  3.25941     99.581
4 0.0020163  0.0020203     0.3861  0.41932    100.000

Test of H0: The canonical correlations in the
current row and all that follow are zero

  LR test stat approx F numDF  denDF  Pr(> F)
1      0.66359   3.9009    16 434.45 7.01e-07 ***
2      0.94567   0.8982     9 348.18   0.5270
3      0.98255   0.6364     4 288.00   0.6369
4      0.99798   0.2930     1 145.00   0.5892
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Sólo es significativo el primer eje canónico.

La función `plot()` de un objeto `candisc` dibuja por defecto los *scores* como puntos en los ejes canónicos y unas elipses que cubren el 68 % de los datos para cada grupo. También se representan las posiciones de las medias de cada grupo.

En el siguiente gráfico se representan los dos primeros ejes canónicos, las medias de cada grupo y los círculos de confianza al 95 %. Para mayor claridad se han eliminado los puntos de todas las observaciones.

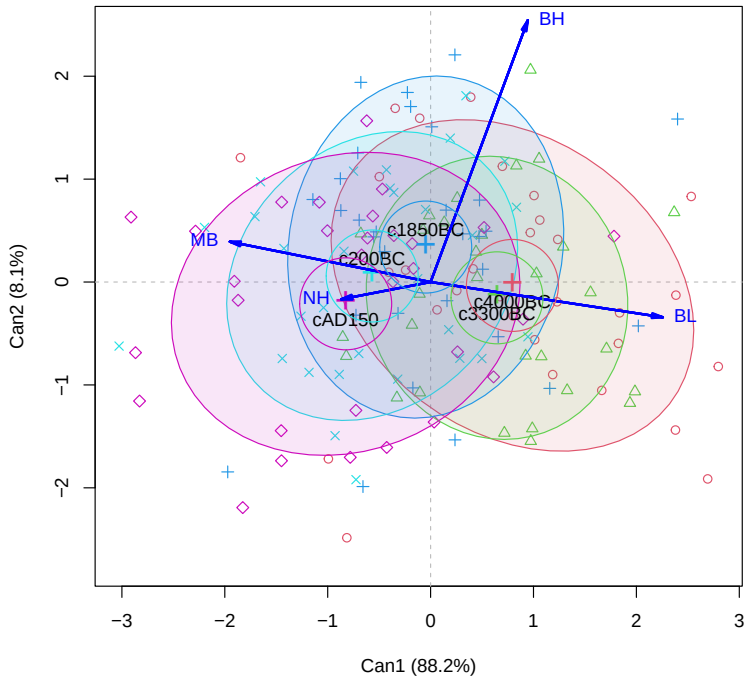
```
> plot(skulls.can, type="n", var.lwd=0, var.labels=NULL,
+                  xlim=c(-1.5,1.5), ylim=c(-1,1))
```



El gráfico es idéntico al que hemos obtenido antes, salvo que el primer eje está cambiado de signo.

También se pueden ver las relaciones entre las variables observadas y los ejes canónicos (de forma similar a un biplot). Cada vector se define por las correlaciones (coeficientes de estructura) que tiene respecto a los ejes canónicos.

```
> plot(skulls.can, ellipse=TRUE, var.lwd=2)
```



Del análisis canónico de poblaciones sobre las medias de las cinco poblaciones, usando la matriz de covarianzas común, resulta una representación significativa sobre un solo eje canónico que representa la evolución en las medidas craneales de las muestras. Esta escala es colineal con el paso del tiempo puesto que las medias se ordenan cronológicamente. Los círculos de confianza al 95 % muestran como, si bien estas poblaciones se solapan entre ellas, las más alejadas en el tiempo se hallan bien separadas mostrando diferencias significativas. El primer eje se explica principalmente por la variable MB, y NH en menor medida, en un sentido y BL en el otro. El segundo eje está muy relacionado con la variable BH.

c) *Calcular las distancias de Mahalanobis entre cada pareja de épocas.*

Para ello, considerar la matriz de covarianzas común

$$\hat{\mathbf{S}} = \frac{29\hat{\mathbf{S}}_1 + 29\hat{\mathbf{S}}_2 + 29\hat{\mathbf{S}}_3 + 29\hat{\mathbf{S}}_4 + 29\hat{\mathbf{S}}_5}{145}$$

donde $\hat{\mathbf{S}}_1, \ldots, \hat{\mathbf{S}}_5$ *son las matrices de covarianzas en cada época.*

Recordemos que `means` contiene las medias de las poblaciones y que `S` contiene la matriz de covarianzas común.

La siguiente función calcula la matriz de distancias de Mahalanobis al cuadrado entre las filas de una matriz de datos

```
> D2M <- function(X,S){
+   S.inv <- solve(S)
+   n <- nrow(X)
+   D <- matrix(0,nrow=n,ncol=n)
+   for(i in 1:(n-1)) for(j in (i+1):n) D[i,j] <- t(X[i,]-X[j,]) %*% S.inv %*% (X[i,]-X[j,])
+   D <- D + diag(0,n) + t(D)
+   rownames(D) <- colnames(D) <- rownames(X)
+   D
+ }
```

de modo que

```
> d2Mahal <- D2M(means,S)
> round(d2Mahal,4)

        c4000BC c3300BC c1850BC c200BC cAD150
c4000BC  0.0000  0.0910  0.9031 1.8811 2.6968
c3300BC  0.0910  0.0000  0.7289 1.5940 2.1757
c1850BC  0.9031  0.7289  0.0000 0.4431 0.9109
c200BC   1.8811  1.5940  0.4431 0.0000 0.2193
cAD150   2.6968  2.1757  0.9109 0.2193 0.0000
```

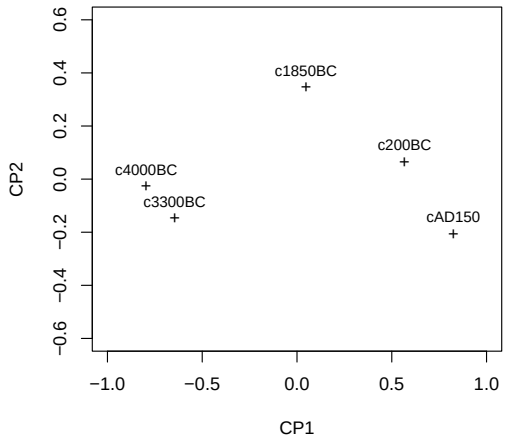
d) *Realizar un escalado multidimensional con la matriz de distancias obtenida en el apartado anterior y representar las cinco épocas con las dos coordenadas principales.*

Escalado multidimensional de la matriz de distancias de Mahalanobis

```
> mds <- cmdscale(as.dist(sqrt(d2Mahal)),eig=T); mds

$points
               [,1]        [,2]
c4000BC -0.79515585  0.03311963
c3300BC -0.64549355  0.15347046
c1850BC  0.04762133 -0.33983601
c200BC   0.56774779 -0.05868023
cAD150   0.82528029  0.21192615

$eig
[1] 2.054628e+00 1.884947e-01 7.590489e-02 9.765007e-03 3.167755e-17

$x
NULL

$ac
[1] 0

$GOF
[1] 0.9632127 0.9632127
```

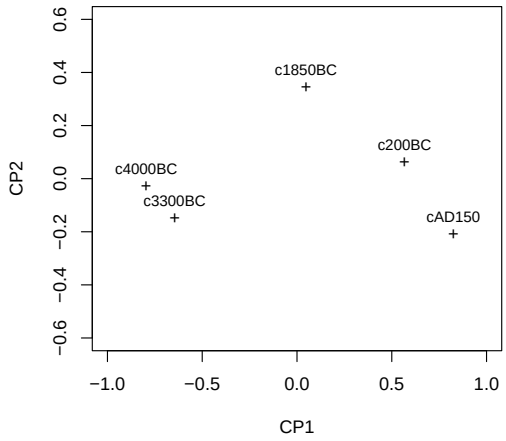
Representación gráfica de las coordenadas principales de la matriz de distancias de Mahalanobis, en dos dimensiones.

```
> plot(mds$points[,1],-mds$points[,2],pch="+",
+                     xlim=c(-1,1),ylim=c(-0.6,0.6),xlab="CP1",ylab="CP2")
> text(mds$points[,1],-mds$points[,2]+0.07,labels=rownames(means),cex=0.8)
```



Representación de las coordenadas canónicas de las medias centradas

```
> Z <- (X %*% V2)[,1:2]
> plot(Z, pch="+",
+         xlim=c(-1,1),ylim=c(-0.6,0.6),xlab="CP1",ylab="CP2")
> text(Z[,1],Z[,2]+0.07,labels=rownames(means),cex=0.8)
```



Con esto se ha mostrado como la distancia de Mahalanobis entre cada par de poblaciones (i, j) coincide con la distancia euclídea (la que se observa sobre el gráfico) entre las filas (i, j) de la matriz de coordenadas canónicas. Esto se justifica por el uso de la métrica de la matriz de covarianzas común, en ambos casos.

Ejercicio 7 Escalado multidimensional no métrico

En los problemas de escalado no métrico se parte de una matriz de similaridades, disimilaridades o distancias (δ_{ij}) entre objetos o individuos que se ha obtenido por la estimación directa de un juez o grupo de expertos, o por estimación por rangos o rangos por pares, es decir, sin una métrica a partir de unas variables observadas.
La distancia así obtenida se supone que está relacionada con la verdadera distancia pero de una manera compleja. Es decir, seguramente los expertos utilizan en sus valoraciones ciertas variables o dimensiones, además de elementos de error y variabilidad personal. Así, la verdadera distancia se puede considerar una función monótona creciente de las distancias dadas

$$\hat{d}_{ij} = \varphi(\delta_{ij})$$

Escalado multidimensional isotónico
Una versión de escalado multidimensional no métrico se basa en una vieja idea de Kruscal y Shepard (1960). Dada una disimilaridad (δ_{ij}), se trata de elegir una configuración de puntos con su distancia euclídea (d_{ij}) que minimize

$$\text{stress}^2 = \sum_{i<j}[\varphi(\delta_{ij} - d_{ij})]^2 / \sum_{i<j} d_{ij}^2$$

para ambas, la configuración y la función creciente φ. Así, la localización, la rotación, la reflexión y la escala de la configuración de puntos son todas indeterminadas.
*En **R** tenemos la función `isoMDS()` del paquete MASS en la que se ha implementado un algoritmo de optimización, lógicamente, un poco lento.*
Aplicar este escalado isotónico a la matriz de distancias del ejercicio 1.

El dato de partida es la matriz de distancias

```
> D <- matrix(c(0,7,5,9,5,7,9,
+               7,0,4,6,4,6,7,
+               5,4,0,3,4,5,6,
+               9,6,3,0,3,2,2,
+               5,4,4,3,0,5,4,
+               7,6,5,2,5,0,4,
+               9,7,6,2,4,4,0), ncol=7, byrow=T)
> colnames(D) <- rownames(D) <- LETTERS[1:7]
```

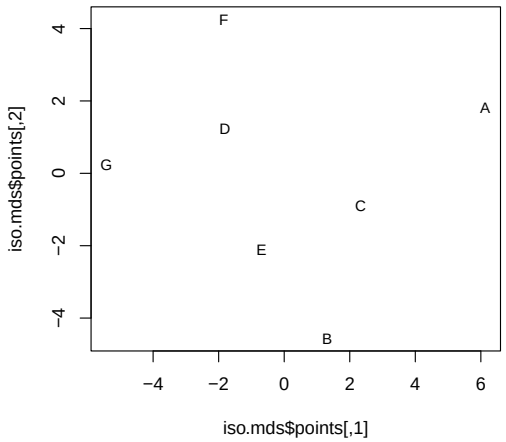
Escalado multidimensional isotónico

```
> require(MASS)
> iso.mds <- isoMDS(D); iso.mds

initial  value 16.599527
iter   5 value 10.723704
iter  10 value 7.461641
final  value 7.086010
converged
$points
        [,1]       [,2]
A  6.1365452  1.8289088
B  1.3142377 -4.5569073
C  2.3287162 -0.8903469
D -1.8105626  1.2392767
E -0.6958321 -2.1193530
F -1.8422913  4.2480542
G -5.4308132  0.2503675
```

```
$stress
[1] 7.08601
```

Representación en coordenadas principales de los animales

```
> plot(iso.mds$points,type="n")
> text(iso.mds$points[,1],iso.mds$points[,2],labels=rownames(D),cex=0.8)
```



Distancia euclídea entre los puntos resultantes del escalado multidimensional isotónico

```
> dist(iso.mds$points)

          A         B         C         D         E         F
B  8.002081
C  4.679093  3.804318
D  7.968952  6.584841  4.654989
E  7.891144  3.159438  3.264713  3.538788
F  8.337511  9.353664  6.618192  3.008945  6.469795
G 11.674570  8.282850  7.842928  3.752886  5.294867  5.372056

> round(fd <- dist(iso.mds$points),4)

        A       B       C       D       E       F
B  8.0021
C  4.6791  3.8043
D  7.9690  6.5848  4.6550
E  7.8911  3.1594  3.2647  3.5388
F  8.3375  9.3537  6.6182  3.0089  6.4698
G 11.6746  8.2828  7.8429  3.7529  5.2949  5.3721
```

Distancia proporcionada por el "experto"

```
> as.dist(D)

  A B C D E F
B 7
C 5 4
D 9 6 3
E 5 4 4 3
F 7 6 5 2 5
G 9 7 6 2 4 4
```

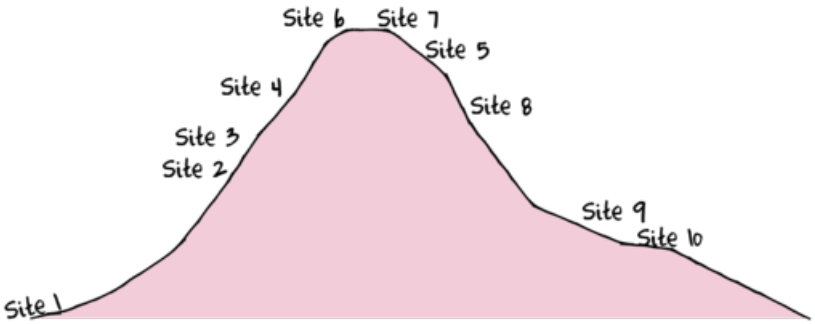
Diferencias entre la distancia del experto y la aproximada euclídea

```
> round(as.dist(D)-fd,2)

      A     B     C     D     E     F
B -1.00
C  0.32  0.20
D  1.03 -0.58 -1.65
E -2.89  0.84  0.74 -0.54
F -1.34 -3.35 -1.62 -1.01 -1.47
G -2.67 -1.28 -1.84 -1.75 -1.29 -1.37
```

Ejercicio 8 Escalado multidimensional no métrico en Ecología

Una de las principales aplicaciones del MDS en Ecología es el estudio de los perfiles entre lugares de muestreo (sites) según la abundancia de ciertas especies.
Consideremos los datos simulados del ejemplo en la página web
`https://www.flutterbys.com.au/stats/tut/tut15.1.html`
Estos datos se pueden generar con el código que allí se explicita o se pueden cargar así:

```
> load("tut15_1_data.Rdata")
> the.data

       Sp1 Sp2 Sp3 Sp4 Sp5 Sp6 Sp7 Sp8 Sp9 Sp10
Site1    5   0   0  65   5   0   0   0   0    0
Site2    0   0   0  25  39   0   6  23   0    0
Site3    0   0   0   6  42   0   6  31   0    0
Site4    0   0   0   0   0   0   0  40   0   14
Site5    0   0   6   0   0   0   0  34  18   12
Site6    0  29  12   0   0   0   0   0  22    0
Site7    0   0  21   0   0   5   0   0  20    0
Site8    0   0   0   0  13   0   6  37   0    0
Site9    0   0   0  60  47   0   4   0   0    0
Site10   0   0   0  72  34   0   0   0   0    0
```



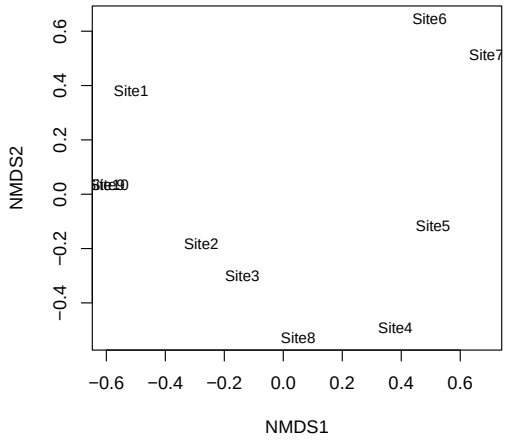
El escalado multidimensional no métrico puede realizarse usando la función `isoMDS()` *del paquete* `MASS`*. Esta función necesita como entrada las disimilaridades entre observaciones (sites). La función* `vegdist()` *del paquete* `vegan` *contiene algunas disimilaridades apropiadas en ecología de comunidades (donde pueden haber muchos ceros). El valor por defecto es la disimilaridad de Bray-Curtis, también conocida como como disimilaridad de Steinhaus, o en Finlandia como índice de Sørensen.*

a) *Calcular el NMDS isométrico en 2 dimensiones con la disimilaridad de Bray-Curtis para estos datos y representar el mapa con las posiciones de las observaciones (mejor con los nombres de las sites). ¿Cual es el stress (en porcentaje) de la representación?*

```
> library(MASS)
> library(vegan)
> data.dist <- vegdist(the.data, "bray")
> nmds.iso <- isoMDS(data.dist, trace = FALSE)
```

El mapa lo podemos representar así:

```
> plot(nmds.iso$points[,1], nmds.iso$points[,2], type = "n",
+      xlab="NMDS1", ylab="NMDS2")
> text(nmds.iso$points[,1], nmds.iso$points[,2],
+      labels = rownames(the.data), cex=0.8)
```

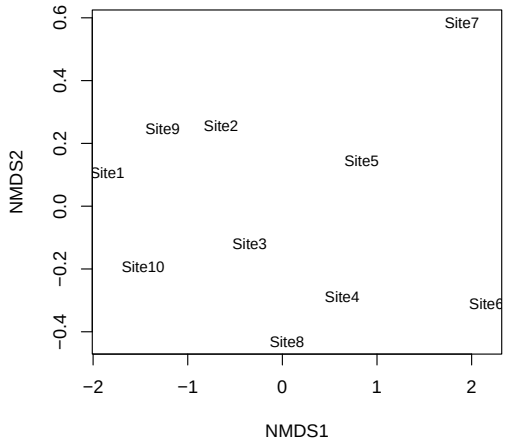


El *stress* es:

```
> nmds.iso$stress

[1] 4.649589
```

b) *El paquete* `vegan` *dispone de la función* `metaMDS()` *más flexible. Aunque el algoritmo es distinto al de la función* `isoMDS()`*, podemos utilizar el mismo* `engine`*, la disimilaritad de Bray, la configuración inicial* `cmdscale()` *y* `autotransform=FALSE` *para obtener un resultado parecido al del apartado anterior. Dibujar el mapa y calcular el stress. ¿Mejor que el anterior?*

```
> nmds.iso2 <- metaMDS(the.data, distance="bray", engine="isoMDS",
+                      y = cmdscale(data.dist),
+                      autotransform=FALSE, trace = FALSE)
> plot(nmds.iso2$points[,1], -nmds.iso2$points[,2], type = "n",
+      xlab="NMDS1", ylab="NMDS2")
> text(nmds.iso2$points[,1], -nmds.iso2$points[,2],
+      labels = rownames(the.data), cex=0.8)
> nmds.iso2$stress

[1] 0.1215281
```

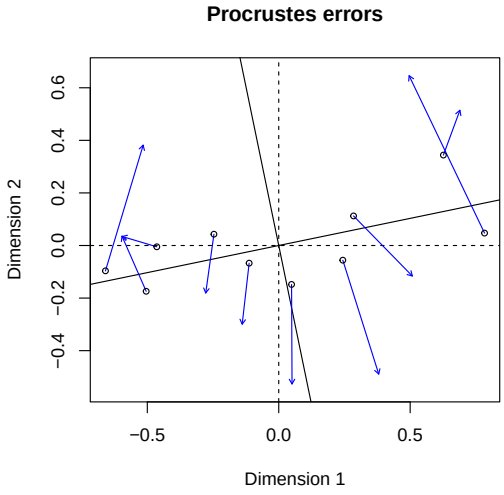


El *stress* ha mejorado, de forma que este segundo mapa es una mejor representación euclídea en dos dimensiones de las disimilaridades iniciales.

Nota: Para entender mejor la comparación del siguiente apartado, he cambiado el signo del segundo eje.

c) *Para comparar dos configuraciones o mapas, en el paquete* `vegan` *disponemos de la maravillosa función* `procrustes()`*. Comparar los dos mapas que tenemos con esta función y su* `plot()` *por defecto.*

```
> proc <- procrustes(nmds.iso,nmds.iso2)
> plot(proc)
```

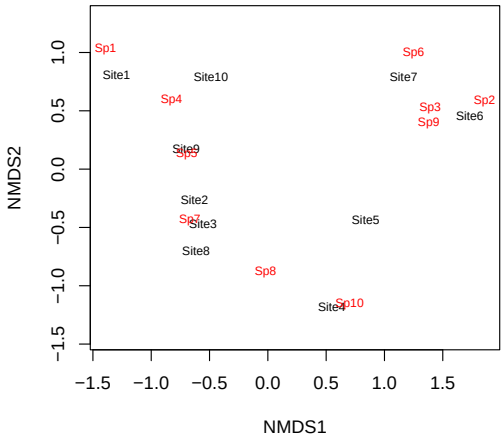


No son tan distintos como podría parecer al principio.

d) *Finalmente vamos a calcular un NMDS con la función* `metaMDS()` *y el* `engine = "monoMDS"`*, dejando los otros parámetros por defecto. Representar el mapa y calcular el stress (multiplicado por* 100 *para que sea un porcentaje).*

Observar que la solución es muy similar a la que se obtiene en la web con la función `ordiplot()`*.*

```
> nmds.mono <- metaMDS(the.data, distance="bray", engine="monoMDS", trace = FALSE)
> nmds.mono$stress*100

[1] 0.009388134

> plot(nmds.mono, type="text")
```



El resultado es muy similar al que podemos ver en la web.

En cuanto a la interpretación, es evidente que hay una clara aproximación de las observaciones según la altura en la montaña. Para mostrarla deberíamos contar con las variables de entorno de las que en este ejercicio no disponemos.