INTRODUCCION AL ANALISIS MULTIVARIADO

Escalamiento Multidimensional

1. Cargue la base, y nombrela "base" con el siguiente comando

```
base<-cbind(c(3,5,6,1,4,2,0,0,7,2),
            c(4,1,2,1,7,2,4,6,6,1),
            c(4,1,0,1,3,5,1,4,5,4),
            c(6,7,2,0,6,1,1,3,1,3),
            c(1,3,6,3,2,0,1,5,4,1))
```

2. Calcule la matriz de distancias euclídeas y nombrela "d"

```
d = dist(base)
```

3. Realize el escalamiento multidimensional y nombrelo "cmds" con el siguiente comando:

'cmds<-cmdscale(d,k=5,eig=TRUE,x.ret=TRUE)', donde

- d <- es la matriz de distancias
- k <- indica el número de dimensiones requeridas
- eig <- indica si debería devolver los valores propios
- X.ret <- devuelve parte de la matriz B (La matriz B se obtiene multiplicando (-1/2)*cmds$x)

```
cmds = cmdscale(d,k=5,eig=TRUE,x.ret=TRUE)
```

Tiene que agregarse todos estos argumentos porque caso contrario solo da los primeros vectores propios

4. Calcule la matriz B y compárela con la que se obtiene directamente en R. Los elementos de la matriz B se pueden obtener de acuerdo a las fórmulas vistas en la presentación en clase o también utilizando la siguiente fórmula: $B = -\frac{1}{2}CD^2C$. Donde D^2 es la matriz de distancias al cuadrado y C está definida como $C = I - \frac{1}{n}J_n$. Aquí I es la matriz identidad de dimensión $n \times n$ y J_n es una matriz de unos, de dimensión $n \times n$.

```
n = dim(base)[1]
D = as.matrix(dist(base,diag = T,upper = T))
Jn = matrix(1,n,n)
C = diag(n)-(1/n)*Jn

B = -0.5*C%*%(D^2)%*%C
```

5. Obtenga los valores propios con el comando 'cmds$eig' y también obténgalos de la matriz B que calculó en el punto anterior y compruebe que son los mismos.

```
round(cmds$eig,2)
```

```
##  [1] 75.19 58.81 49.61 30.43 10.37  0.00  0.00  0.00  0.00  0.00
```

```
eB = eigen(B)
round(eB$values,2)
```

```
##  [1] 75.19 58.81 49.61 30.43 10.37  0.00  0.00  0.00  0.00  0.00
```

6. Calcule la bondad de ajuste usando los dos criterios vistos en clase y escoja un k número de dimensiones según esos criterios.

```
round(abs(cmds$eig)/sum(abs(cmds$eig))*100,2)
```

```
##  [1] 33.51 26.21 22.11 13.56  4.62  0.00  0.00  0.00  0.00  0.00
```

```
33.51+26.21
```

```
## [1] 59.72
```

```
33.51+26.21+22.11
```

```
## [1] 81.83
```

Con el criterio 1 se necesitan 3 coordenadas para tener más de 80%.

```
round(cmds$eig^2/sum(cmds$eig^2)*100,2)
```

```
##  [1] 44.85 27.43 19.52  7.34  0.85  0.00  0.00  0.00  0.00  0.00
```

```
44.85+27.43
```

```
## [1] 72.28
```

```
44.85+27.43+19.52
```

```
## [1] 91.8
```

Con el criterio 2 también se necesitan 3 coordenadas para tener más de 80%.

7. Calcule el stress para cada k dimensiones, $k = 1, \ldots, 5$, y escoja un valor de k de acuerdo a este criterio. Las nuevas coordenadas se obtienen con el comando 'cmds$points'.

```
coor2=cmds$points[,1:2]
d2=as.matrix(dist(coor2))
d=as.matrix(d)
sqrt(sum((d-d2)^2)/sum(d^2))
```

```
## [1] 0.3371175
```

```
coor3=cmds$points[,1:3]
d3=as.matrix(dist(coor3))

sqrt(sum((d-d3)^2)/sum(d^2))
```

```
## [1] 0.170789
```

```
coor4=cmds$points[,1:4]
d4=as.matrix(dist(coor4))

sqrt(sum((d-d4)^2)/sum(d^2))
```

```
## [1] 0.05019229
```

Con el criterio de stress se recomiendan 4 coordenadas para que el ajuste sea bueno.

8. ¿Coinciden los tres criterios? ¿Cuál valor de k seleccionaría?

Coinciden los primeros 2 criterios. Seleccionaría 3 ya que con el stress se llega a 0.17 cuando se tienen 3 coordenadas y eso está entre bueno y aceptable.

9. Calcule los puntos en las nuevas coordenadas y compárelos con los valores obtenidos automáticamente en R (cmds$points). Haga un gráfico de esos puntos.

Para graficarlos se van a usar solo 2 coordenadas aunque no den un buen ajuste.

```r
cmds$points[,1:2]
```

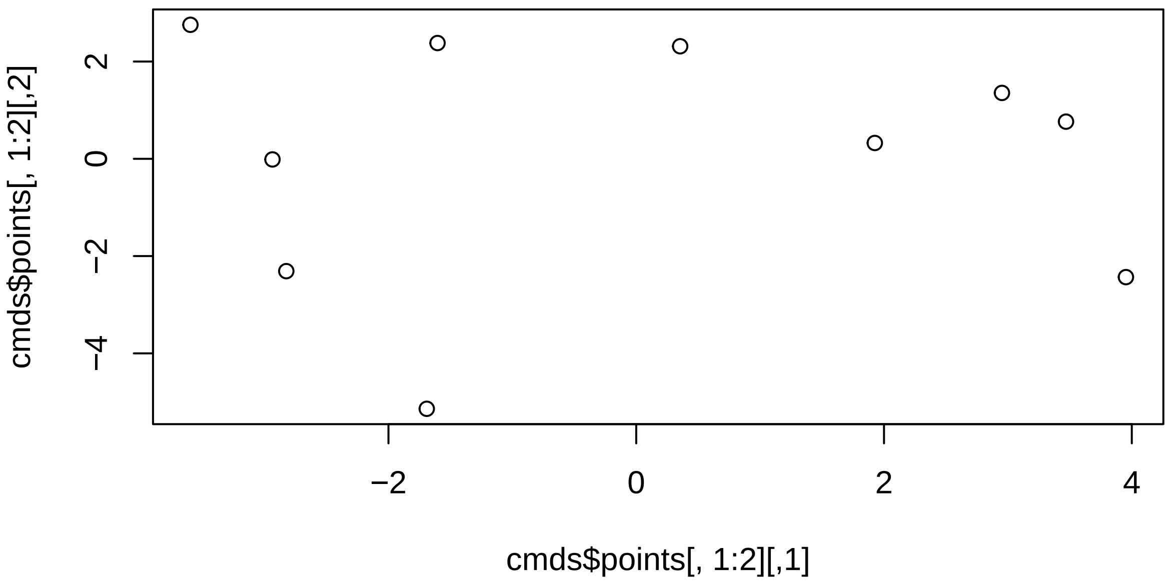
```
##              [,1]        [,2]
##  [1,] -1.6038325  2.38060903
##  [2,] -2.8246377 -2.30937202
##  [3,] -1.6908272 -5.13970089
##  [4,]  3.9527719 -2.43233961
##  [5,] -3.5984894  2.75538195
##  [6,]  2.9520356  1.35475175
##  [7,]  3.4689928  0.76411068
##  [8,]  0.3545235  2.31408566
##  [9,] -2.9362323 -0.01279597
## [10,]  1.9256952  0.32526941
```

```r
eB$vectors[,1:2]
```

```
##               [,1]         [,2]
##  [1,] -0.18496398  0.310440748
##  [2,] -0.32575486 -0.301151163
##  [3,] -0.19499676 -0.670237142
##  [4,]  0.45585835 -0.317186619
##  [5,] -0.41500028  0.359312607
##  [6,]  0.34044719  0.176664938
##  [7,]  0.40006592  0.099643028
##  [8,]  0.04088587  0.301765840
##  [9,] -0.33862466 -0.001668644
## [10,]  0.22208321  0.042416406
```

No dan igual, pero se pueden graficar ambos y se obtiene el mismo gráfico.

```r
plot(cmds$points[,1:2])
```




```
plot(eB$vectors[,1:2])
```