Ex MDS

Vicent Caselles Ballester

2024-04-17

Exercici 1

```
dat <- 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)
all(dat == t(dat)) # to make sure I haven't made mistakes
```

```
## [1] TRUE
```

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

From [1] :

> **Theorem**: Let D be a distance matrix and define K by (2). Then D is Euclidean if and only if K is positive semi-definite.

K is the Kernel matrix, refered to as B in Everitt, 2011. Therefore, let's use some of the functions I've created to check if a symmetric matrix is positive definite.

```
source('../../funcs/sym.R')
require(ade4); require(matrixcalc)
```

```
## Loading required package: ade4
```

```
## Loading required package: matrixcalc
```

```
positive_definite(dat)
```

```
## [1] FALSE
```

```
is.positive.semi.definite(dat)
```

```
## [1] FALSE
```

```
is.euclid(as.dist(dat))
```

```
## [1] FALSE
```

With this proof, I think we can be somewhat sure that the distances are not euclidean.

[1] https://www.math.uwaterloo.ca/ aghodsib/courses/f10stat946/notes/lec10-11.pdf

About the first part, we can calculate B using some functions I've written.

```
source('../../funcs/MDS.R')
D <- dat
B <- KernelMatrix(D**2)
B2 <- B_from_D(D)

max(abs(B-B2)) # We see that both functions using different approaches accomplish the same
```

```
## [1] 7.105427e-15
```

Eigenvalues of B:

```
evB <- eigen(B)$values
sum(evB < 0)
```

```
## [1] 2
```

We see that two eigenvalues of B are < 0, therefore B is not positive semi-definite, therefore D is not euclidean.

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`.

I'm gonna choose the first option. We select fist two eigen{values, vectors} pairs of B.

```
evecs_2 <- eigen(B)$vectors[,1:2]
evalues_2 <- diag(sqrt(eigen(B)$values[1:2]))
new_X <- evecs_2%*%evalues_2

max(abs(new_X - cmdscale(dat, k=2)))
```

```
## [1] 5.107026e-15
```

Let's check how different the new euclidean distances are using the reduced data matrix (`new_X`).

```
max(abs(dat - as.matrix(dist(new_X))))
```

```
## [1] 2.934933
```

We see that there is a sufficiently enough large distance difference between the points in the original space and the ones in the new reduced space.

Using the criteria from Everitt 2011:

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

```
## [1] 0.7881461
```

They also mention the following criteria in case B is not positive definite:

```
sum(abs(eigen(B)$values[1:2]))/sum(abs(eigen(B)$values))
```

```
## [1] 0.7062874
```

```
sum(eigen(B)$values[1:2] ** 2)/sum(eigen(B)$values ** 2)
```

```
## [1] 0.9135125
```

We see that the results differ a lot.

Exercici 2

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

```
data(crabs, package='MASS')
X <- data.matrix(crabs[, 4:8])
```

Theoretically, PCA analysis and MDS are equivalent as long as you select the k largest eigen{values, vectors} when constructing the new X.

```
require(matrixcalc)
D <- as.matrix(dist(X))
B <- B_from_D(D)
evalues <- eigen(B)$values
evectors <- eigen(B)$vectors
k <- min(matrix.rank(t(X)%*%X), sum(evalues>0))
```

```
k_evectors <- evectors[, 1:k, drop=FALSE]
k_evalues <- evalues[1:k]

new_X_MDS <- k_evectors%*%diag(sqrt(k_evalues))
max(abs(as.matrix(dist(new_X_MDS)) - D)) # cool
```

```
## [1] 1.070255e-13
```

```
new_X_PCA <- prcomp(X)$x
max(abs(new_X_PCA - new_X_MDS))
```

```
## [1] 3.02109e-13
```

Cool.

Exercici 3

Too long to paste here

First let's define the data.

```
X <- 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, 0.15, 0.67,
              0.23, 0, 0.28, 0.49,
              0.3, 0, 0.06, 0.64,
              0.1, 0.06, 0.13, 0.71,
              0.27, 0.04, 0.06, 0.63,
              0.21, 0.05, 0.2, 0.54), ncol = 4, byrow=T)
```

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

Function definition:

```
bhattacharyya <- function(X){
  # X: data matrix
  n <- nrow(X)
  D <- matrix(0, nrow=n, ncol=n)
```

```
  for (i in 1:(n-1)){
    for (j in (i+1):n){
      pi <- X[i,]
      pj <- X[j,]
      D[i,j] <- acos(sum(sqrt(pi*pj)))
      D[j,i] <- D[i,j]
    }
  }
  D
}
```

Calculation on our data:

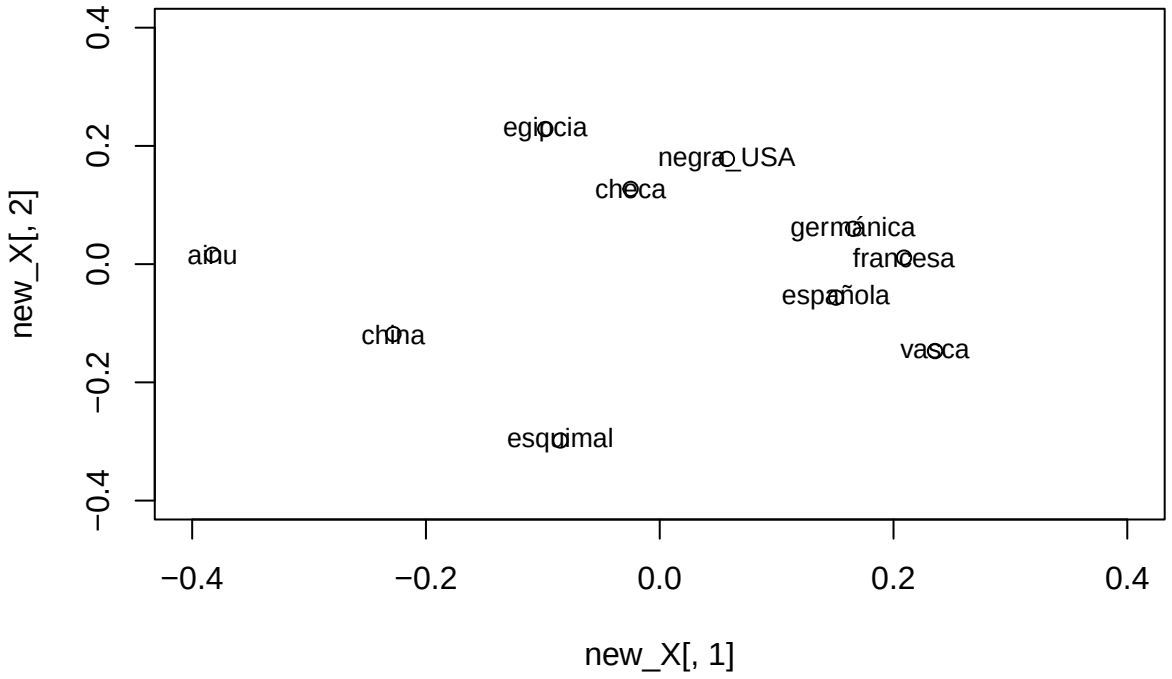
```
D_bhat <- bhattacharyya(X)
```

Same result as the one on the solution, albeit different methodology.

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

```
B <- B_from_D(sqrt(D_bhat))
evalues <- eigen(B)$values[1:2]
evectors <- eigen(B)$vectors[, 1:2]

new_X <- evectors %*% diag(sqrt(evalues))
plot(new_X[, 1], new_X[, 2], ylim=c(-0.4, 0.4), xlim = c(-0.4, 0.4))
labels = c("francesa","checa","germánica","vasca","china",
           "ainu","esquimal","negra_USA","española","egipcia")
text(new_X[, 1], new_X[, 2], labels=labels,cex=0.8)
```



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.

Following the same logic as before:

```
require(ade4)
is.positive.semi.definite(B)
```

```
## [1] TRUE
```

```
positive_definite(B)
```

```
## [1] TRUE
```

```
is.euclid(as.dist(sqrt(D_bhat)))
```

```
## [1] TRUE
```

```
sum(eigen(B)$values<0)
```

```
## [1] 0
```

Therefore, we can almost 100% certainly conclude that B is positive semi-definite. The dimension of the original space of the data is 4.

Second part of the question, same solution as before:

```
sum(eigen(B)$values[1:2])/sum(eigen(B)$values) * 100
```

```
## [1] 56.58885
```