Escalamiento multidimensional

Manuel Torres Acosta

Índice

Tarea **1**
- Datos originales . . . 2
- Datos ampliados . . . 4
  - Obtención de los datos . . . 4
  - Aplicación del modelo . . . 5
  - Correlaciones entre las coordenadas . . . 7
  - Reflexión . . . 8

Tarea

El objetivo de ésta práctica es interpretar la salida de un modelo de escalamiento multidimensional. Para ello, partiremos de unos datos dados para la práctica y aplicaremos la técnica sobre los mismos.

A continuación, extenderemos los datos añadiendo nuevas ciudades y deberemos interpretar la salida del modelo. Se indica que creemos un par de ciudades adicionales manualmente, pero en su lugar obtendremos un fichero con las ciudades más importantes de España a través de internet.

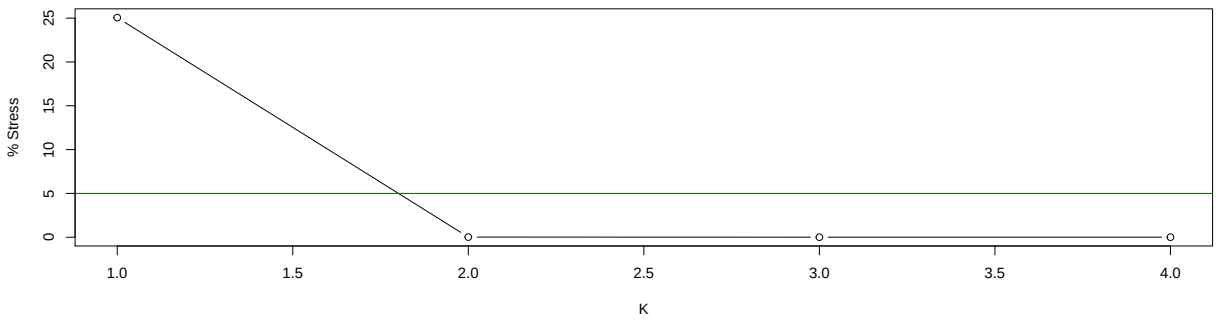
Datos originales

En primer lugar, cargamos los datos y aplicamos el modelo sobre los datos dados para la práctica

```
#Dir es el directorio en el que estan los datos
Datos <- read.table(paste(Dir, "/CIUDADES_MDS.dat",
                          sep = ""),
                    header = TRUE,
                    fill = TRUE)
Datos <- as.dist(Datos)
rownames(Datos) <- colnames(Datos)

#Aplicamos el modelo con diferentes valores de k
Resultados1 <- 1:4
for(i in 1:length(Resultados1)){
  Resultados1[i] = isoMDS(Datos, k = i,
                trace = FALSE)$stress
}

#Representamos los valores de ajuste
plot(Resultados1, type = "b",
     xlab = "K", ylab = "% Stress")
  abline(h = 5, col = "darkgreen")
```

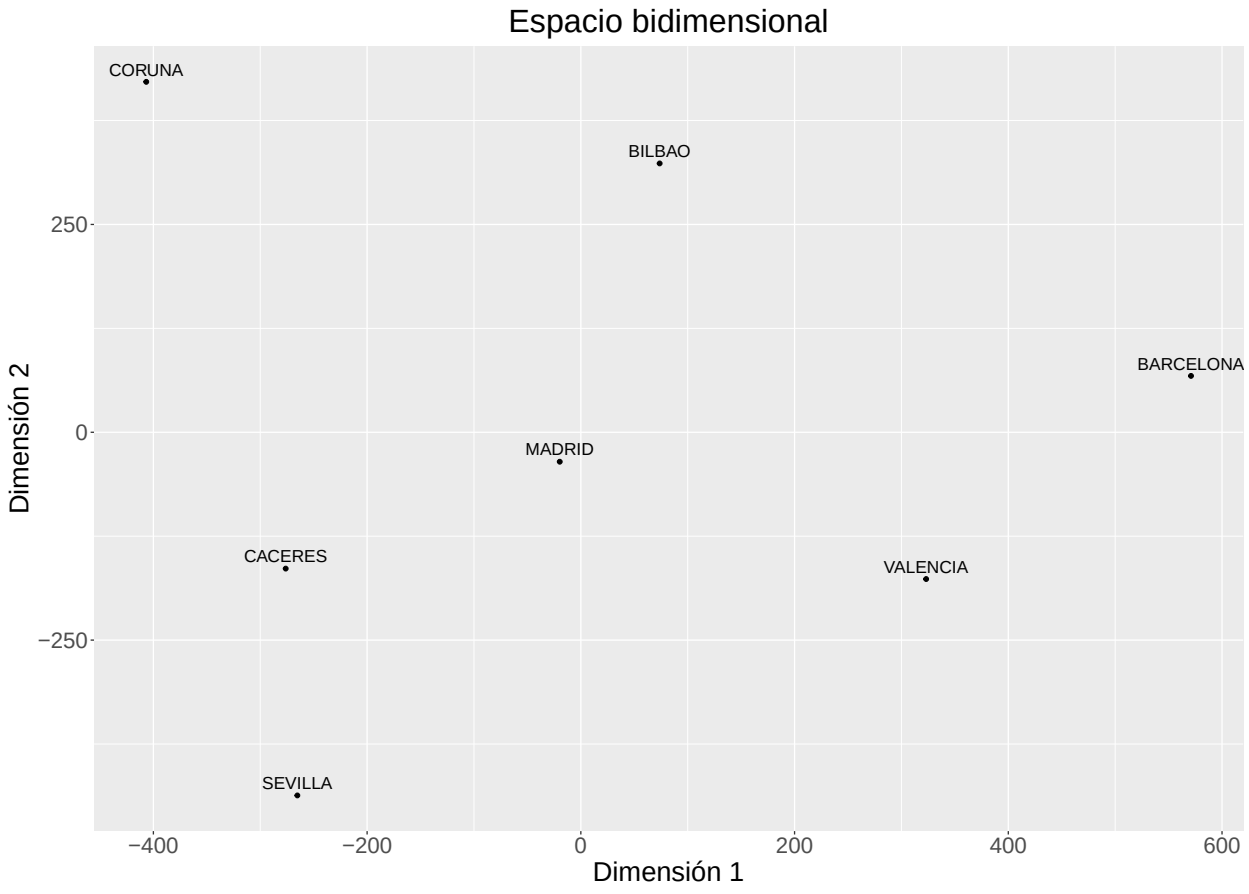


Vemos que con 2 dimensiones conseguimos un buen ajuste, por lo que definimos el modelo con 2 dimensiones.

```
mds <- isoMDS(Datos, k = 2,
              trace = FALSE)
```

A continuación, representamos las ciudades en un espacio bidimensional.

```r
DatosGrafico <- data.frame(
  Dim1 = mds$points[, 1],
  Dim2 = mds$points[, 2]
)
DatosGrafico <- DatosGrafico * -1
ggplot(DatosGrafico, aes(x = Dim1, y = Dim2)) +
  geom_point() +
  labs(x = "Dimensión 1",
       y = "Dimensión 2",
       title = "Espacio bidimensional") +
  geom_text(label = rownames(DatosGrafico),
            vjust = -0.5,
            size = 5) +
  theme(text = element_text(size = PlotFont),
        plot.title = element_text(hjust = 0.5))
```



Datos ampliados

A continuación repetimos las tareas añadiendo más ciudades. Lo primero es conseguir los datos. En ésta web tenemos disponible una tabla con las distancias entre muchas ciudades de España.

Dado que es importante que todas las distancias se calculen según la misma métrica, utilizaremos éstos datos exclusivamente. Para ello, primero tenemos de incorporar los datos a R.

Obtención de los datos

```
library(rvest)
library(dplyr)

web <- url %>%
  read_html() %>%
  html_nodes("table") %>%
  html_nodes("tr") %>%
  html_text()

LimpiaTexto <- function(texto){
  texto <- gsub("[[:space:]]+", ",", texto)
  texto <- unlist(strsplit(texto, ","))
  texto <- texto[-length(texto)]
  texto <- paste(texto, collapse = ",")

  texto
}

Datos <- unname(sapply(web, LimpiaTexto))
Datos <- Datos[2:length(Datos)]
```

A continuación, vertimos los datos a un archivo utilizando la función `writeLines()`. Se corrigieron algunas filas y se reorganizó la información manualmente utilizando una hoja de cálculo de Libre Office.

Aplicación del modelo

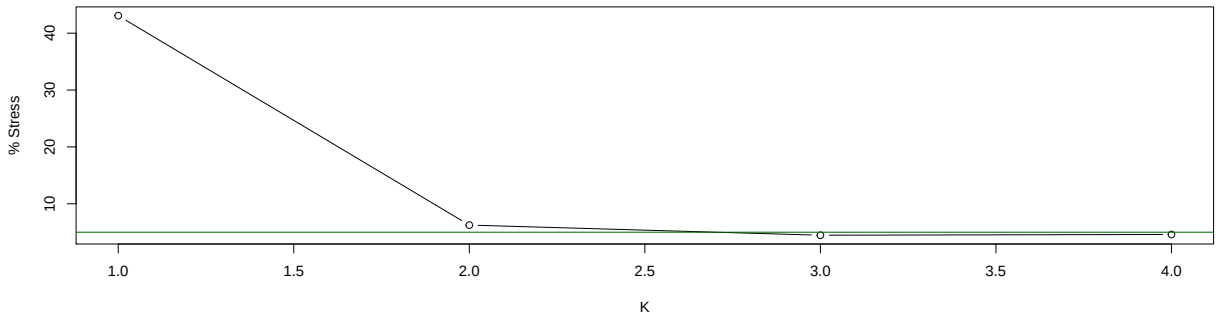
Una vez que tenemos los datos disponibles los leemos y repetimos el proceso anterior

```
Datos <- read.table(paste(Dir, "/CiudadesCompleto.csv",
                          sep = ""),
                    header = TRUE,
                    fill = TRUE,
                    sep = ",")

rownames(Datos) <- Datos$X
Datos$X <- NULL
DatosOriginal <- Datos
Datos <- as.dist(Datos)

#Aplicamos el modelo con diferentes valores de k
Resultados <- 1:4
for(i in 1:length(Resultados)){
  Resultados[i] = isoMDS(Datos, k = i,
                trace = FALSE)$stress
}

#Representamos los valores de ajuste
plot(Resultados, type = "b",
     xlab = "K", ylab = "% Stress")
  abline(h = 5, col = "darkgreen")
```

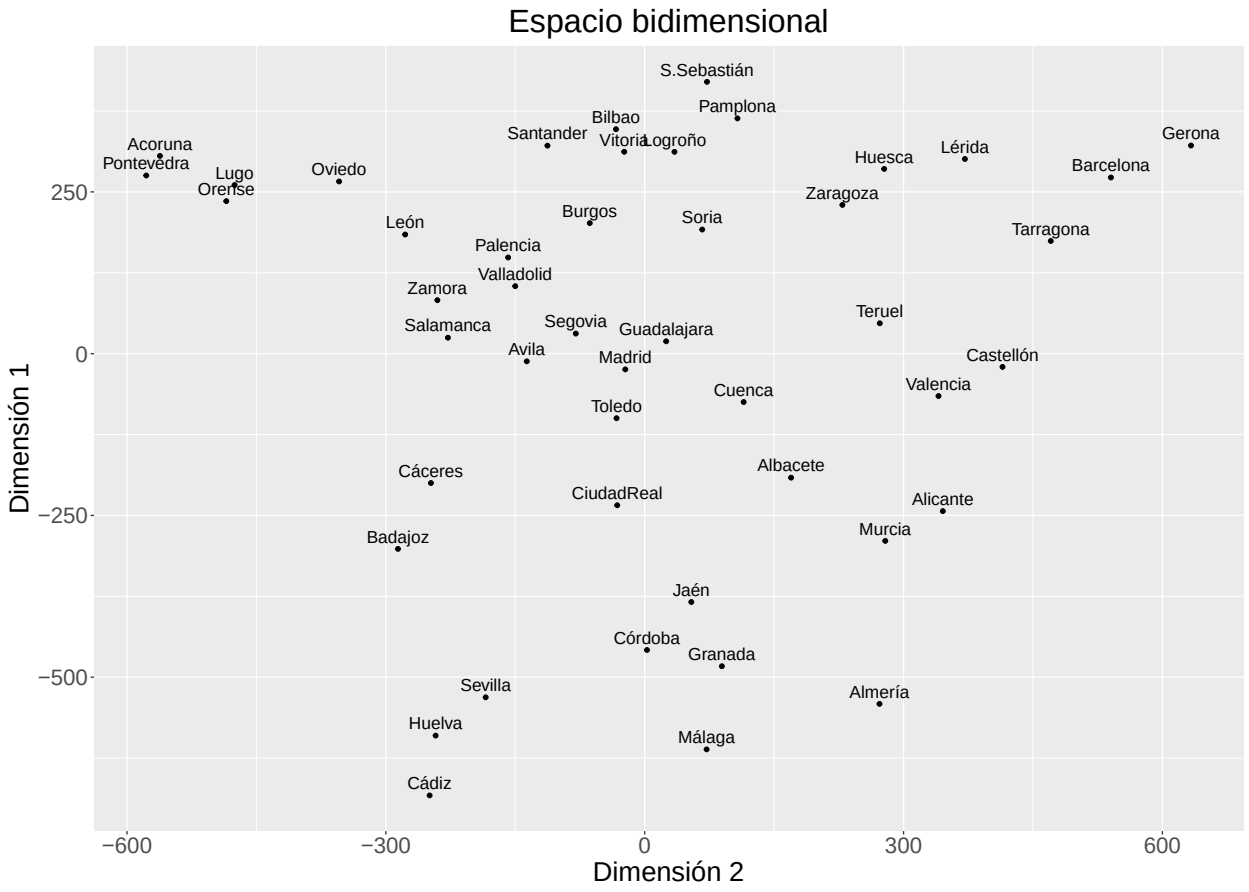


Nuevamente vemos que al ajustar el modelo con dos dimensiones obtenemos un buen resultado. Podemos concluir por tanto que **se mantiene el número de dimensiones**.

No obstante, el *stress* ha aumentado bastante. Con los datos originales adoptó un valor de **0.02**, mientras que con las 47 ciudades el *stress* vale **6.25**, que sigue siendo un valor excelente.

A continuación, representamos las ciudades en un espacio bidimensional.

```r
DatosGrafico <- data.frame(
  Dim1 = mds$points[, 1],
  Dim2 = mds$points[, 2]
)
DatosGrafico <- DatosGrafico
ggplot(DatosGrafico, aes(x = Dim2, y = Dim1)) +
  geom_point() +
  labs(x = "Dimensión 2",
       y = "Dimensión 1",
       title = "Espacio bidimensional") +
  geom_text(label = rownames(DatosGrafico),
            vjust = -0.5,
            size = 5) +
  theme(text = element_text(size = PlotFont),
        plot.title = element_text(hjust = 0.5))
```



Vemos que efectivamente el modelo es capaz de organizar las nuevas ciudades de forma satisfactoria.

Correlaciones entre las coordenadas

Para evaluar el grado de similaridad entre las coordenadas calculadas con diferentes valores de k debemos aplicar el modelo varias veces, con los valores 7 y 12 para k.

```
Coordenadas <- list(
  k7 = isoMDS(Datos, k = 7,
              trace = FALSE)$points,
  k12 = isoMDS(Datos, k = 12,
               trace = FALSE)$points
)

corr <- c(cor(Coordenadas$k7[, 1], Coordenadas$k12[, 1]),
          cor(Coordenadas$k7[, 2], Coordenadas$k12[, 2]))
names(corr) <- c("Primera dimensión", "SegundaDimensión")
corr #Correlaciones entre las dimensiones
```

```
## Primera dimensión  SegundaDimensión
##                 1                 1
```

La correlación entre la primera dimensión con $k = 7$ y la primera dimensión con $k = 12$ es perfecta (1). Lo mismo ocurre para la segunda dimensión.

Dado que la construccion de las dimensiones se hace de forma ortogonal, las correlaciones entre ellas son muy bajas. Por ejemplo, para $k = 7$ la matriz de correlaciones tiene el siguiente aspecto:

```
MatrizCorr <- round(cor(Coordenadas$k7), 4)
colnames(MatrizCorr) <- paste("Dim", 1:7, sep = "")
rownames(MatrizCorr) <- colnames(MatrizCorr)
kable(MatrizCorr,
      row.names = TRUE)
```

	Dim1	Dim2	Dim3	Dim4	Dim5	Dim6	Dim7
Dim1	1	0	0	0	0	0	0
Dim2	0	1	0	0	0	0	0
Dim3	0	0	1	0	0	0	0
Dim4	0	0	0	1	0	0	0
Dim5	0	0	0	0	1	0	0
Dim6	0	0	0	0	0	1	0
Dim7	0	0	0	0	0	0	1

Lo mismo ocurre para el caso de $k = 12$.

Reflexión

Queda patente que ésta técnica es capaz de organizar una serie de puntos de datos en el espacio dadas las distancias entre ellos. Por tanto, es importante que la metodología para calcular las distancias entre las ciudades sea consistente.

En el fichero que he consultado, las distancias son las distancias más cortas por carretera. Para reproducir fielmente la posición de las ciudades en el país sería más conveniente usar las distancias en línea recta.

No obstante, para una compañía de logística, por ejemplo, es preferible conocer las distancias que tendrán que recorrerse a través de las carreteras, de modo que el formato actual sería más conveniente.

Incluso se podría aumentar o disminuir la distancia en función de factores como las retenciones habituales de la zona, el consumo de combustible o el tiempo de viaje para obtener un plano deformado en función de los intereses de la empresa.

Si los vehículos de transporte contasen con equipos que registren éstos datos podría actualizarse de forma dinámica. Incluso podría crearse un plano personalizado para cada conductor o medio de transporte y compararlos para ver la forma más óptima de enviar una mercancía.

Si bien es cierto que al aumentar el número de ciudades aumentó el *stress* de forma considerable, sigue manteniendo un nivel aceptable.