Estadística Aplicada III

Métodos de agrupación: Conglomerados

Jorge de la Vega Góngora

Departamento de Estadística,
Instituto Tecnológico Autónomo de México

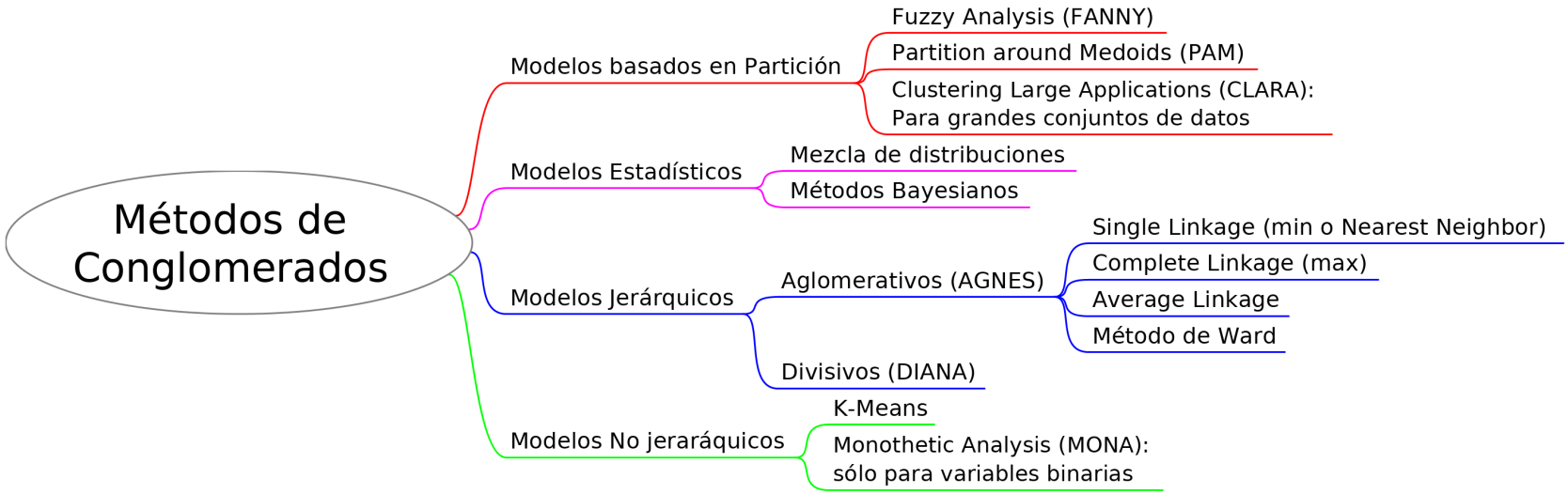
Semana 13: 13/16 de noviembre de 2018

Conglomerados (Clusters)

Introducción I

- El análisis de conglomerados (clusters) tiene por objeto *descubrir formas de agrupar* elementos de una matriz de datos $\mathbf{X}_{n\times p}$ en grupos homogéneos (llamados conglomerados o clusters) en función de las similitudes entre ellos.
- El análisis de conglomerados usualmente se aplica a los casos, o items de $\mathbf{X}$, pero también puede aplicarse a la agrupación de variables (las columnas de $\mathbf{X}$).
- Son métodos de aprendizaje estadístico no supervisado, pues no se conoce de antemano el número de grupos que se deben formar. Para determinar el número de grupos examina *algunas* de las posibles formas de agrupar los items.
- Hay muchos algoritmos de conglomerados que buscan agrupaciones maximales sin tener que buscar en todas las combinaciones posibles. Algunos de los algoritmos se clasifican como se indica en la siguiente gráfica, los cuales están programados en el paquete `cluster`, basados en el libro de Kaufman y Rousseeuw (1990).

Introducción II



- Los modelos son los siguientes:
  - **Métodos basados en partición**: (`kmeans`, `pam`, `clara`, `fanny`): producen conglomerados para una K fija. Necesitan un cluster incial, tienen muchos posibles criterios para optimizar, algunos basados en modelos probabilísticos. Pueden tener grupo(s) distinto(s) 'outliers'.

Introducción III

- **Métodos jerárquicos aglomerativos**: (`hclust`, `agnes`, `mclust`). Estos métodos producen un conjunto de conglomerados, usualmente uno con k clusters para cada $k = n, \ldots, 2$ grupos que se amalgaman sucesivamente. Las principales diferencias son en el cálculo de las (di)similitudes de grupo a grupo a partir de las (di)similitudes de item a item. Estos métodos son computacionalmente fáciles.
- **Métodos jerárquicos divisivos**: (`diana`, `mona`). Producen un conjunto de conglomerados, usualmente uno para cada $k = 2, \ldots, K \ll n$. Es casi imposible encontrar divisiones óptimas computacionalmente. Los métodos usualmente dividen una variable en cada etapa, son *monotéticos*.

Advertencia:
Los métodos de clustering NO son los mejores métodos para descubrir agrupaciones 'interesantes' de los datos. Los métodos de visualización son más efectivos. Los diferentes métodos de conglomerados pueden dar soluciones muy diferentes y esto puede llevar a una sobreinterpretación.

Métodos de conglomerados jerárquicos

Métodos de conglomerados jerárquicos I

- Los métodos jerárquicos aplican una serie de uniones o divisiones sucesivas.
  - Los métodos aglomerativos comienzan con los items individuales formando n clusters individuales, y se procede hasta que todos los items quedan en un sólo cluster.
  - Los métodos divisivos se mueven en dirección opuesta, de 1 cluster a n clusters.
- Los resultados de las divisiones o uniones sucesivas se grafican en un *dendograma* (*dendron*, árbol)
- A continuación revisaremos ejemplos de algoritmos aglomerativos.

Algoritmos aglomerativos I

Los algorimos aglomerativos se basan en algoritmos muy similares al siguiente. En este algoritmo, se usa distancia, pero se puede cambiar por similitud. También se requiere un método de *enlace* (linkage). Comenzando con N items o variables,

1. Se consideran N clusters con un sólo item, y una respectiva matriz de distancias $\mathbf{D} = \{d_{ij}\}$
2. Buscar en la matriz $\mathbf{D}$ los pares más cercanos (más similares) de clusters, U y V con distancia d_{UV}
3. Se unen los clusters U y V y se etiqueta como un nuevo cluster (UV), y se actualiza la matriz para eliminar los renglones y columnas de distancias U y V y se agregan las correspondientes de (UV) a todos los otros clusters.
4. Repetir 1-3 $N - 1$ veces, ya que todos los items quedarán aglomerados en un sólo cluster. Registrar en cada iteración la identidad de los clusters que se unieron y los niveles de distancia en donde se hicieron las uniones.

- El criterio de enlace que que aplica en el paso 3 se puede definir de diferentes maneras:

Algoritmos aglomerativos II

- Como la distancia máxima entre pares de items en los diferentes clusters,

$$\text{máx}\{d_{uv}, u \in U, v \in V\}.$$

. Este tipo de enlace se conoce como **enlace completo**.

- Como la distancia mínima entre pares de items en los diferentes clusters,

$$\text{mín}\{d_{uv}, u \in U, v \in V\}.$$

Este tipo de enlace se conoce como **enlace sencillo**.

- Como la distancia promedio entre los items de los diferentes clusters,

$$\frac{\sum_{v \in V} \sum_{u \in U} d_{uv}}{|U||V|}.$$

Este tipo de enlace se conoce como **enlace promedio**.

Ejemplo conglomerados jerárquicos Aglomerativos I

Ejemplo. [Clustering con enlace sencillo]

- Supongamos que se tienen los 5 items: *a*, *b*, *c*, *d*, *e* con las siguientes distancias:

$$\mathbf{D} = \begin{bmatrix} a & b & c & d & e \\ 0 & & & & \\ 9 & 0 & & & \\ 3 & 7 & 0 & & \\ 6 & 5 & 9 & 0 & \\ 11 & 10 & 2 & 8 & 0 \end{bmatrix}$$

- En la primera iteración se tienen los clusters: $\{a\}$, $\{b\}$, $\{c\}$, $\{d\}$ y $\{e\}$. Los más cercanos son a y c, así que se crea el cluster $\{ce\}$ y se actualiza la matriz de distancias, considerando el mínimo de las distancias de los clusters $\{a\}$, $\{b\}$ y $\{d\}$ al cluster $\{ce\}$. Agrego al final de la matriz el nuevo cluster y elimino los renglones correspondientes a $\{c\}$ y $\{e\}$:

$$\mathbf{D} = \begin{bmatrix} a & b & d & ce \\ 0 & & & \\ 9 & 0 & & \\ 6 & 5 & 0 & \\ 3 & 7 & 8 & 0 \end{bmatrix}$$

Ejemplo conglomerados jerárquicos Aglomerativos II

Por ejemplo: $d(\{ce\}, a) = \text{mín}\{11, 3\} = 3$, $d(\{ce\}, b) = \text{mín}\{10, 7\} = 7$, $d(\{ce\}, d) = \text{mín}\{9, 8\} = 8$.

- En la segunda iteración se tienen los clusters: $\{a\}$, $\{b\}$, $\{d\}$ y $\{ce\}$. Los más cercanos ahora son $\{a\}$ y $\{ce\}$, así que se crea el cluster $\{ace\}$ y se actualiza la matriz de distancias, recalculando las distancias a $\{b\}$ y $\{d\}$ del cluster $\{ce\}$.

$$\mathbf{D} = \begin{bmatrix} b & d & ace \\ 0 & & \\ 5 & 0 & \\ 7 & 8 & 0 \end{bmatrix}$$

Por ejemplo: $d(\{ce\}, a) = \text{mín}\{11, 3\} = 3$, $d(\{ce\}, b) = \text{mín}\{10, 7\} = 7$, $d(\{ce\}, d) = \text{mín}\{9, 8\} = 8$.

- En la tercera iteración la distancia más corta es la de b a d, así que se crea el cluster $\{bd\}$ y se actualizan distancias a $\{ace\}$.

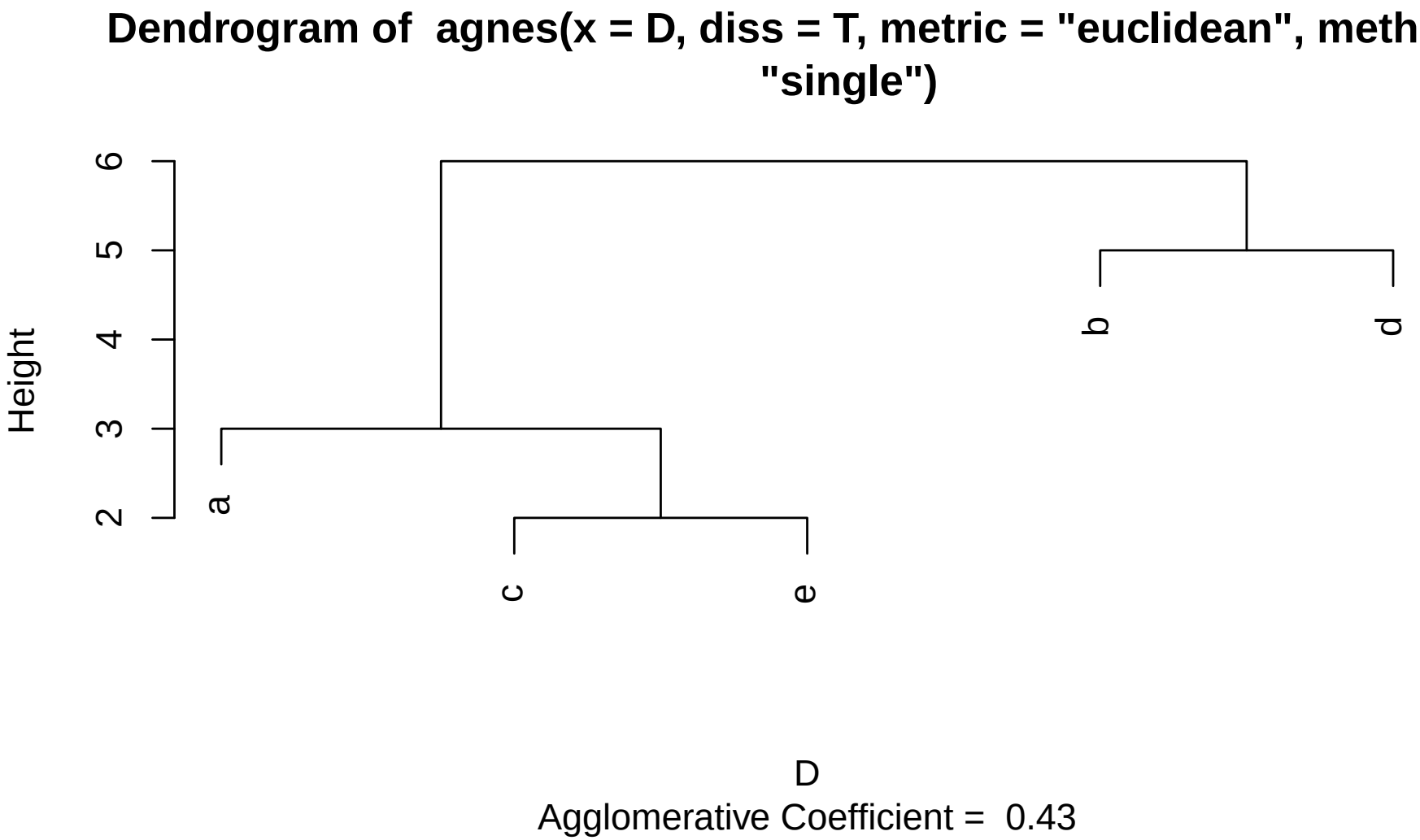
$$\mathbf{D} = \begin{bmatrix} bb & ace \\ 0 & \\ 6 & 0 \end{bmatrix}$$

Ejemplo conglomerados jerárquicos Aglomerativos III

En R se puede hacer el problema del siguiente modo (contempla los tres métodos de enlace que mencioné antes, más otros tres métodos más sofisticados):

```
library(cluster)

X <- c("a","b","c","d","e")
D <- as.data.frame(matrix(c(0,9,3,6,11,9,0,7,5,10,3,7,0,9,2,6,5,9,0,8,11,10,2,8,0),nrow=5),
row.names = X)
names(D) <- X
singlelink <- agnes(D, metric="euclidean", method = "single",diss=T)
singlelink


Call:  agnes(x = D, diss = T, metric = "euclidean", method = "single")
Agglomerative coefficient:  0.4333333
Order of objects:
[1] a c e b d
Height (summary):
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   2.00    2.75    4.00    4.00    5.25    6.00

Available components:
[1] "order"     "height"    "ac"        "merge"     "diss"      "call"      "method"    "order.lab"


plot(singlelink, which.plots = 2)
```

Ejemplo conglomerados jerárquicos Aglomerativos IV



Ejemplo conglomerados jerárquicos Aglomerativos V

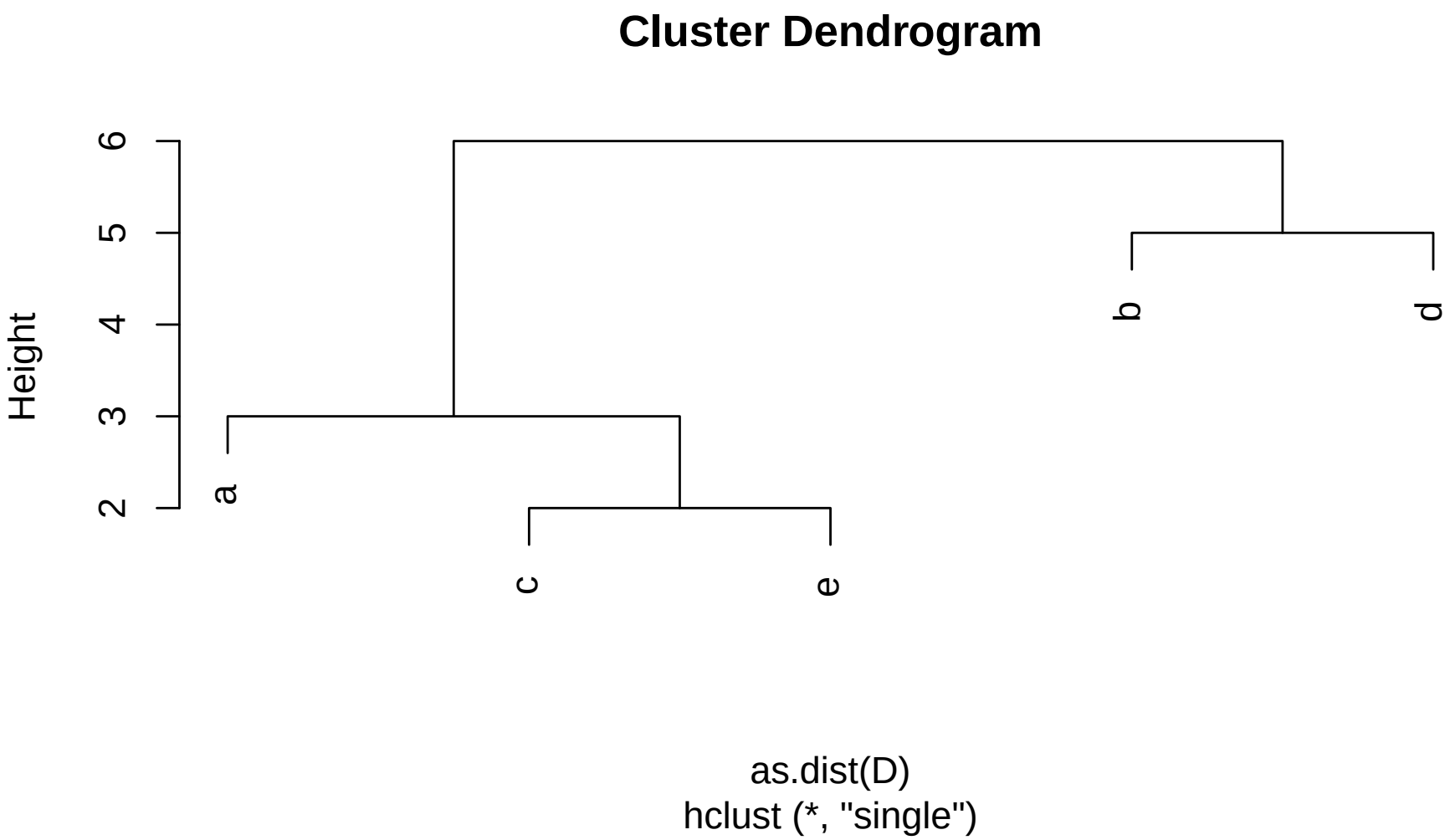
El *coeficiente aglomerativo* o *divisivo* mide la estructura de conglomerados que se obtuvo. Se calcula con la siguiente fórmula:

$$1 - \frac{\sum_{i=1}^{n} m(i)}{n}$$

donde $m(i)$ se define como la distancia al primer cluster con que se une la observación i, dividida por la distancia del cluster al que se une en el último paso del algoritmo. Alternativamente, se puede usar la función `hclust`:

```
plot(hclust(d = as.dist(D), method = "single"))
```

Ejemplo conglomerados jerárquicos Aglomerativos VI



□

Método de aglomeración jerárquica de Ward (1963) I

- Este método se basa en la menor pérdida de información al unir dos grupos, partiendo de que cada item es un cluster inicialmente. No se basa en la matriz de distancias sino en los items directamente.
- La pérdida de información se mide como el incremento en un criterio de error basado en una suma de cuadrados que se define para cada cluster K:

$$\mathrm{ESS}_K = \sum_{i \in K} (\mathbf{x}_i - \bar{\mathbf{x}}_K)(\mathbf{x}_i - \bar{\mathbf{x}}_K)'$$

donde $\bar{\mathbf{x}}_K$ es el centroide del cluster K. Entonces, para los N clusters,

$$\mathrm{ESS} = \sum_{i=1}^{N} \mathrm{ESS}_i = \sum_{i=1}^{N} \sum_{j \in i} (\mathbf{x}_j - \bar{\mathbf{x}}_i)'(\mathbf{x}_j - \bar{\mathbf{x}}_i)$$

- En este método, se consideran los $\binom{K}{2}$ pares de clusters y se combinan los que dan un incremento mínimo a ESS
- Inicialmente, se tiene que $\mathrm{ESS} = 0$ y al final se tendrá $\mathrm{ESS} = \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})'(\mathbf{x}_i - \bar{\mathbf{x}})$.

Método de aglomeración jerárquica de Ward (1963) II

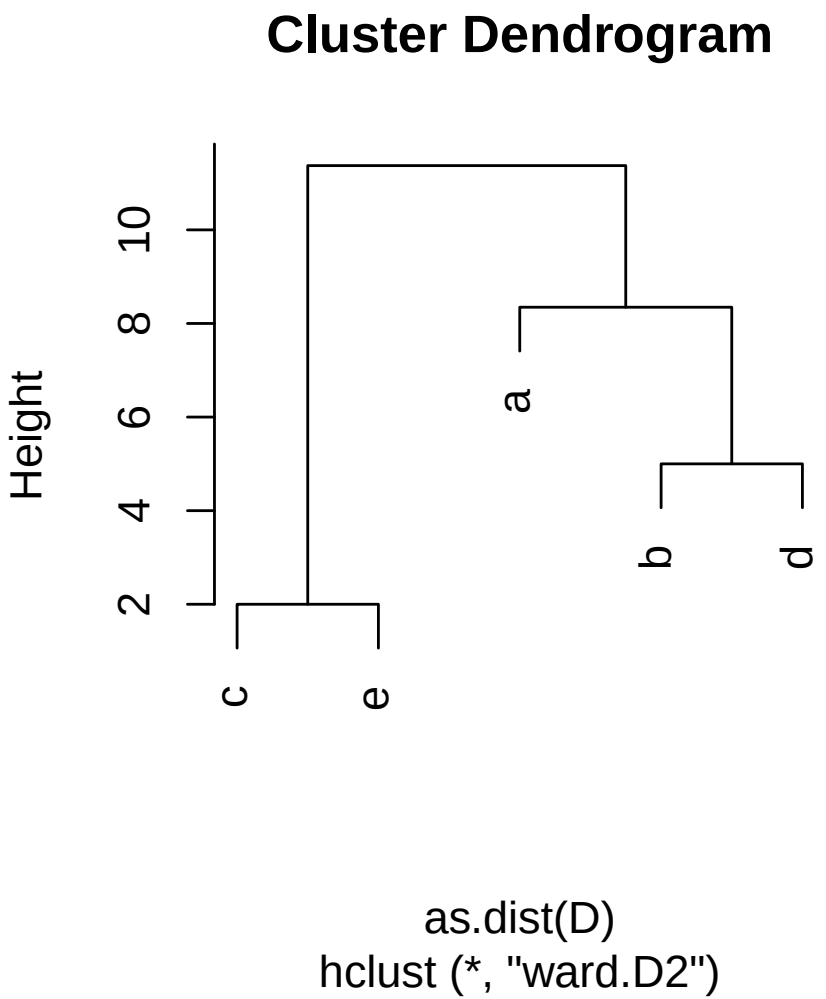
- Los valores de ESS_i pueden graficarse y usarse para decidir cuántos clusters 'naturales' se pueden considerar.

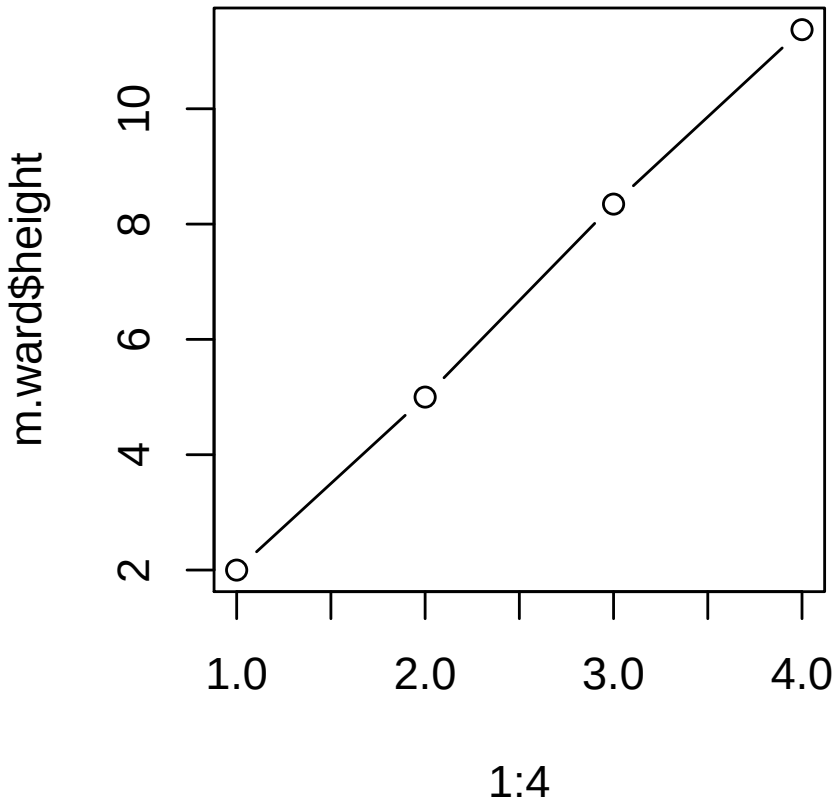
Ejemplo. [método de Ward]

Podemos realizar el ejercicio anterior con el método de Ward, para comparar resultados.

```
m.ward <- hclust(d = as.dist(D), method = "ward.D2") #ward.D2 es el método original de Ward (1963)
m.ward



Call:
hclust(d = as.dist(D), method = "ward.D2")

Cluster method   : ward.D2
Number of objects: 5


par(mfrow=c(1,2))
plot(m.ward)
plot(1:4,m.ward$height,type="b")
```

Método de aglomeración jerárquica de Ward (1963) III





□

Métodos de conglomerados no jerárquicos

Características generales de los métodos no jerárquicos I

- Los métodos de conglomerados no jerárquicos están diseñados para agrupar items, no variables, en K conglomerados.
- K se puede determinar de antemano o como parte del procedimiento. Sin embargo, no se recomienda fijar de antemano el número de conglomerados:
  - Puede haber clusters no bien diferenciados cuando se eligen valores iniciales dentro de uno de los clusters 'naturales'.
  - La existencia de valores extremos puede afectar la configuración final, creando clusters artificiales.
  - Se pueden obtener clusters que no tienen sentido.
- Sin embargo, la función `kmeans` fija de antemano el número de clusters.
- No requiere especificar la matriz de similitudes o distancias.
- Usualmente puede trabajar con conjuntos de datos mucho más grandes que en los casos de los modelos jerárquicos, debido a que no requiere almacenar simultáneamente todos los datos.
- Se requiere definir una partición inicial de los items en grupos, o bien de un conjunto inicial de puntos semillas que formarán los núcleos de los clusters. La solución final dependerá de las condiciones iniciales.

K-medias I

El algoritmo general se basa en asignar cada item al cluster que tenga el centroide o media más cercano.

1. Se seleccionan K puntos como centros de los grupos iniciales, que se puede hacer:
   - asignando aleatoriamente los items a los grupos y tomando los centros de los grupos formados,
   - tomando como centros los K puntos más alejados entre sí, o
   - contruyendo grupos iniciales con información *a priori*.
2. Se calculan las distancias euclídeas de cada item a los centros de los K grupos, y se asigna ese item al más próximo, de manera secuencial. Con cada asignación, se recalcula el centro del grupo al que el item fue asignado y del grupo que perdió el item.
3. A partir de algún criterio de optimalidad, se verifica si con la reasignación de los items mejora el criterio. Si no es posible mejorar el criterio, terminar el proceso.

Hay varias versiones de este algoritmo que se encuentran programados en la función `kmeans`: Hartigan & Wong (1979), MacQueen (1967), Lloyd (1957) (1982), (basado en regiones de Voronoi), y Forgy (1965).

Ejemplo I

Consideraremos los datos `iris` para comparar con los métodos de clasificación que vimos antes.

```
data("iris")
iris2 <- iris
iris2$Species <- NULL
mod1 <- kmeans(iris2,3)
mod1


K-means clustering with 3 clusters of sizes 50, 62, 38

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     5.006000    3.428000     1.462000    0.246000
2     5.901613    2.748387     4.393548    1.433871
3     6.850000    3.073684     5.742105    2.071053

Clustering vector:
  [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 [48] 1 1 1 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2
 [95] 2 2 2 2 2 2 3 2 3 3 3 3 2 3 3 3 3 3 2 2 3 3 3 2 3 2 3 2 3 3 2 2 3 3 3 3 2 3 3 3 2 3 3
[142] 3 2 3 3 3 2 3 3 2

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
 (between_SS / total_SS =  88.4 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
```

Podemos comparar el resultado de clasificación con los datos originales

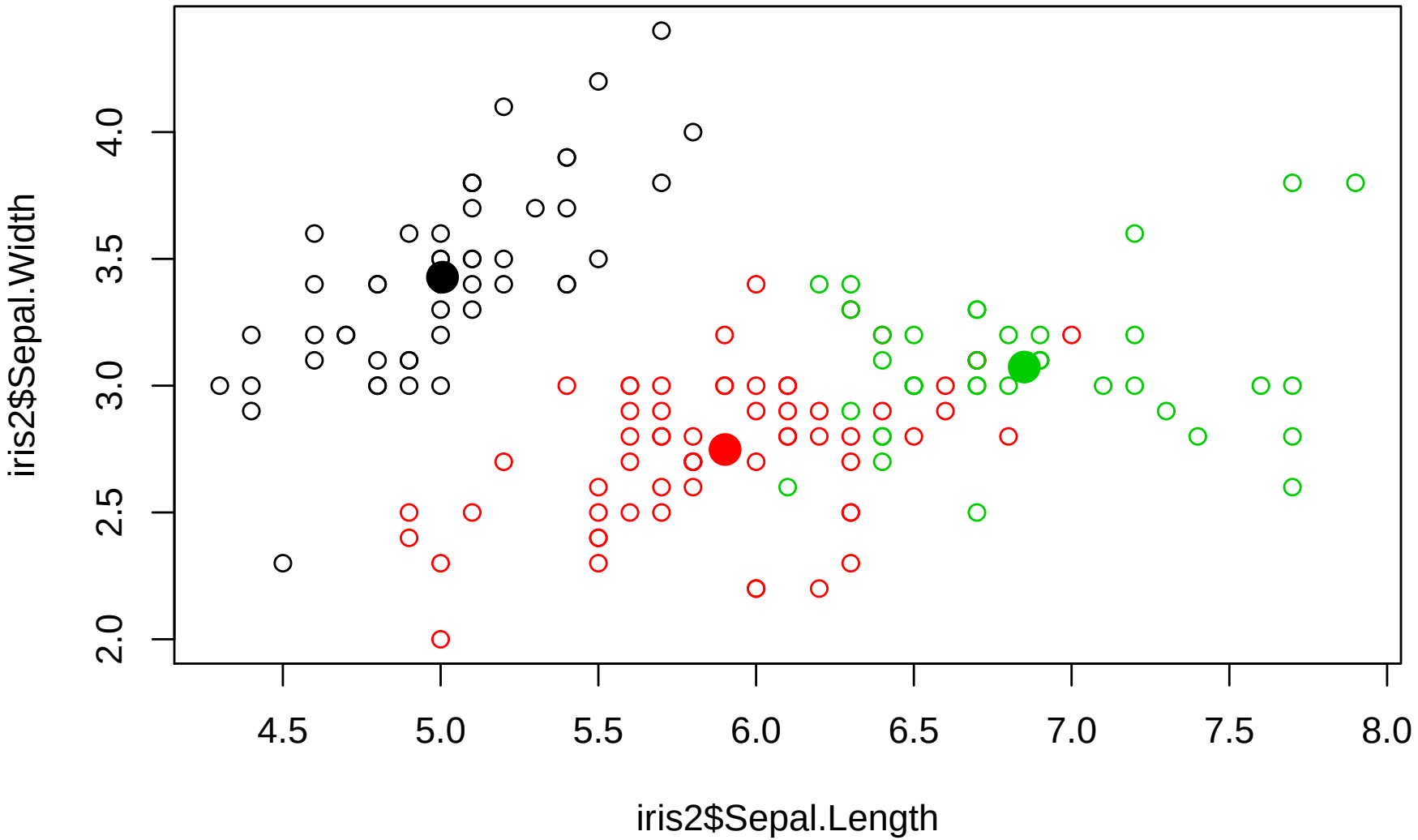
Ejemplo II

```
table(iris$Species,mod1$cluster)
```

```
            1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36
```

Ejemplo

```
plot(iris2$Sepal.Length,iris2$Sepal.Width,col=mod1$cluster)
points(mod1$centers[,c("Sepal.Length","Sepal.Width")], col=1:3, pch=16,cex=2)
```



Análisis monotético (MONA) I

- Especialmente para vectores binarios, correspondientes a indicadoras de atributos. Por ejemplo, los datos en `animals`:

```
data(animals)
head(animals)
```

```
    war fly ver end gro hai
ant   1   1   1   1   2   1
bee   1   2   1   1   2   2
cat   2   1   2   1   1   2
cpl   1   1   1   1   1   2
chi   2   1   2   2   2   2
cow   2   1   2   1   2   2
```

  En este conjunto de datos cada renglón corresponde a un animal y las columnas son los siguientes atributos:

  - `war` si es de sangre fría o caliente
  - `fly` si vuela o no vuela
  - `ver` si tiene ve.. perdón, si es vertebrado o invertebrado.
  - `end` si está en peligro o no
  - `gro` si vive en grupos o no
  - `hai` si tiene cabello o no.

- El procedimiento para este caso es el siguiente:
  1. Imputar la matriz si hay datos faltantes.

Análisis monotético (MONA) II

2. Dividir cada cluster de acuerdo a una variable X_J: un cluster es para todos los casos que tienen $X_{lJ} = 1$ y el otro cluster es para los que tienen $X_{mJ} = 0$. La variable usada para dividir un cluster es la variable con la mayor asociación a las otras variables.
   La asociación entre variables f y g está dada por $A_{fg} = |\det(Cont_{fg})|$ donde $Cont_{fg} = \begin{pmatrix} a_{fg} & b_{fg} \\ c_{fg} & d_{fg} \end{pmatrix}$ es la tabla de contingencia de las variables f y g y a_{fg} es la frecuencia de unos en ambas variables condicional al cluster que se va a dividir. La asociación total de una variable f es $A_f = \sum_{g \neq f} A_{fg}$. La variable h que satisface $A_h = \max_f A_f$ se selecciona para dividir el cluster.
3. Repetir el paso 2 hasta que cada cluster consista de objetos que tengan idénticos valores para todas las variables

Ejemplo. [Ejemplo de MONA para los datos de animales]

Análisis monotético (MONA) III

```
animales <- animals
animales <- animales -1
animales[is.na(animales)] <- 0 #imputación (se puede mejorar)
head(animales)
```

```
    war fly ver end gro hai
ant   0   0   0   0   1   0
bee   0   1   0   0   1   1
cat   1   0   1   0   0   1
cpl   0   0   0   0   0   1
chi   1   0   1   1   1   1
cow   1   0   1   0   1   1
```

```
mona1 <- mona(animales)
mona1
```

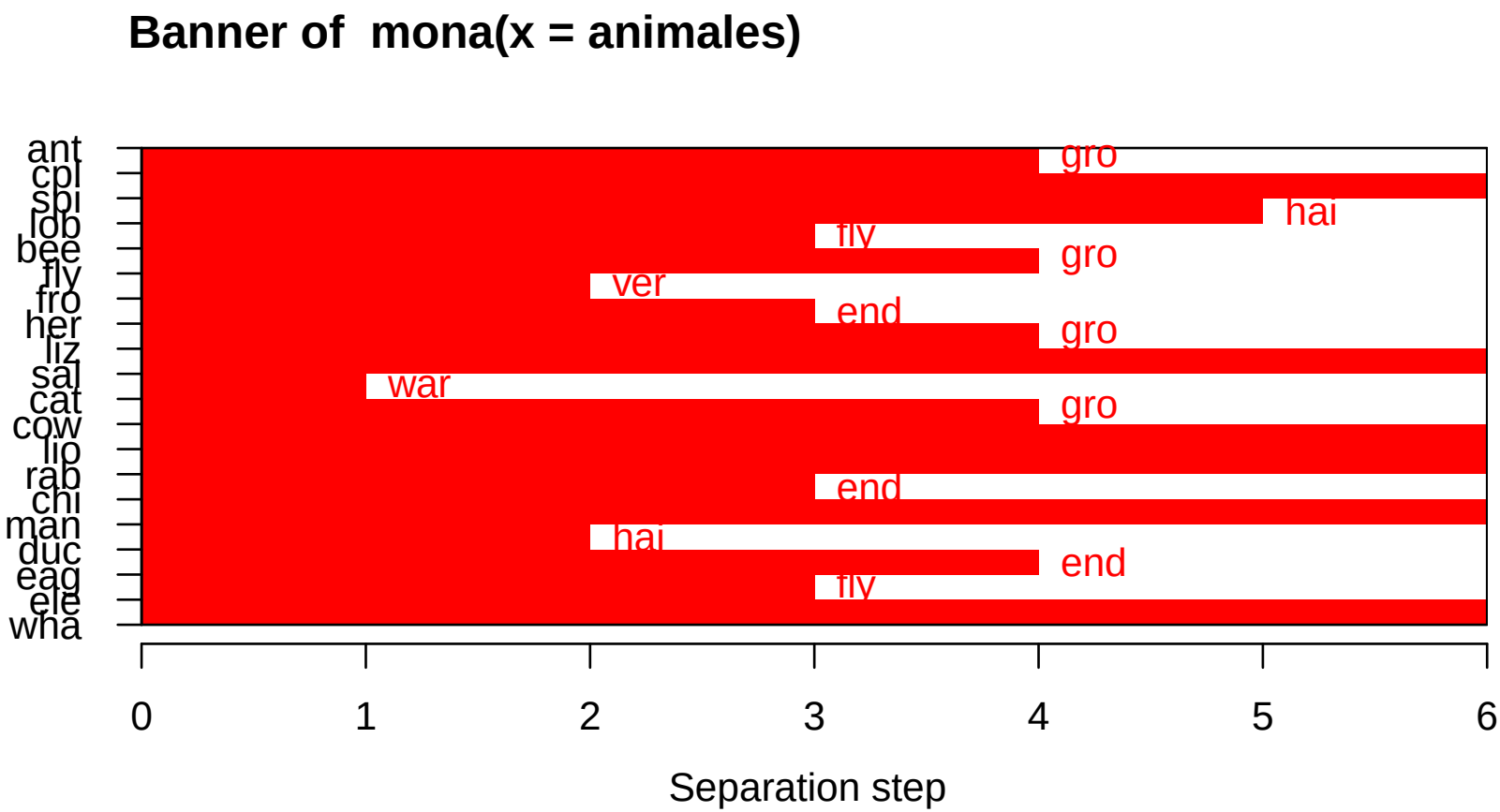
```
mona(x, ..) fit;  x of dimension 20x6
Order of objects:
 [1] ant cpl spi lob bee fly fro her liz sal cat cow lio rab chi man duc eag ele wha
Variable used:
 [1] gro  NULL hai  fly  gro  ver  end  gro  NULL war  gro  NULL NULL end  NULL hai  end  fly  NULL
Separation step:
 [1] 4 0 5 3 4 2 3 4 0 1 4 0 0 3 0 2 4 3 0

Available components:
[1] "data"      "hasNA"     "order"     "variable"  "step"      "order.lab" "call"
```

Gráfica de Banner I

- Una gráfica de Banner muestra las divisiones sucesivas de izquierda a derecha. La longitud de una barra es el número de pasos divisivos necesarios para hacer una división. Junto a cada barra, se muestra cuál es la variable que causa la división. Una barra que continúa en el margen derecho, indica un cluster que no se puede dividir.

```
plot(mona1)
```

Gráfica de Banner II



Determinación óptima del número de clusters I

- Como se ha mencionado, sería ideal no determinar de antemano el número de clusters. Se sugiere hacer una evaluación adecuada para determinar cuál debería ser el número óptimo de clusters a considerar. Hay tres métodos que son comunes:
  - Método del codo
  - Método de la silueta
  - estadística gap
- Revisaremos aquí sólo los dos primeros casos. El tercero se puede consultar en el artículo de Tibishirani, Walther, Hastie (2001).

Método de codo I

Método de codo

1. Se evalúa el algoritmo de conglomerados (e.g. k-medias) para diferentes valores de k.
2. Para cada k se calcula $ESS(k)$[a].
3. Graficar $ESS(k)$ vs k
4. Si se forma un codo, es donde se especifica el número de clusters óptimo.

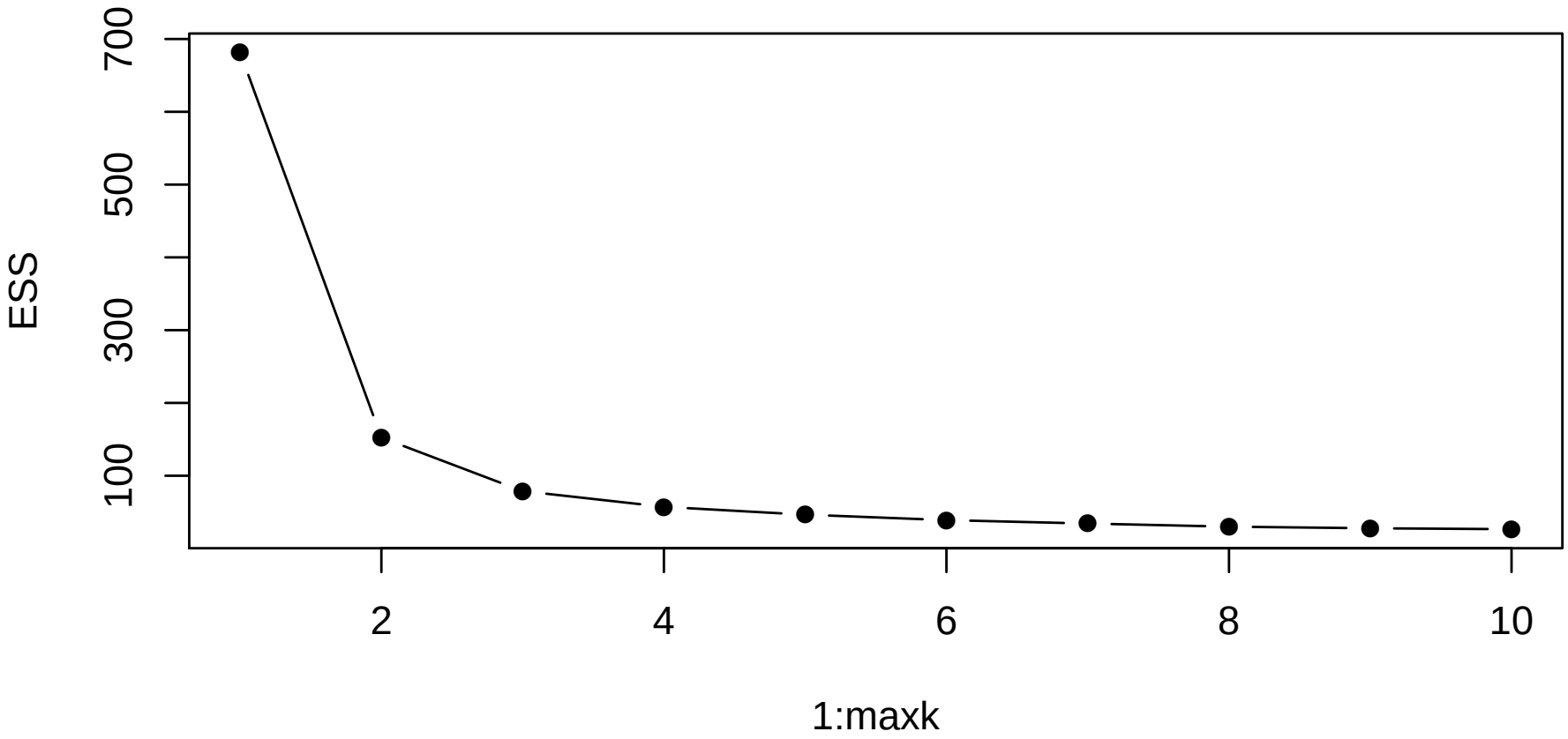
[a] Notar que $ESS(k) = ESS$ no se refiere a los componentes de ESS, que denotamos con ESS_k.

Ejemplo. [Cálculo para `iris`]

A continuación evaluamos el procedimiento indicado

```
maxk <- 10 #número de clusters a considerar
ESS <- sapply(1:maxk,function(k)kmeans(iris2,k,nstart=10)$tot.withinss)
plot(1:maxk,ESS, type="b",pch=16)
```

Método de codo II



□

Método de la silueta I

- A veces el método del codo es un poco ambiguo en la determinación del número óptimo de clusters. Por eso es conveniente tener más de una opción.
- El *análisis de silueta* mide qué tan bien una observación es conglomerada y estima la distancia promedio entre conglomerados. La gráfica de la silueta muestra una medida de que tan cerca cada punto en un cluster está de puntos en los clusters vecinos.

Algoritmo para silueta

Para cada observación i, el ancho de la silueta s_i se calcula de la siguiente manera:

1. Se calcula la disimilaridad promedio a_i entre i y todos los puntos del cluster al que pertenece el item i.
2. Para todos los otros clusters C a los que no pertenece i, se calcula la disimilaridad promedio $d(i, C)$ de i a todos los items de C. Se define $b_i = \text{mín}_C\, d(i, C)$, y se interpreta como la disimilaridad entre i y el cluster más cercano al que no pertenece i.
3. El ancho de la silueta del item i se define como $s_i = \frac{b_i - a_i}{\text{máx}\{a_i, b_i\}}$

Método de la silueta II

Podemos interpretar el ancho de la silueta de la siguiente manera:

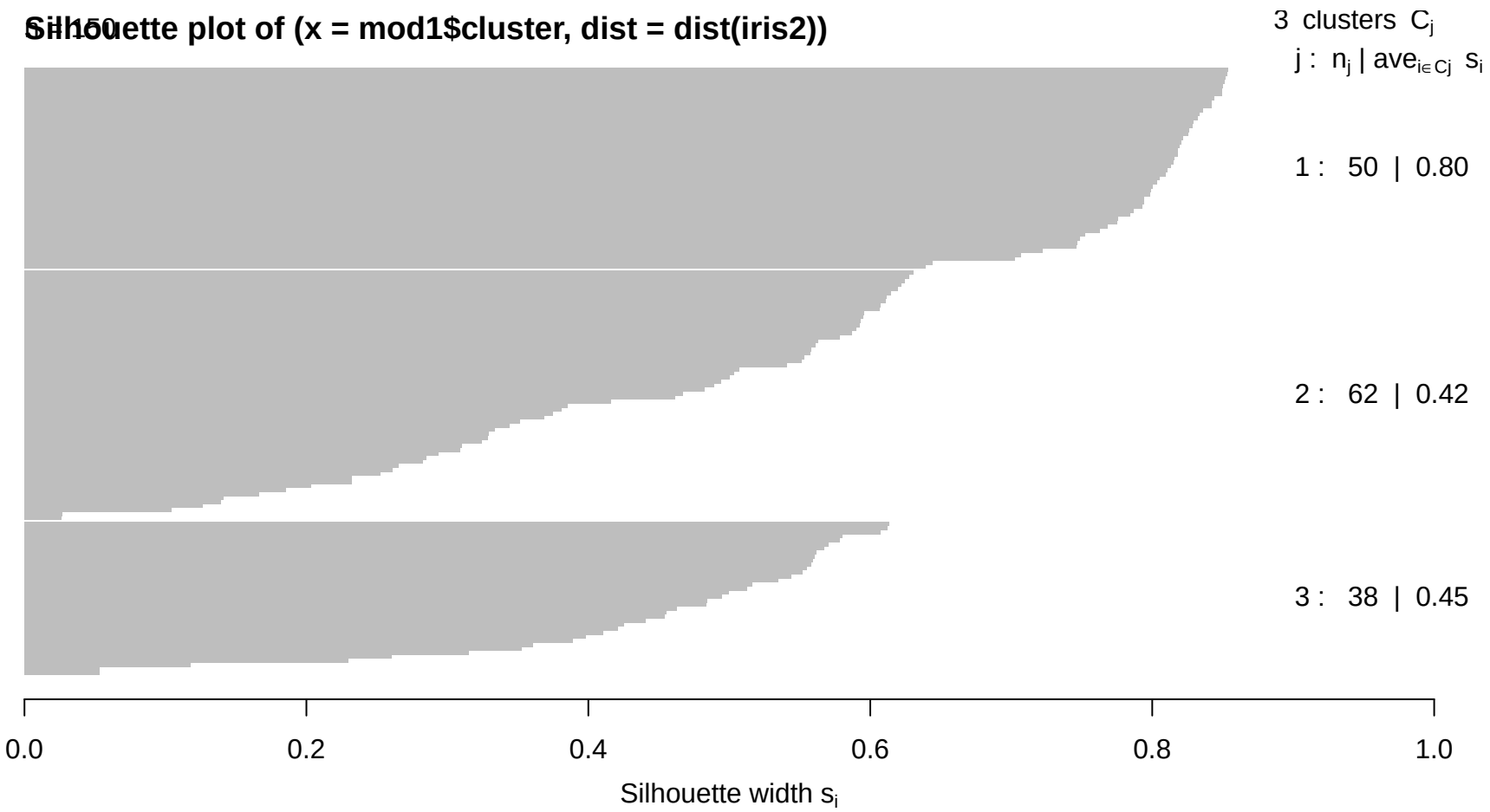
- Los items con un valor grande de s_i (casi 1) están bien agrupados
- Los items con un valor pequeño de s_i (casi 0), significa que los items quedan entre dos clusters.
- Items con valores negativos de s_i están mal agrupados.

- En el paquete `cluster` se puede usar la función `silhouette`.
- En el paquete `factoextra` la función `fviz_nbclust` genera las gráficas de codo y de silueta.

Ejemplo. [silueta para `iris`]

```
plot(silhouette(mod1$cluster,dist(iris2)))
```

Método de la silueta III



Métodos estadísticos

Mezcla de distribuciones I

- El supuesto principal es que los vectores $\mathbf{x}_1, \ldots, \mathbf{x}_n$ se obtienen de una mezcla de distribuciones:

$$f(\mathbf{x}) = \sum_{k=1}^{K} \pi_k f_k(\mathbf{x})$$

  donde $\pi_k \geqslant 0, \quad \sum_{k=1}^{K} \pi_k = 1$

- El caso más estudiado es el caso de mezcla de normales, pero en teoría la mezcla puede ser arbitraria. En el caso normal, los conglomerados tienen formas elípticas.
- Bajo el supuesto de que los datos son generados por una mezcla, se estiman conjuntamente los parámetros que forman la mezcla y las probabilidades posteriores de cada dato de pertenecer a cada uno de los componentes de la mezcla.

Mezcla de distribuciones II

- La función de log-verosimilitud es:

$$l(\theta|\mathbf{X}) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i) \right)$$

- Suponiendo que las distribuciones de la mezcla son normales $\mathcal{N}_p(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$, el vector de parámetros es de la forma:

$$\theta = (\pi_1, \ldots, \pi_K, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_K)$$

y la log-verosimilitud es

$$l(\theta|\mathbf{X}) = \sum_{i=1}^{n} \log \left(\sum_{k=1}^{K} \pi_k |\boldsymbol{\Sigma}_k|^{-1/2} (2\pi)^{-p/2} \exp \left\{ -\frac{1}{2} (\mathbf{x}_i - \boldsymbol{\mu}_k)' \boldsymbol{\Sigma}_k^{-1} (\mathbf{x}_i - \boldsymbol{\mu}_k) \right\} \right)$$

La fórmula anterior no se puede maximizar si hay una observación por conglomerado, por lo que hay que suponer que en cada cluster hay por lo menos l observaciones.

Optimización de máxima verosimilitud I

- Para maximizar la función de log-verosimilitud, se incorpora la restricción de que $\sum_{k=1}^{K} \pi_k = 1$ y se aplican multiplicadores de Lagrange para obtener:

$$\frac{\partial l(\theta|\mathbf{X})}{\partial \pi_g} = \sum_{i=1}^{n} \frac{f_g(\mathbf{x}_i)}{\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i)} - \lambda = 0$$

multiplicando ambos lados por π_g:

$$\sum_{i=1}^{n} \frac{f_g(\mathbf{x}_i)\pi_g}{\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i)} = \sum_{i=1}^{n} \pi_{ig} = \lambda \pi_g$$

y sumando sobre todos los grupos:

$$\begin{aligned} \sum_{g=1}^{K} \lambda \pi_g &= \sum_{g=1}^{K} \sum_{i=1}^{n} \frac{f_g(\mathbf{x}_i)\pi_g}{\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i)} \\ \lambda &= \sum_{i=1}^{n} \frac{\sum_{g=1}^{K} f_g(\mathbf{x}_i)\pi_g}{\sum_{k=1}^{K} \pi_k f_k(\mathbf{x}_i)} \\ &= n \end{aligned}$$

Optimización de máxima verosimilitud II

Entonces:

$$\hat{\pi}_g = \frac{1}{n}\sum_{i=1}^{n}\hat{\pi}_{ig}$$

- Derivando ahora respecto a $\boldsymbol{\mu}_g$ y $\boldsymbol{\Sigma}_g$ se llega a los estimadores ponderados, donde cada peso es la probabilidad relativa de que el item i pertenezca a la población g:

$$\hat{\boldsymbol{\mu}}_g = \sum_{i=1}^{n}\omega_{ig}\mathbf{x}_i, \qquad \hat{\boldsymbol{\Sigma}}_g = \sum_{i=1}^{n}\omega_{ig}(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)(\mathbf{x}_i - \hat{\boldsymbol{\mu}}_g)'$$

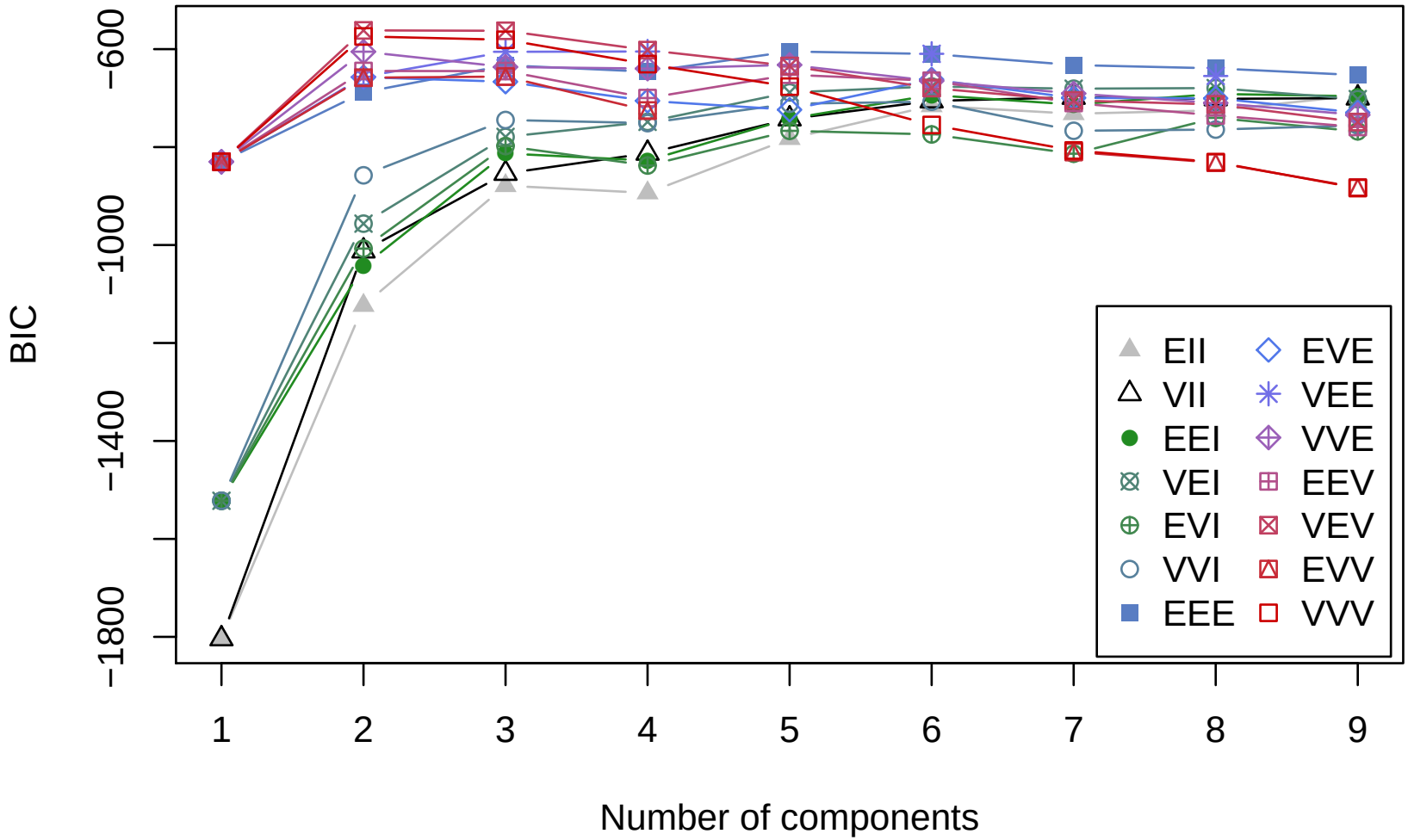
  donde $\omega_{ig} = \frac{\pi_{ig}}{\sum_{i=1}^{n}\pi_{ig}}$.
- El sistema de ecuaciones encontrados tiene que resolverse iterativamente a través del algoritmo EM (*Expectation-Maximization*). El paquete `mclust` resuelve el numéricamente el problema.
- Para determinar el número de clusters adecuado se utiliza el criterio AIC o BIC.

Ejemplo I

- Con los datos `iris`:

```
library(mclust)

Package 'mclust' version 5.4.1
Type 'citation("mclust")' for citing this R package in publications.

# Primero calcula el número de grupos a considerar
BIC <- mclustBIC(iris2)
plot(BIC)
```

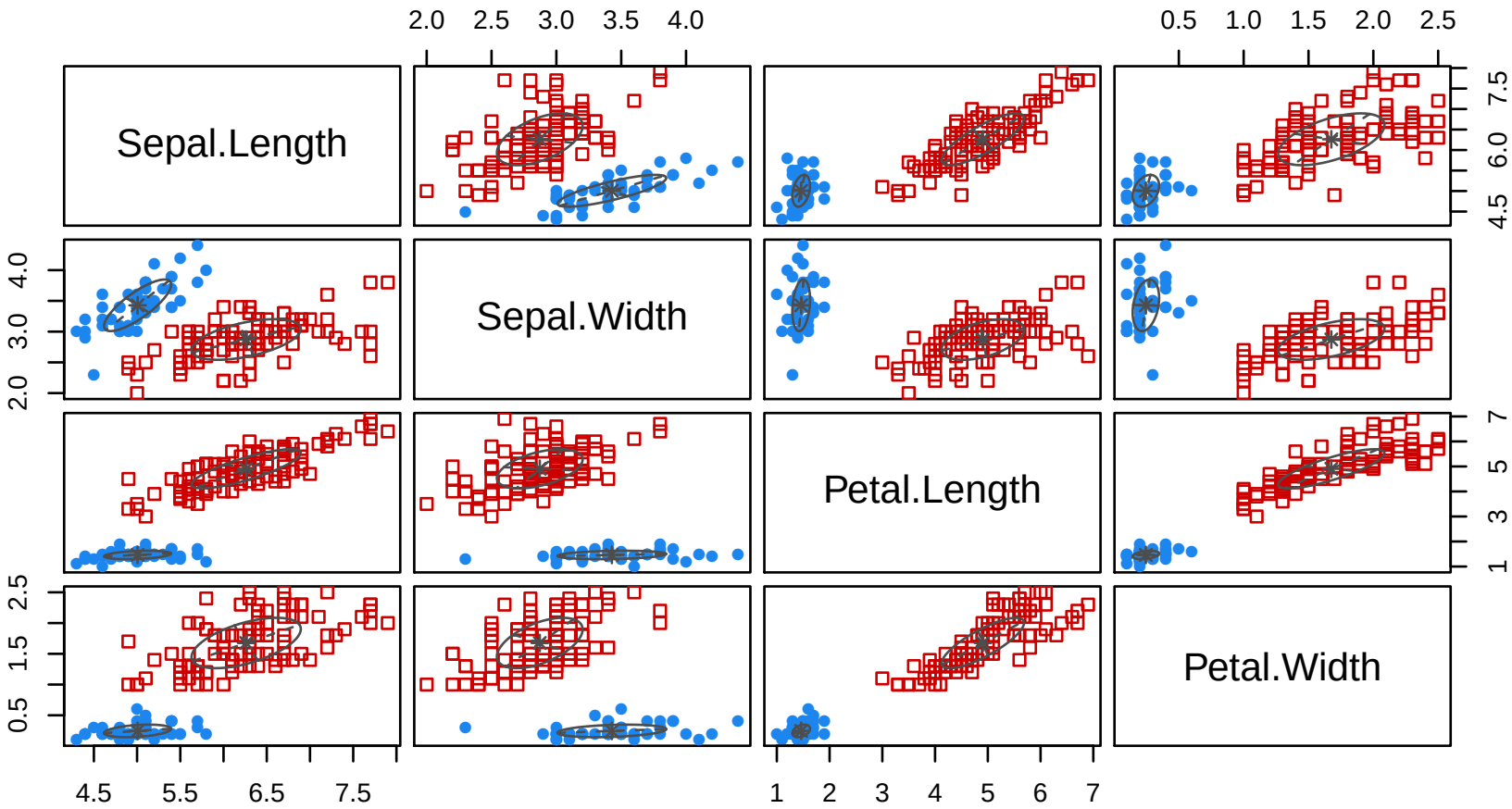
Ejemplo II


BIC
−600
−1000
−1400
−1800
1
2
3
4
5
6
7
8
9
Number of components
EII
VII
EEI
VEI
EVI
VVI
EEE
EVE
VEE
VVE
EEV
VEV
EVV
VVV

Ejemplo III

```
summary(BIC)

Best BIC values:
             VEV,2        VEV,3      VVV,2
BIC      -561.7285 -562.5522369 -574.01783
BIC diff    0.0000   -0.8237748  -12.28937

mod1 <- Mclust(iris2,x=BIC)
mclustModelNames("VEV")

$model
[1] "VEV"

$type
[1] "ellipsoidal, equal shape"

summary(mod1)

----------------------------------------------------
Gaussian finite mixture model fitted by EM algorithm
----------------------------------------------------

Mclust VEV (ellipsoidal, equal shape) model with 2 components:

 log.likelihood   n df       BIC       ICL
       -215.726 150 26 -561.7285 -561.7289

Clustering table:
  1   2
 50 100

plot(mod1,what="classification")
```

Ejemplo IV


Sepal.Length
Sepal.Width
Petal.Length
Petal.Width
2.0 2.5 3.0 3.5 4.0
0.5 1.0 1.5 2.0 2.5
4.5 6.0 7.5
2.0 3.0 4.0
1 3 5 7
0.5 1.5 2.5
4.5 5.5 6.5 7.5
1 2 3 4 5 6 7

Ejemplo V

```
#Calcula la tabla de clasificación:
table(iris$Species,mod1$classification)
```

```
            1  2
  setosa     50  0
  versicolor  0 50
  virginica   0 50
```

Métodos Bayesianos I

- El problema de estimación en mezclas de distribuciones puede ser complejo: la verosimilitud tiene K^n términos correspondientes a la asignación de las n observaciones de la muestra a las K poblaciones.
- Una forma alternativa de poder resolver el problema es plantearlo como un modelo Bayesiano que puede resolverse via Gibbs sampler, como una alternativa al algoritmo EM.
- En el enfoque Bayesiano, el vector de parámetros $\theta = (\pi_1, \ldots, \pi_K, \mu_1, \ldots, \mu_K, \Sigma_1, \ldots, \Sigma_K)$ es aleatorio.
- Para cada observación i, sea z_i la variable indicadora de la población a su grupo de pertenencia. Entonces z_i tiene dimensión K y tiene componentes 0 y un uno en el lugar del grupo correspondiente. Entonces podemos escribir la distribución de cada observación condicional en $z_i = (z_{i1}, z_{i2}, \ldots, z_{iK})$:

$$x_i | z_i \theta \sim \mathcal{N}_p \left(\prod_{i=1}^{K} \mu_i^{z_{ig}}, \prod_{i=1}^{K} \Sigma_i^{z_{ig}} \right)$$

Métodos Bayesianos II

- Por otra parte, $\mathbf{z}_i|\theta$ es un vector que tiene distribución multinomial con vector de probabilidades $\boldsymbol{\pi}$.

$$\mathbf{z}|\theta \sim \mathcal{M}\left(1, K, \boldsymbol{\pi}\right)$$

- Entonces, la verosimilitud de la muestra está dada por:

$$L(\theta|\mathbf{X}, \mathbf{Z}) = f(\mathbf{X}, \mathbf{Z}|\theta) = \prod_{i=1}^{n} f(\mathbf{x}_i|\mathbf{z}_i\theta) f(\mathbf{z}_i|\theta)$$

que tiene $2n$ términos, n normales y n coeficientes π_{ig} determinados por las $\mathbf{z}_i$.

- Por conveniencia computacional, y por conveniencia asintótica de la posterior, se toma una familia conjugada a la distribución multinomial, que es la distribución de Dirichlet sobre el vector de probabilidades $\boldsymbol{\pi}$, una normal para la media dada la varianza, y una distribución Wishart para la precisión:

$$\begin{aligned} \boldsymbol{\pi} &\sim \text{Dirichlet}(\boldsymbol{\alpha}) \\ \boldsymbol{\mu}_i|\boldsymbol{\Sigma}_i^{-1} &\sim \mathcal{N}_p\left(\boldsymbol{\mu}_{i0}, \boldsymbol{\Sigma}_i/n_{i0}\right) \\ \boldsymbol{\Sigma}_i^{-1} &\sim \mathcal{W}\left(p, m_{i0}, \mathbf{M}_i/m_{i0}\right) \end{aligned}$$

Métodos Bayesianos III

- La distribución posterior, la podemos expresar entonces de la siguiente manera:

$$\pi(\boldsymbol{\theta}, \mathbf{Z}|\mathbf{X}) \propto \mathrm{f}(\mathbf{X}|\boldsymbol{\theta}, \mathbf{Z})\mathrm{f}(\mathbf{Z}|\boldsymbol{\theta})\pi(\boldsymbol{\theta}) = \mathrm{L}(\boldsymbol{\theta}|\mathbf{X}, \mathbf{Z})\pi(\boldsymbol{\theta})$$

  Entonces, si queremos obtener una muestra de la distribución posterior, $\pi(\boldsymbol{\theta}, \mathbf{Z}|\mathbf{X})$, podemos iterar el Gibbs Sampler para obtener una muestra de la condicional $\boldsymbol{\theta}|\mathbf{X}, \mathbf{Z}$ y de la condicional $\mathbf{Z}|\mathbf{Z}, \boldsymbol{\theta}$.

  - La primera condicional suponemos una asignación de las observaciones entre los grupos, y estimamos los parámetros.
  - En la segunda condicional, suponemos un valor para los parámetros y calculamos la probabilidad de que cada observación venga de cada grupo.

Muestreo de $\theta|\mathbf{X}, \mathbf{Z}$ I

- Supongamos que $\mathbf{Z} = \mathbf{Z}^{(h)}$ es dado. Entonces, obtenemos un valor para los parámetros de la siguiente manera:
  1. Para cada media con distribución $\boldsymbol{\mu}_g|\boldsymbol{\Sigma}_g^{-1}, \mathbf{X}, \mathbf{Z}^{(h)} \sim \mathcal{N}_p\left(\boldsymbol{\mu}_{gp}, \boldsymbol{\Sigma}_{gp}\right)$, donde

$$\boldsymbol{\mu}_{gp} = \frac{n_{g0}\boldsymbol{\mu}_{g0} + n_g\bar{\mathbf{x}}_g}{n_{g0} + n_g}$$

y $n_g = \sum_{i=1}^{n} z_{ig}$ es el número de observaciones en el grupo g dado por las indicadoras observadas, y $\bar{\mathbf{x}}_g$ es la media de esas mismas observaciones. Del mismo modo, la varianza posterior es

$$\boldsymbol{\Sigma}_{gp} = \frac{\boldsymbol{\Sigma}_g}{n_{g0} + n_g}$$

  2. Para las matrices de precisión, con $\boldsymbol{\Sigma}_g^{-1} \sim \mathcal{W}\left(p, n_{g0} + m_{g0}, \mathbf{M}_{gp}\right)$, donde

$$\mathbf{M}_{gp}^{-1} = m_{g0}\mathbf{M}^{-1} + n_g\mathbf{S}_g + \frac{n_g n_{g0}}{n_g + n_0}(\bar{\mathbf{x}}_g - \boldsymbol{\mu}_{g0})(\bar{\mathbf{x}}_g - \boldsymbol{\mu}_{g0})'$$

y la varianza muestral se estima con las observaciones de ese grupo por:

$$n_g\mathbf{S}_g = \sum_{i=1}^{n} z_{ig}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})'$$

Muestreo de $\theta|\mathbf{X}, \mathbf{Z}$ II

3. Para el vector de probabilidades, se tiene que

$$\pi_g|\mathbf{X}, \mathbf{Z}^{(h)} \sim \text{Dirichlet}(\alpha_1 + n_1, \ldots, \alpha_K + n_K)$$

donde cada n_i depende de la matriz $\mathbf{Z}^{(h)}$.

Muestreo de $\mathbf{Z}|\boldsymbol{\theta}, \mathbf{X}$ I

Una vez obtenido el vector de parámetros $\boldsymbol{\theta}^{(h)}$, obtenemos el nuevo valor de $\mathbf{Z} = \mathbf{Z}^{(h+1)}$ mediante simulación de una multinomial.

- Simular $\mathbf{Z} = (\mathbf{z}_1, \ldots, \mathbf{z}_n)$ de

$$\mathbf{z}_i|\mathbf{X}, \boldsymbol{\theta}^{(h)} \sim \mathcal{M}\left(1, K, \boldsymbol{\pi}(\boldsymbol{\theta}^{(h)})\right)$$

donde las probabilidades posteriores de las observaciones vienen dadas por:

$$\pi_{ih}(\boldsymbol{\theta}^{(h)}) = \frac{\pi_h f_h(\mathbf{x}_i|\boldsymbol{\theta}^{(h)})}{\sum_{l=1}^{K} \pi_l f_l(\mathbf{x}_i|\boldsymbol{\theta}^{(h)})}$$

estimación del modelo en R

- El paquete `bayesm` dedicado a estimación de funciones para clasificación en mercadotecnia, tiene la función `rnmixGibbs` que extrae muestras de mezclas de normales a través del Gibbs Sampler con los supuestos descritos arriba.

Name	Description	Model	Priors	Required arguments			Output
				Data	Prior	Mcmc	
`rbprobitGibbs`	Gibbs sampler for binary probit	$z = X\beta + \varepsilon, \varepsilon \sim N(0, 1)$ $y = 1$ if $z > 0$	$\beta \sim N(\bar{\beta}, A^{-1})$	X, y		R (no. of MCMC draws)	betadraw
`rhierBinLogit`	Hybrid sampler for hierarchical binary logit – designed for conjoint data. Note: x variables are expressed as a difference between alternatives. See Appendix A.	$y_{hi} = 1$ with $\text{pr} = \frac{\exp(x_{hi}'\beta_h)}{1+\exp(x_{hi}'\beta_h)}$ $h = 1, \ldots, H;$ $i = 1, \ldots, I_h$ H panel units $\beta_h \sim N(\Delta' z_h, V_\beta)$	$V_\beta \sim \text{IW}(\nu, V)$ $\text{vec}(\Delta) \sim N(\text{vec}(\bar{\Delta}), V_\beta \otimes A^{-1})$	lgtdata, lgtdata[[h]]$X is X matrix for unit h lgtdata[[h]]$y is dep. var. for unit h		R	betadraw Deltadraw Vbetadraw llike reject
`rnmixGibbs`	Gibbs sampler for mixture of normals	$y_i \sim \text{N}(\mu_{\text{ind}_i}, \Sigma_{\text{ind}_i})$ $\text{ind}_i \sim$ Multinomial(pvec) pvec is ncomp -vector	pvec $\sim$ Dirichlet(a) $\mu_j \sim \text{N}(\bar{\mu}, \Sigma_j \otimes a^{-1})$ $\Sigma_j \sim \text{IW}(\nu, V)$	y is $n \times k$ matrix of k-dim. obs	ncomp	R	probdraw zdraw (indicators) compdraw a list of lists of draws of normal comps
`rscaleUsage`	Gibbs sampler for multivariate ordinal data with scale usage heterogeneity. Designed for survey questions on k-point scale	$x_{ij} = d$ if $c_{d-1} < y_{ij} < c_d$ $d = 1, \ldots, k$ $c_d = a + bd + ed^2$ $y_i = \mu + \tau_i \iota + \sigma_i z_i$ $z_i \sim N(0, \Sigma)$	$\text{vec}(\tau_i, \log(\sigma_i)) \sim N(\phi, \Lambda)$ $\phi = \begin{pmatrix} 0 \\ \lambda_{22} \end{pmatrix}$ $\mu \sim N(\bar{\mu}, A_\mu^{-1})$ $\Sigma \sim \text{IW}(\nu, V), \Lambda \sim \text{IW}(\nu_\Lambda, V_\Lambda)$ $e \sim$ unif. on grid	k, x x is nobs $\times$ nvar matrix of responses to nvar survey questions, each one is on k-point scale			Sigmadraw mudraw taudraw sigmadraw Lambdadraw edraw use defaults!