Algoritmi gerarchici agglomerativi

In R sono disponibili diverse funzioni per implementare i metodi di clustering gerarchico:

- Approccio agglomerativo (*bottom-up*): `hclust()` (pacchetto **stats**), `agnes()` (pacchetto **cluster**);
- Approccio divisivo (*top-down*): `diana()` (pacchetto **cluster**)

I metodi gerarchici più utilizzati sono quelli di tipo agglomerativo, che aggregano insieme unità "simili" fino a ottenere un gruppo in cui sono presenti tutte le unità. Tale processo crea dei gruppi annidati riportati nel cosiddetto *dendrogramma*.

Le funzioni `hclust()` e `agnes()` sono abbastanza simili; tuttavia, con la funzione `agnes` è possibile ottenere anche il coefficiente di agglomerazione, che misura la quantità di struttura di clustering trovata (valori più vicini a 1 suggeriscono una forte struttura di clustering).

Per cominciare con un semplice esempio, considereremo il dataset R `eurodist`, contenente le distanze geografiche tra le città europee (digitare `?eurodist` per l'Help):

```
data(eurodist)

# few rows and columns
as.matrix(eurodist)[1:8, 1:8]
```

```
##            Athens Barcelona Brussels Calais Cherbourg Cologne Copenhagen Geneva
## Athens          0      3313     2963   3175      3339    2762       3276   2610
## Barcelona    3313         0     1318   1326      1294    1498       2218    803
## Brussels     2963      1318        0    204       583     206        966    677
## Calais       3175      1326      204      0       460     409       1136    747
## Cherbourg    3339      1294      583    460         0     785       1545    853
## Cologne      2762      1498      206    409       785       0        760   1662
## Copenhagen   3276      2218      966   1136      1545     760          0   1418
## Geneva       2610       803      677    747       853    1662       1418      0
```

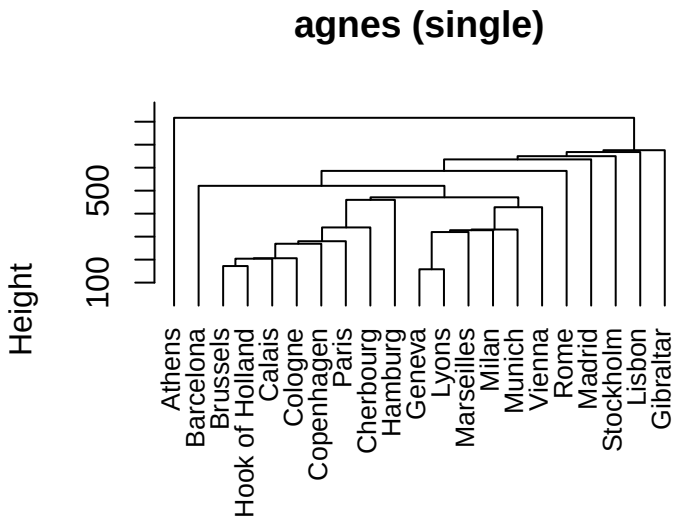
Eseguiamo il clustering gerarchico agglomerativo utilizzando `agnes()` (nel pacchetto **cluster**) sulle distanze **eurodist**, utilizzando `method=single` per il legame singolo (notare che passare un oggetto `dist` come primo argomento implica che tale oggetto sia assunto come matrice di dissimilarità):

```
library(cluster)
#single linkage
agnes.single<-agnes(eurodist, method="single")
# agglomerative coefficient
agnes.single$ac
```

```
## [1] 0.5115696
```

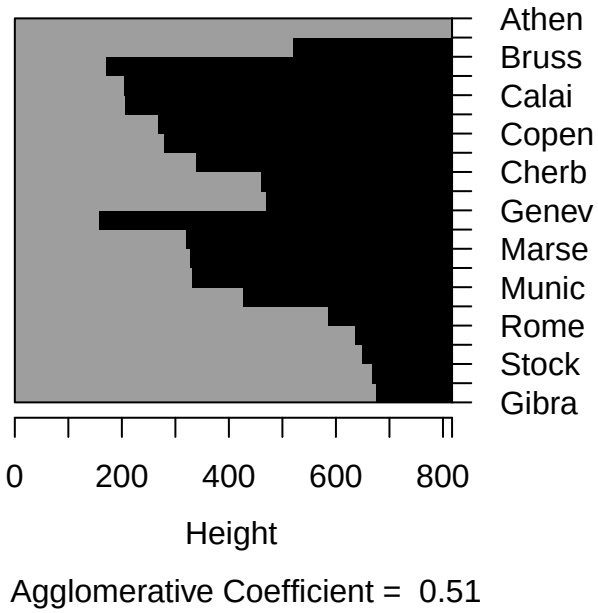
Otteniamo il dendrogramma:

```
# Put the labels at the same height: hang = -1
pltree(agnes.single, cex=0.8, hang = -1, main = "agnes (single)", xlab="", sub ="")
```



Attraverso il comando `plot()` possiamo anche ottenere una rappresentazione del livello di aggregazione delle diverse osservazioni, insieme al coefficiente di agglomerazione calcolato da `agnes`

```
plot(agnes.single, which=1, col=c(8,1), main="")
```



In alternativa, si può ottenere il dendrogramma nel modo seguente:

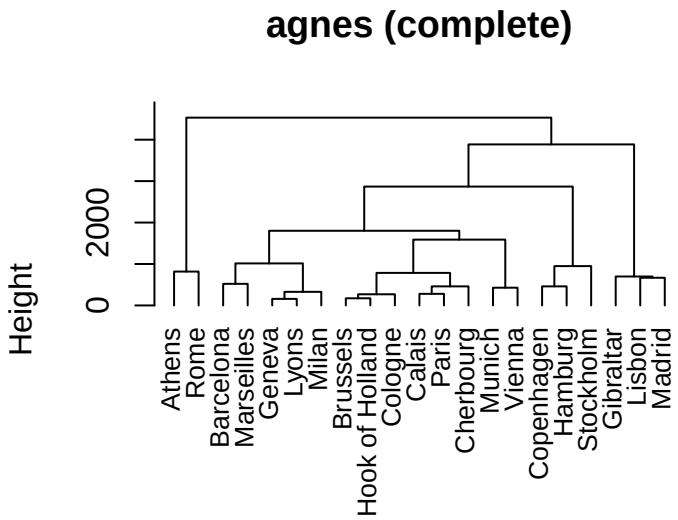
```
?hclust
hc<-hclust(eurodist, method="single")
plot(hc, hang=-1)
```

Il risultato del legame singolo mostra il consueto effetto a catena. Applichiamo allora gli altri metodi di calcolo delle distanze tra gruppi:

```
#complete linkage
agnes.comp<-agnes(eurodist, method="complete")
agnes.comp$ac
```

```
## [1] 0.8979532
```

```
pltree(agnes.comp, cex = 0.8, hang = -1, main = "agnes (complete)", xlab="", sub ="")
```



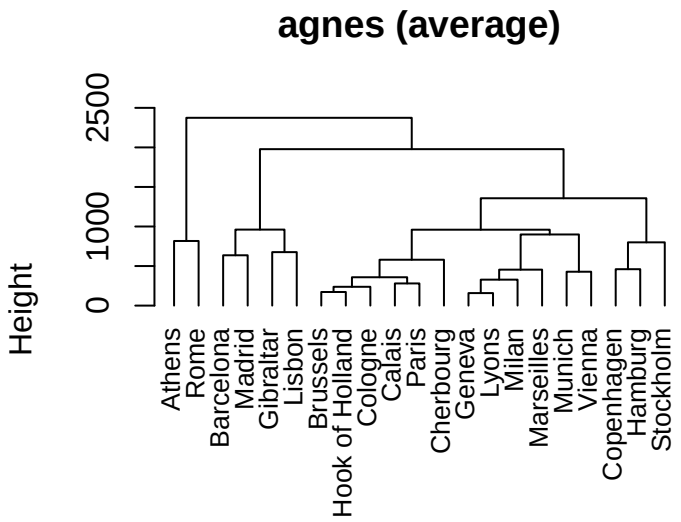
Il collegamento completo identifica bene i gruppi "estremi", che corrispondono al Sud (Roma, Atene), alla penisola iberica (Madrid, Gibilterra e Lisbona) e ad un cluster di città del Nord Europa (Amburgo, Copenaghen e Stoccolma).

Consideriamo infine il legame medio:

```
#average linkage
agnes.ave<-agnes(eurodist, method="average")
agnes.ave$ac
```

```
## [1] 0.8063934
```

```
pltree(agnes.ave, cex = 0.8, hang = -1, main = "agnes (average)", xlab="", sub ="")
```

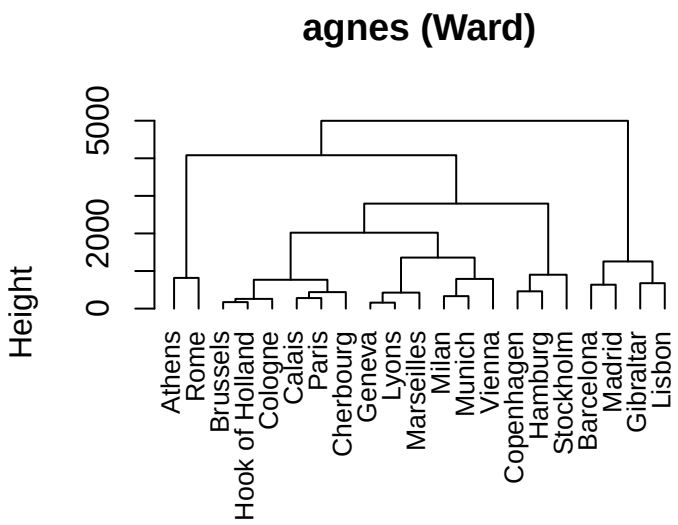


e il metodo di Ward:

```
# Ward's method
agnes.Ward<-agnes(eurodist, method="ward")
agnes.Ward$ac
```

```
## [1] 0.905946
```

```
pltree(agnes.Ward, cex = 0.8, hang = -1, main = "agnes (Ward)", xlab="", sub ="")
```



Il linkage medio e il metodo di Ward producono risultati simili: vengono ancora identificati il cluster meridionale, la penisola iberica e una regione del nord-est. Inoltre, possiamo identificare

- una regione continentale centro-settentrionale (Bruxelles, Hoek van Holland, Colonia)
- una regione Centrale (Ginevra, Lione, Marsiglia, Milano, Monaco e Vienna)
- il cluster della Francia settentrionale (Calais, Parigi, Cherbourg)

Per determinare il vettore del clustering finale occorre scegliere il numero di gruppi desiderato o il livello in corrispondenza del quale tagliare il dendrogramma. Con la funzione `cutree()` otteniamo la soluzione con $K = 4$

```
hc<-cutree(agnes.ave, 4)
table(hc)
```

```
## hc
##  1  2  3  4
##  2  4 12  3
```

```
cnames<-row.names(as.matrix(eurodist))
cnames[hc==1]
```

```
## [1] "Athens" "Rome"
```

```
cnames[hc==2]
```

```
## [1] "Barcelona" "Gibraltar" "Lisbon"    "Madrid"
```

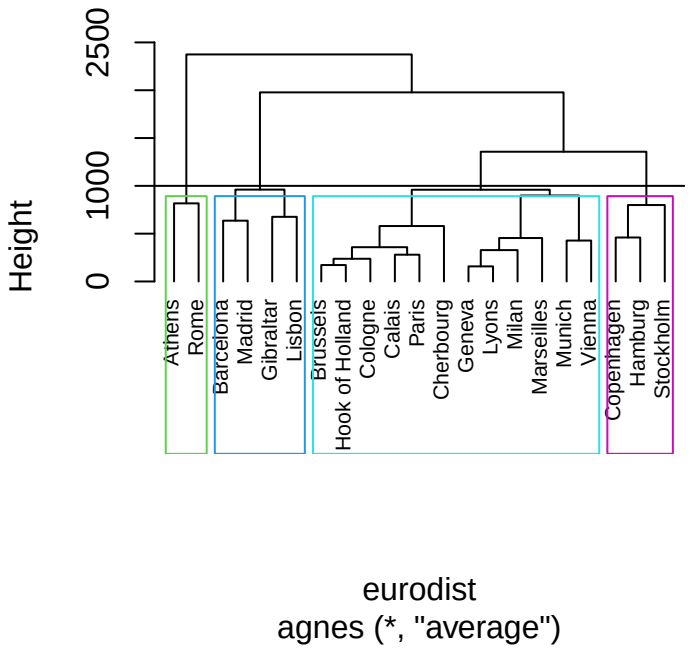
```
cnames[hc==3]
```

```
##  [1] "Brussels"        "Calais"          "Cherbourg"       "Cologne"
##  [5] "Geneva"          "Hook of Holland" "Lyons"           "Marseilles"
##  [9] "Milan"           "Munich"          "Paris"           "Vienna"
```

```
cnames[hc==4]
```

```
## [1] "Copenhagen" "Hamburg"    "Stockholm"
```

```
pltree(agnes.ave, hang=-1, cex = 0.7, main="")
abline(h=1000)
rect.hclust(agnes.ave, k = 4, border = 3:7)
```



eurodist
agnes (*, "average")

Infine, un possibile metodo per confrontare due partizioni è quello di calcolare il cosiddetto *Adjusted rand index*:

```
library(mclust)
```

```
## Package 'mclust' version 6.0.0
## Type 'citation("mclust")' for citing this R package in publications.
```

```
#?adjustedRandIndex
adjustedRandIndex(cutree(agnes.ave, 4),
                  cutree(agnes.single, 4))
```

```
## [1] 0.2474028
```

Esercizio *Si consideri il dataset 'world.cities' nel pacchetto* `maps`

```
library(maps)
#data(world.cities)
```

Si utilizzi il seguente comando `capital=subset(world.cities, capital==1)` *per selezionare le capitali nel dataset. Infine, si estraggano dal data frame le colonne relative a latitudine (lat) e longitudine (long). Si effettui una analisi di raggruppamento gerarchico con i legami medio, completo e il metodo di Ward. Utilizzando il legame 'average', si confrontino le soluzioni con* $K = 4$ *e* $K = 8$

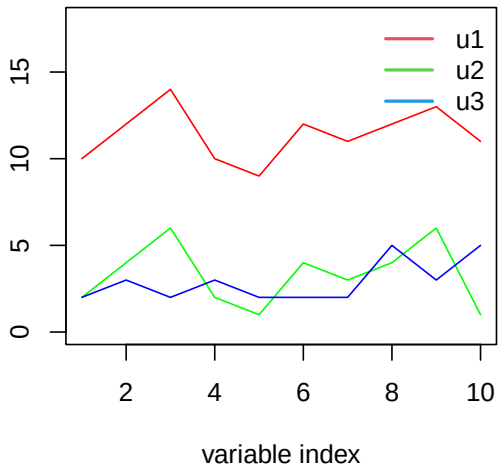
Clustering basato sulla correlazione

In alcune applicazioni può essere di interesse raggruppare le unità sulla base di un comportamento simile osservato su diverse variabili. Si pensi, ad esempio, al caso in cui si voglia raggruppare dei clienti sulla base degli acquisti già effettuati dagli stessi, in modo da ottenere gruppi di clienti con comportamenti simili. La matrice dei dati presenta gli acquirenti per riga e diversi tipi di prodotti per colonna disponibili all'acquisto. Ogni cella della matrice contiene il numero di acquisti effettuati su uno specifico prodotto.

```
u1<-c(10,12,14,10,9,12,11,12,13,11)
u2<-c(2,4,6,2,1,4,3,4,6,1)
u3<-c(2,3,2,3,2,2,2,5,3,5)

d<-rbind(u1, u2, u3) # data matrix

par(mfrow=c(1,2))
plot(1:10, u1, type="l", col="red", ylim=c(0,18),
     xlab="variable index", ylab="")
points(u2, type="l", col="green")
points(u3, type="l", col="blue")
legend("topright", c("u1", "u2", "u3"), bty="n", lty=1, lwd=2, col = 2:4)
```



In tal caso, la distanza euclidea potrebbe non essere la scelta migliore ed è necessario ricorrere ad una distanza che raggruppi insieme le unità con un profilo di acquisto simile, cioè `u1` e `u2` con riferimento all'esempio considerato. Infatti, sebbene queste unità presentino valori distanti delle variabili, sono altamente correlati.

```
r<-cor(t(d))
r
```

```
##            u1         u2          u3
## u1 1.00000000  0.9201031  0.08630145
## u2 0.92010315  1.0000000 -0.13702054
## u3 0.08630145 -0.1370205  1.00000000
```

Pertanto, la distanza in questo caso può essere definita come segue

$$d(x, y) = 1 - r_{xy}$$

dove r_{xy} è il coefficiente di correlazione di Pearson. La matrice di distanza così ottenuta può essere impiegata come input in un algoritmo gerarchico agglomerativo.

```
cor.d<-as.dist(1-r)
hc.cor<-agnes(cor.d, method ="complete")
pltree(hc.cor, main="Correlation-based HAC")
```