Standard

Paula Costa Fontichiari, Miriam Giuliani

5/6/2020

Contents

Packages needed:

```r
library(dtw)
library(dendextend)
library(factoextra)
library(NbClust)
library(cluster)
library(dplyr)
library(tidyverse)
library(lubridate)
library(dtwclust)
library(BNPTSclust)
```

1. Importing and Manipulating the Data

We imported the data set and inverted the order of the years from 2019 : 2006 to 2006 : 2019. We eliminated the NAs because we would not be able to compute the distance matrix in the further steps.

Originally, we had 157 countries (rows); after removing the NAs we got 106 countries.

```r
D <-
  read.csv2(
    file = 'GGI.csv',
    header = T,
    sep = ';',
    row.names = 1,
    na.strings = ''
  )
str(D)
```

```
## 'data.frame':    157 obs. of  14 variables:
##  $ X2019: num  0.769 0.634 0.66 0.746 0.684 0.731 0.744 0.687 0.72 0.629 ...
##  $ X2018: num  0.734 0.629 0.633 0.733 0.678 0.73 0.718 0.68 0.741 0.627 ...
##  $ X2017: num  0.728 0.629 0.64 0.732 0.677 0.731 0.709 0.676 0.743 0.632 ...
##  $ X2016: num  0.704 0.642 0.643 0.735 0.669 0.721 0.716 0.684 0.729 0.615 ...
##  $ X2015: num  0.701 0.632 0.637 0.734 0.668 0.733 0.733 0.675 0.728 0.644 ...
##  $ X2014: num  0.687 0.618 0.631 0.732 0.662 ...
##  $ X2013: num  0.641 0.597 0.666 0.72 0.663 ...
##  $ X2012: num  0.665 0.611 NA 0.721 0.664 ...
##  $ X2011: num  0.675 0.599 0.662 0.724 0.665 ...
##  $ X2010: num  0.673 0.605 0.671 0.719 0.667 ...
##  $ X2009: num  0.66 0.612 0.635 0.721 0.662 ...
##  $ X2008: num  0.659 0.611 0.603 0.721 0.668 ...
##  $ X2007: num  0.668 0.607 0.603 0.698 0.665 ...
##  $ X2006: num  0.661 0.602 0.604 0.683 NA ...
```

```r
head(D)
```

```
##           X2019 X2018 X2017 X2016 X2015  X2014  X2013  X2012  X2011  X2010
## Albania   0.769 0.734 0.728 0.704 0.701 0.6869 0.6412 0.6655 0.6748 0.6726
## Algeria   0.634 0.629 0.629 0.642 0.632 0.6182 0.5966 0.6112 0.5991 0.6052
## Angola    0.660 0.633 0.640 0.643 0.637 0.6311 0.6659     NA 0.6624 0.6712
## Argentina 0.746 0.733 0.732 0.735 0.734 0.7317 0.7195 0.7212 0.7236 0.7187
## Armenia   0.684 0.678 0.677 0.669 0.668 0.6622 0.6634 0.6636 0.6654 0.6669
## Australia 0.731 0.730 0.731 0.721 0.733 0.7409 0.7390 0.7294 0.7291 0.7271
##            X2009  X2008  X2007  X2006
```

```
## Albania   0.6601 0.6591 0.6685 0.6607
## Algeria   0.6119 0.6111 0.6068 0.6018
## Angola    0.6353 0.6032 0.6034 0.6039
## Argentina 0.7211 0.7209 0.6982 0.6829
## Armenia   0.6619 0.6677 0.6651     NA
## Australia 0.7282 0.7241 0.7204 0.7163
```

```
D <- D[c(14:1)]
names(D) <- c(2006:2019)
D <- na.omit(D)
str(D)
```

```
## 'data.frame':    106 obs. of  14 variables:
##  $ 2006: num  0.661 0.602 0.683 0.716 0.699 ...
##  $ 2007: num  0.668 0.607 0.698 0.72 0.706 ...
##  $ 2008: num  0.659 0.611 0.721 0.724 0.715 ...
##  $ 2009: num  0.66 0.612 0.721 0.728 0.703 ...
##  $ 2010: num  0.673 0.605 0.719 0.727 0.709 ...
##  $ 2011: num  0.675 0.599 0.724 0.729 0.717 ...
##  $ 2012: num  0.665 0.611 0.721 0.729 0.739 ...
##  $ 2013: num  0.641 0.597 0.72 0.739 0.744 ...
##  $ 2014: num  0.687 0.618 0.732 0.741 0.727 ...
##  $ 2015: num  0.701 0.632 0.734 0.733 0.733 0.644 0.704 0.753 0.749 0.71 ...
##  $ 2016: num  0.704 0.642 0.735 0.721 0.716 0.615 0.698 0.745 0.746 0.715 ...
##  $ 2017: num  0.728 0.629 0.732 0.731 0.709 0.632 0.719 0.739 0.758 0.72 ...
##  $ 2018: num  0.734 0.629 0.733 0.73 0.718 0.627 0.721 0.738 0.748 0.715 ...
##  $ 2019: num  0.769 0.634 0.746 0.731 0.744 0.629 0.726 0.75 0.734 0.709 ...
##  - attr(*, "na.action")= 'omit' Named int [1:51] 3 5 8 9 12 13 15 16 17 19 ...
##   ..- attr(*, "names")= chr [1:51] "Angola" "Armenia" "Azerbaijan" "Bahamas" ...
```

```
head(D)
```

```
##             2006   2007   2008   2009   2010   2011   2012   2013   2014  2015
## Albania   0.6607 0.6685 0.6591 0.6601 0.6726 0.6748 0.6655 0.6412 0.6869 0.701
## Algeria   0.6018 0.6068 0.6111 0.6119 0.6052 0.5991 0.6112 0.5966 0.6182 0.632
## Argentina 0.6829 0.6982 0.7209 0.7211 0.7187 0.7236 0.7212 0.7195 0.7317 0.734
## Australia 0.7163 0.7204 0.7241 0.7282 0.7271 0.7291 0.7294 0.7390 0.7409 0.733
## Austria   0.6986 0.7060 0.7153 0.7031 0.7091 0.7165 0.7391 0.7437 0.7266 0.733
## Bahrain   0.5894 0.5931 0.5927 0.6136 0.6217 0.6232 0.6298 0.6334 0.6261 0.644
##            2016  2017  2018  2019
## Albania   0.704 0.728 0.734 0.769
## Algeria   0.642 0.629 0.629 0.634
## Argentina 0.735 0.732 0.733 0.746
## Australia 0.721 0.731 0.730 0.731
## Austria   0.716 0.709 0.718 0.744
## Bahrain   0.615 0.632 0.627 0.629
```

2. Hierarchical Clustering

2.1 Using Dynamic Time Warping Distance

2.1.1 Computing the Distance Matrix

```
distance <- dist(D, method = 'DTW')
```

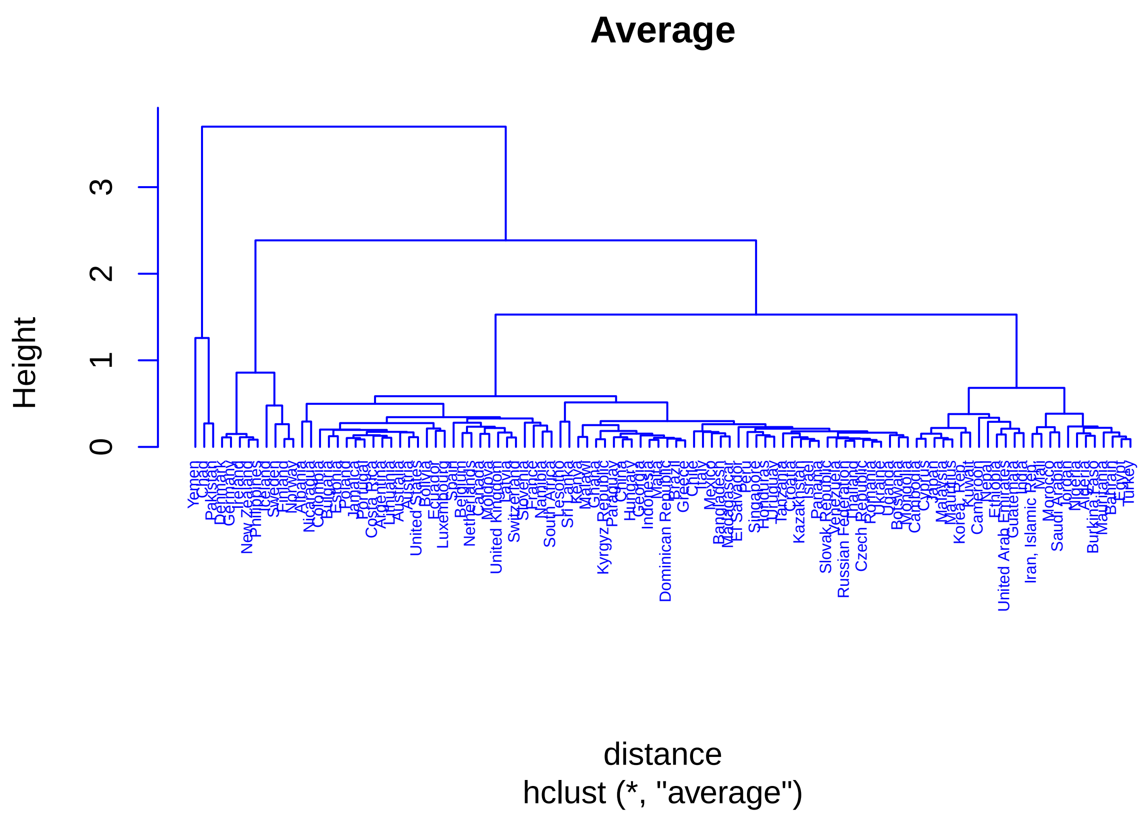
2.1.2 Applying the Hierarchical Algorithm

We tried different linkage methods.

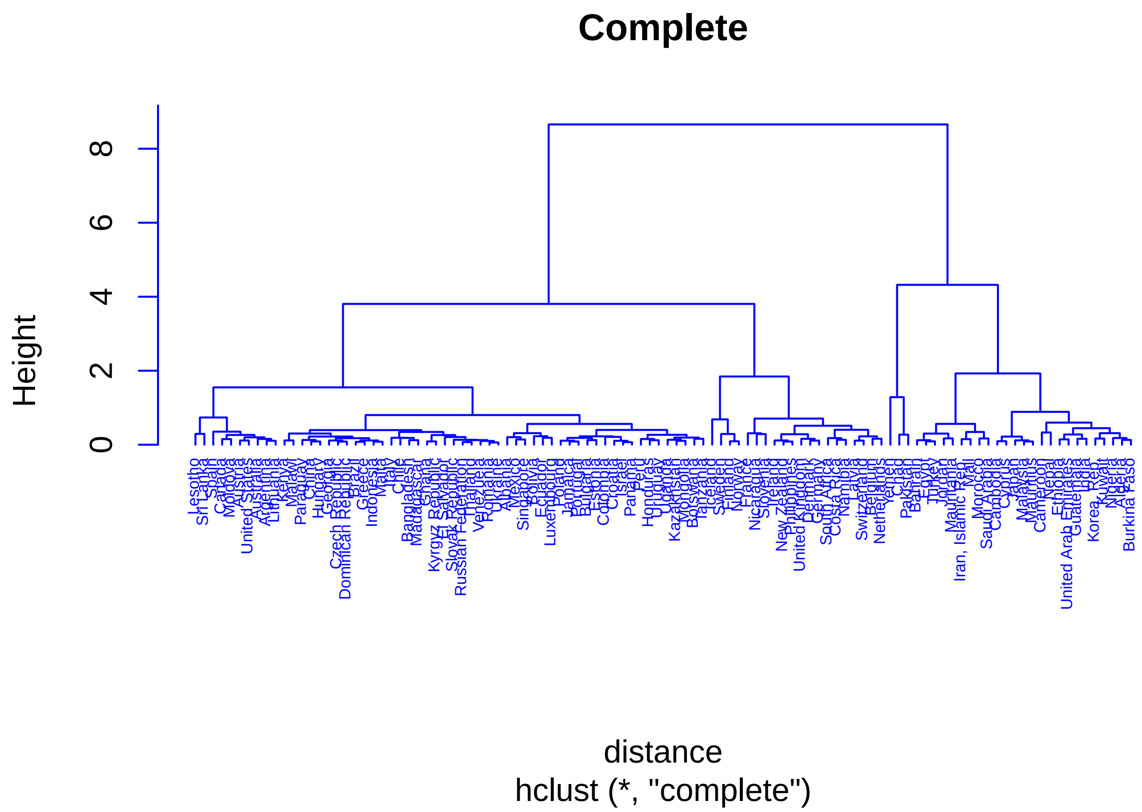
```r
hc <- hclust(distance, method = 'average')
hc2 <- hclust(distance, method = 'complete')
hc3 <- hclust(distance, method = 'single')
```

2.1.3 Plotting the Dendrograms

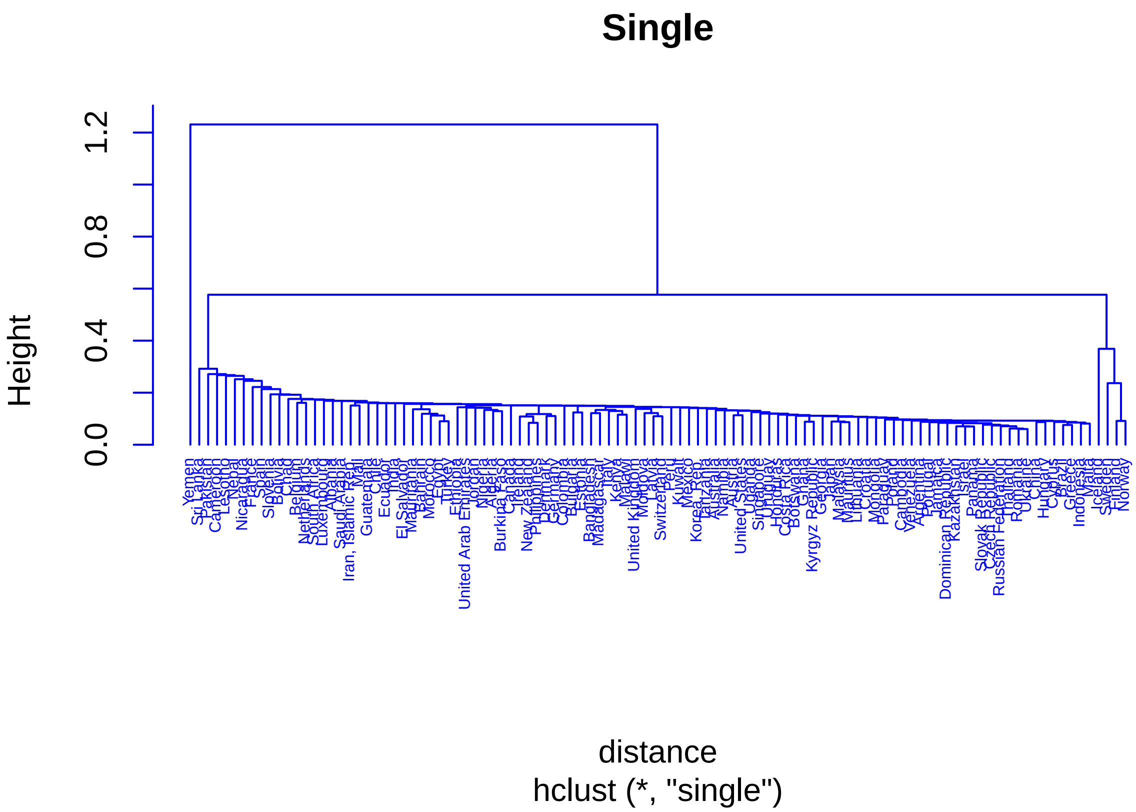
```r
plot(
  hc,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Average'
)
```




```r
plot(
  hc2,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Complete'
)
```




```r
plot(
  hc3,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Single'
)
```

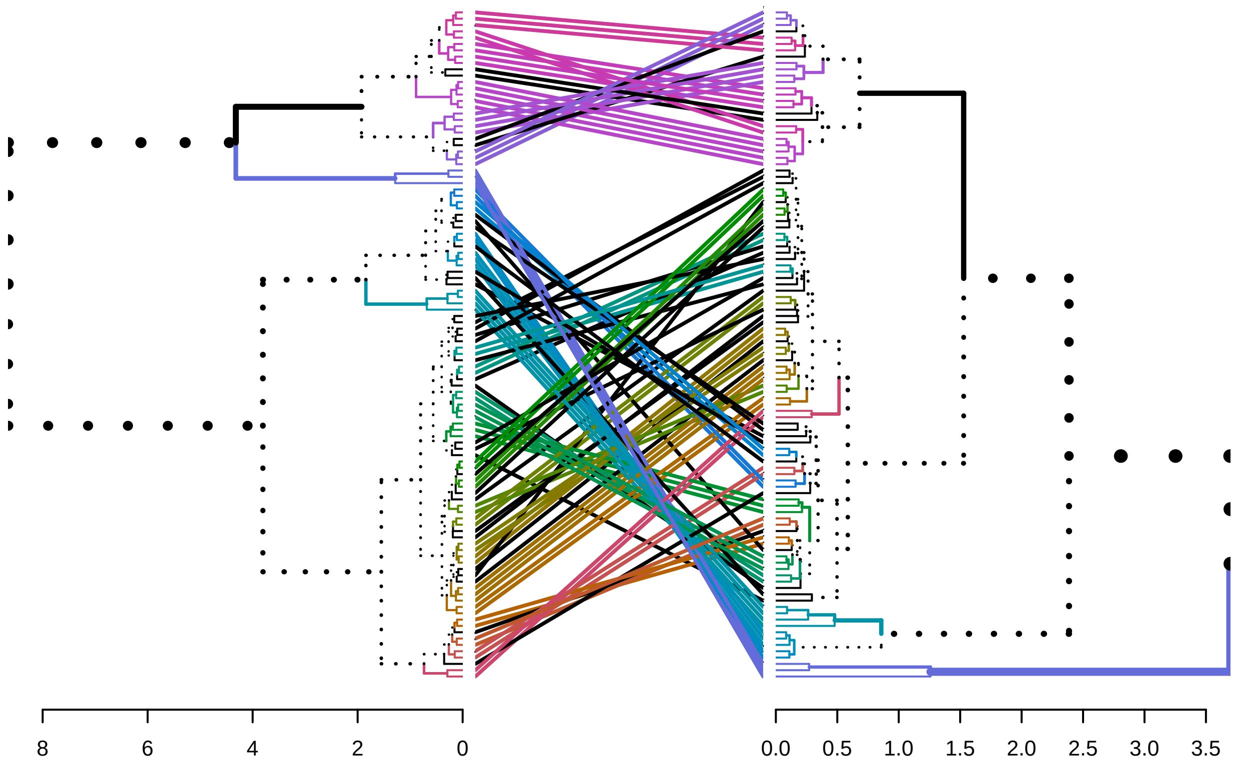


2.1.4 Comparing Complete and Average Linkage

Tanglegram plots the two dendrograms, side by side, with their labels connected by lines. The output displays *unique* nodes, with a combination of labels/items not present in the other tree, highlighted with dashed lines.

Entanglement is a measure between 1 (*full entanglement*) and 0 (*no entanglement*). A lower entanglement coefficient corresponds to a good alignment.

```
dend1 <- as.dendrogram (hc2) #Complete
dend2 <- as.dendrogram (hc) #Average
tanglegram(
  dend1,
  dend2,
  common_subtrees_color_branches = TRUE,
  margin_inner = 0.5,
  lwd = 1.9
)
```



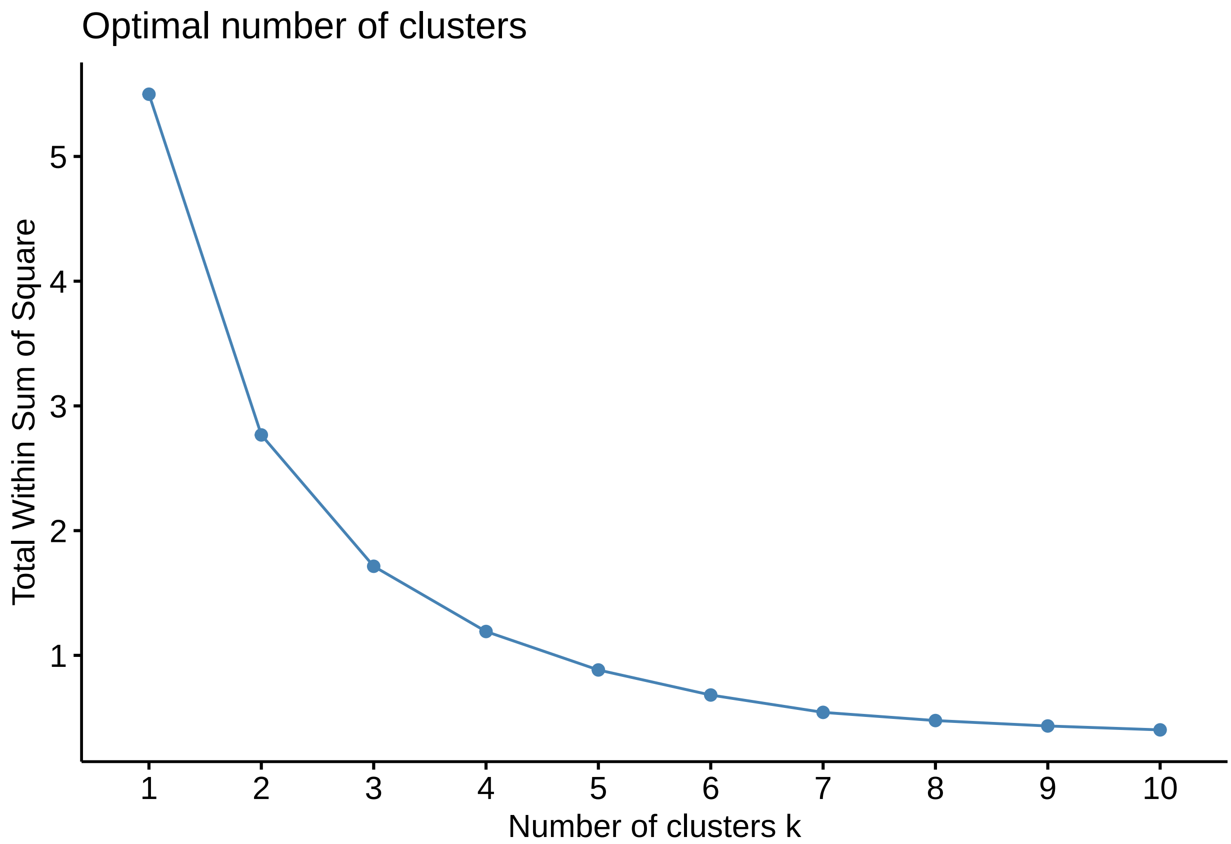

```r
entanglement(dend1, dend2)
```

```
## [1] 0.3991899
```

2.1.5 Choice and Validation of the Optimal Number of Clusters

Elbow Method:

```r
fviz_nbclust(D, FUN = hcut, method = "wss")
```



Silhouette Method:

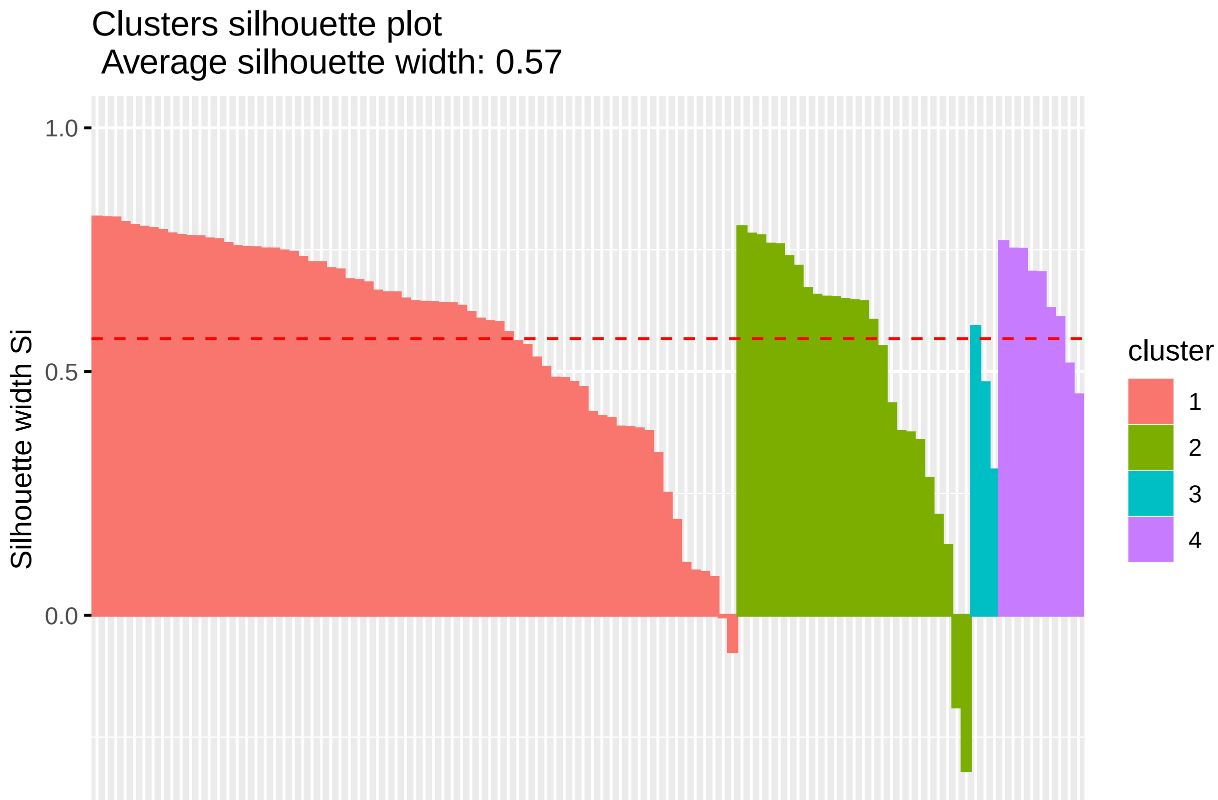
```
# method 1
a <-
  NbClust(
    data = D,
    diss = distance,
    distance = NULL,
    min.nc = 2,
    max.nc = 7,
    method = 'average',
    index = 'silhouette'
  )
fviz_nbclust(a, diss = dist(D, distance), method = "silhouette")
```

```
## Number_clusters     Value_Index
##          2.0000          0.6284
```

```
#method 2

hc.cut <- hcut(distance, k = 4, hc_method = "average")
hc.cut3 <- hcut(distance, k = 3, hc_method = "average")
par(mfrow = c(1, 2))
fviz_silhouette(hc.cut, label = F, print.summary = TRUE)
```

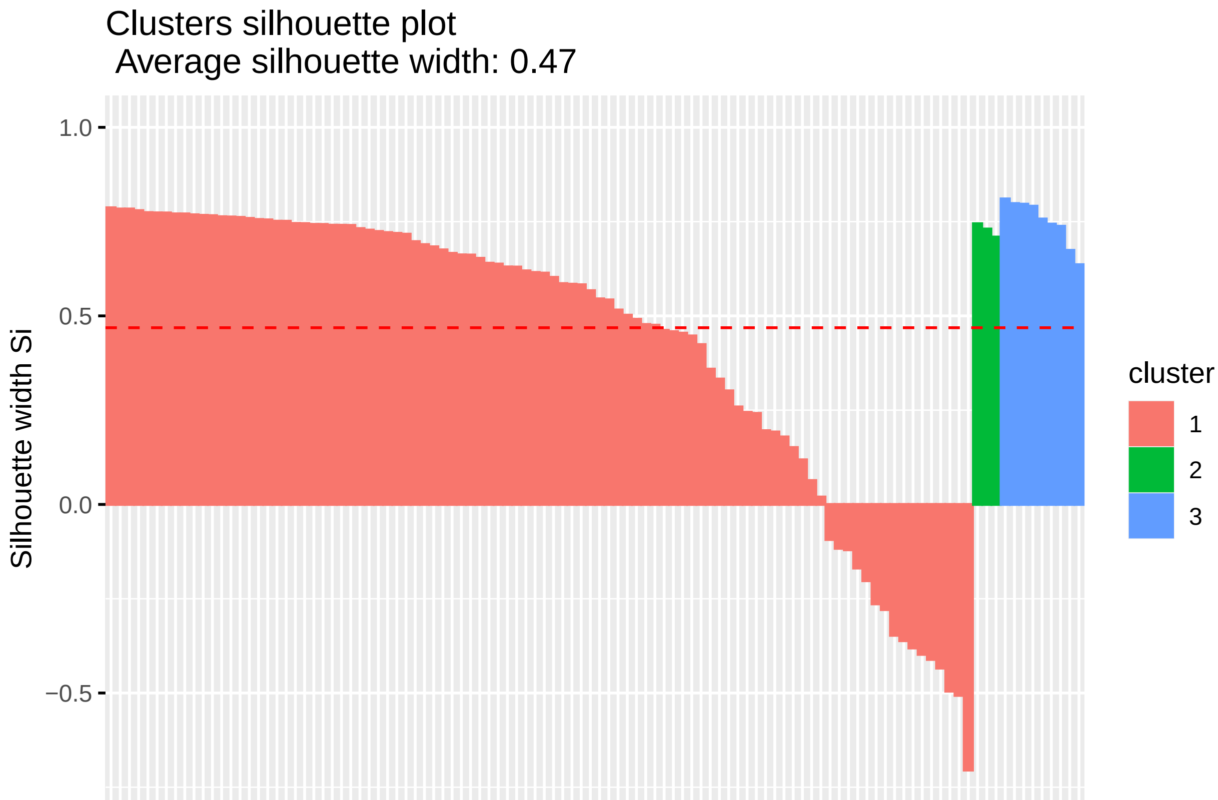
```
##   cluster size ave.sil.width
## 1       1   69          0.58
## 2       2   25          0.51
```

```
## 3         3    3           0.46
## 4         4    9           0.65
```

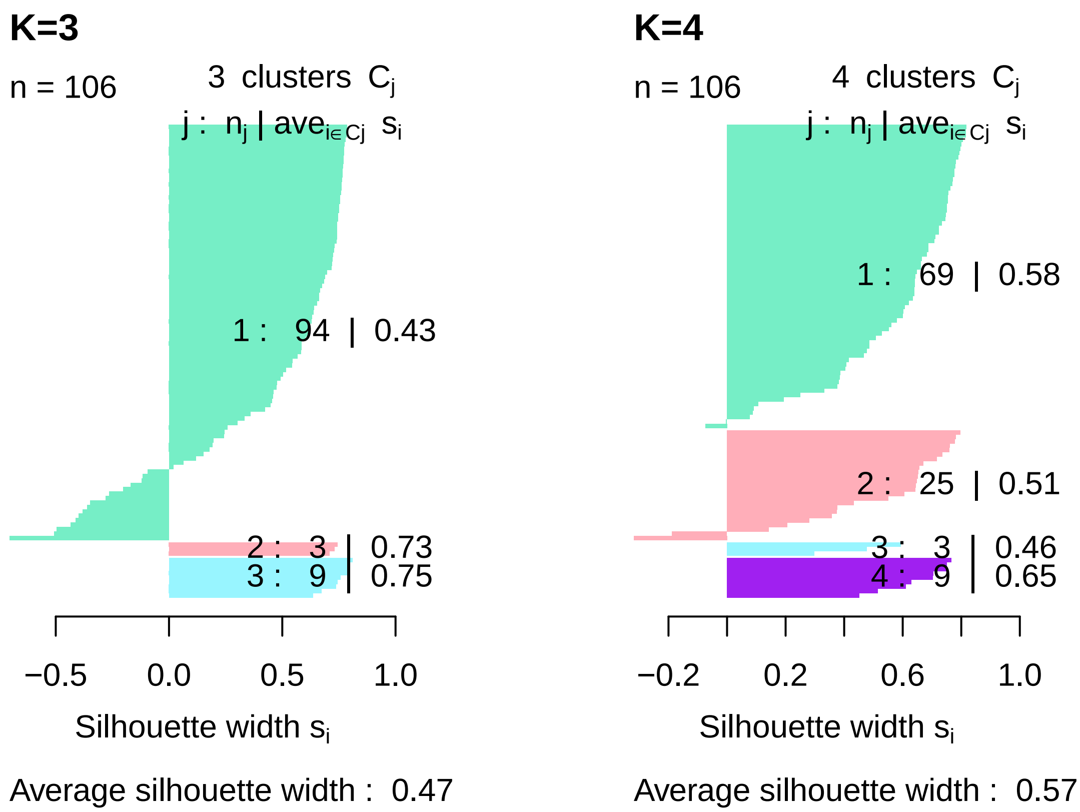



```r
fviz_silhouette(hc.cut3, label = F, print.summary = TRUE)
```

```
##   cluster size ave.sil.width
## 1       1   94          0.43
## 2       2    3          0.73
## 3       3    9          0.75
```



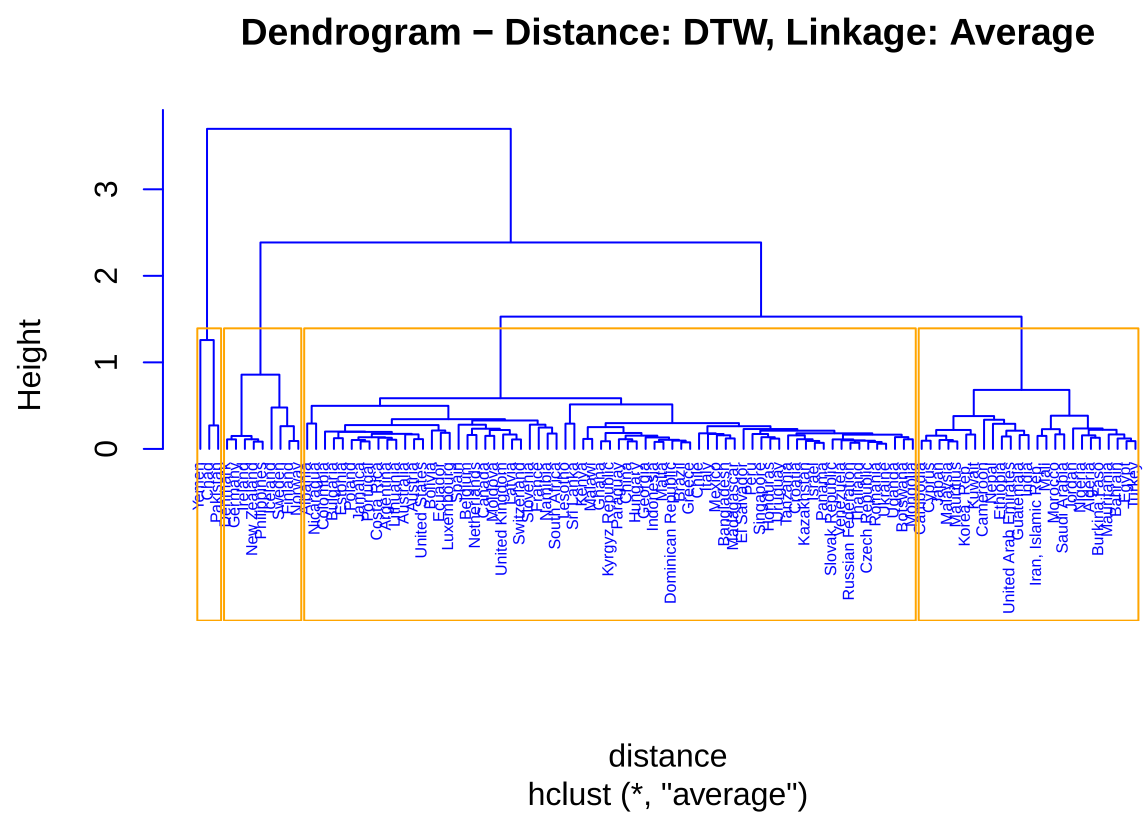

```r
#method 3
ar <- agnes(distance, method = 'average')
si3 <- silhouette(cutree(ar, k = 3),
                  distance)
si4 <- silhouette(cutree(ar, k = 4),
                  distance)


plot(
  si3,
  border = NA,
  main = 'K=3',
  nmax = 80,
  cex.names = 0.7,
  col = c('aquamarine2', 'lightpink1', 'cadetblue1')
)

plot(
  si4,
  nmax = 80,
  cex.names = 0.7,
  col = c('aquamarine2', 'lightpink1', 'cadetblue1', 'purple'),
  main = "K=4",
  border = NA
)
```



2.1.6 Cutting the Dendrogram

By comparison, we chose the *average* linkage.

```
plot(
  hc,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Dendrogram - Distance: DTW, Linkage: Average'
)
rect.hclust(hc, k = 4, border = 'orange')
```



2.1.7 Extracting the Clusters

```
cut_avg <- cutree(hc, k = 4)
D_ <- mutate(D, cluster = cut_avg)
count(D_, cluster)
```

```
## # A tibble: 4 x 2
##   cluster     n
##     <int> <int>
## 1       1    69
## 2       2    25
## 3       3     3
## 4       4     9
```

```
rownames(D_) <- rownames(D)
D_ <- D_[15:1]
head(D_)
```

```
##            cluster  2019  2018  2017  2016  2015   2014   2013   2012   2011
## Albania          1 0.769 0.734 0.728 0.704 0.701 0.6869 0.6412 0.6655 0.6748
## Algeria          2 0.634 0.629 0.629 0.642 0.632 0.6182 0.5966 0.6112 0.5991
## Argentina        1 0.746 0.733 0.732 0.735 0.734 0.7317 0.7195 0.7212 0.7236
## Australia        1 0.731 0.730 0.731 0.721 0.733 0.7409 0.7390 0.7294 0.7291
## Austria          1 0.744 0.718 0.709 0.716 0.733 0.7266 0.7437 0.7391 0.7165
## Bahrain          2 0.629 0.627 0.632 0.615 0.644 0.6261 0.6334 0.6298 0.6232
##              2010   2009   2008   2007   2006
## Albania    0.6726 0.6601 0.6591 0.6685 0.6607
```

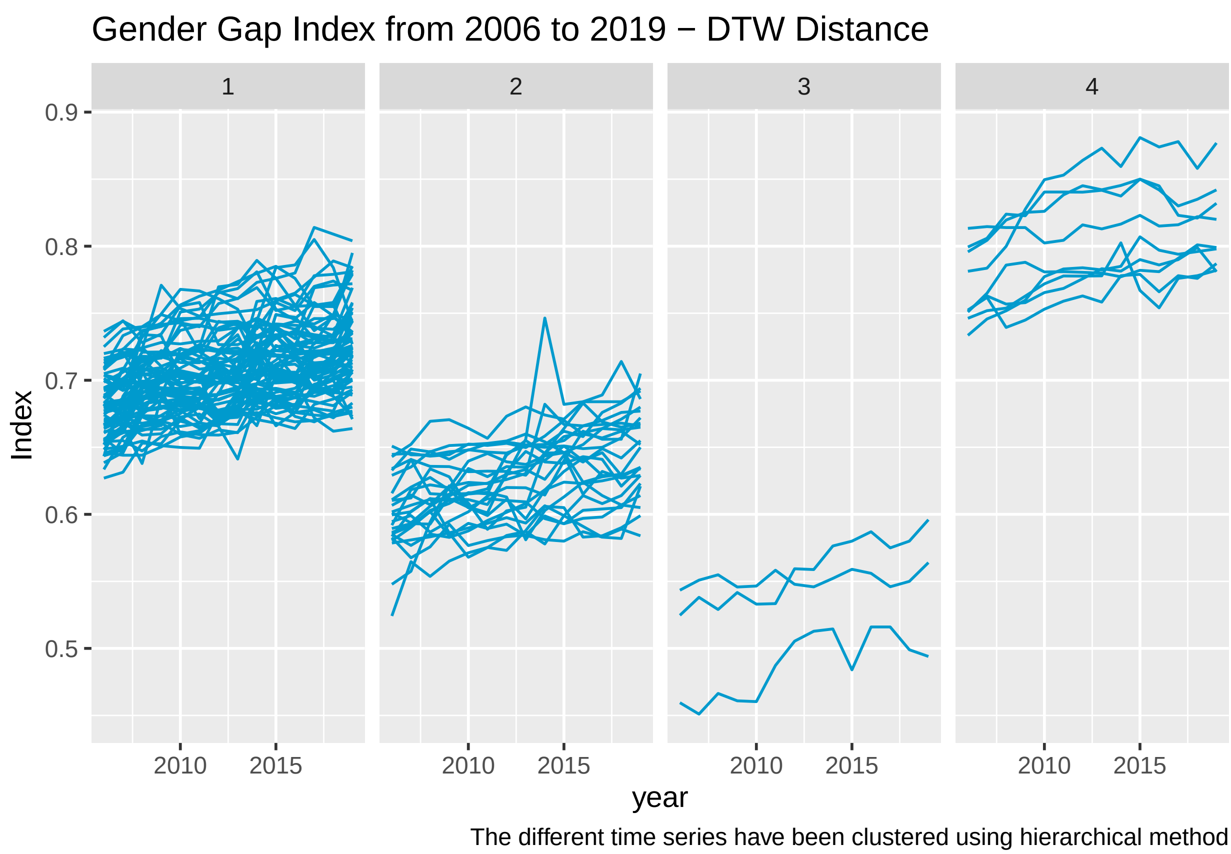
```
## Algeria   0.6052 0.6119 0.6111 0.6068 0.6018
## Argentina 0.7187 0.7211 0.7209 0.6982 0.6829
## Australia 0.7271 0.7282 0.7241 0.7204 0.7163
## Austria   0.7091 0.7031 0.7153 0.7060 0.6986
## Bahrain   0.6217 0.6136 0.5927 0.5931 0.5894
```

```
fnt <- function(i) {
  cluster <- rownames(D_)[D_$cluster == i]
  return(cluster)
}
sapply(1:4, FUN = fnt)
```

```
## [[1]]
##  [1] "Albania"            "Argentina"          "Australia"
##  [4] "Austria"            "Bangladesh"         "Belgium"
##  [7] "Bolivia"            "Botswana"           "Brazil"
## [10] "Bulgaria"           "Canada"             "Chile"
## [13] "China"              "Colombia"           "Costa Rica"
## [16] "Croatia"            "Czech Republic"     "Dominican Republic"
## [19] "Ecuador"            "El Salvador"        "Estonia"
## [22] "France"             "Georgia"            "Ghana"
## [25] "Greece"             "Honduras"           "Hungary"
## [28] "Indonesia"          "Israel"             "Italy"
## [31] "Jamaica"            "Kazakhstan"         "Kenya"
## [34] "Kyrgyz Republic"    "Latvia"             "Lesotho"
## [37] "Lithuania"          "Luxembourg"         "Madagascar"
## [40] "Malawi"             "Malta"              "Mexico"
## [43] "Moldova"            "Mongolia"           "Namibia"
## [46] "Netherlands"        "Nicaragua"          "Panama"
## [49] "Paraguay"           "Peru"               "Poland"
## [52] "Portugal"           "Romania"            "Russian Federation"
## [55] "Singapore"          "Slovak Republic"    "Slovenia"
## [58] "South Africa"       "Spain"              "Sri Lanka"
## [61] "Switzerland"        "Tanzania"           "Thailand"
## [64] "Uganda"             "Ukraine"            "United Kingdom"
## [67] "United States"      "Uruguay"            "Venezuela"
##
## [[2]]
##  [1] "Algeria"              "Bahrain"              "Burkina Faso"
##  [4] "Cambodia"             "Cameroon"             "Cyprus"
##  [7] "Egypt"                "Ethiopia"             "Guatemala"
## [10] "India"                "Iran, Islamic Rep."   "Japan"
## [13] "Jordan"               "Korea, Rep."          "Kuwait"
## [16] "Malaysia"             "Mali"                 "Mauritania"
## [19] "Mauritius"            "Morocco"              "Nepal"
## [22] "Nigeria"              "Saudi Arabia"         "Turkey"
## [25] "United Arab Emirates"
##
## [[3]]
## [1] "Chad"     "Pakistan" "Yemen"
##
## [[4]]
## [1] "Denmark"     "Finland"     "Germany"     "Iceland"     "Ireland"
## [6] "New Zealand" "Norway"      "Philippines" "Sweden"
```

2.1.8 Visualizing the Clusters

```
country <- rownames(D_)
D_ <- cbind(country, D_)
D_long <- D_ %>%
  pivot_longer(
    cols = c(-country, -cluster),
    names_to = "year",
    values_to = "Index"
  ) %>%
  mutate(year = ymd(paste0(year, "-01-01")))

ggplot() +
  geom_line(data = D_long,
            aes(y = Index, x = year, group = country),
            colour = "deepskyblue3") +
  facet_wrap(~ cluster, nrow = 1) +
  labs(title = "Gender Gap Index from 2006 to 2019 - DTW Distance",
       caption = "The different time series have been clustered using hierarchical method")
```



2.2 Using Euclidean Distance

2.2.1 Computing the Distance Matrix

```
distance2 <- dist(D,method = "euclidean")
```

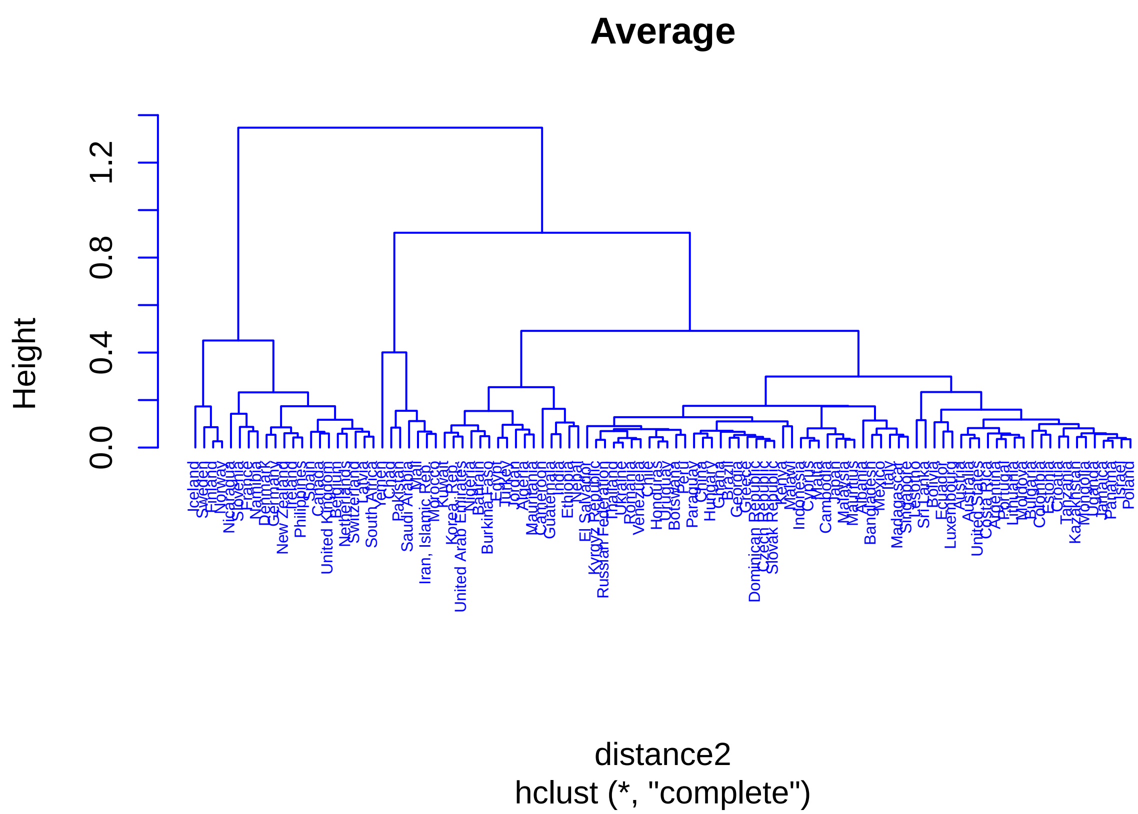
2.2.2 Applying the Hierarchical Algorithm

We also included *ward* linkage, since it exploits Euclidean distance.

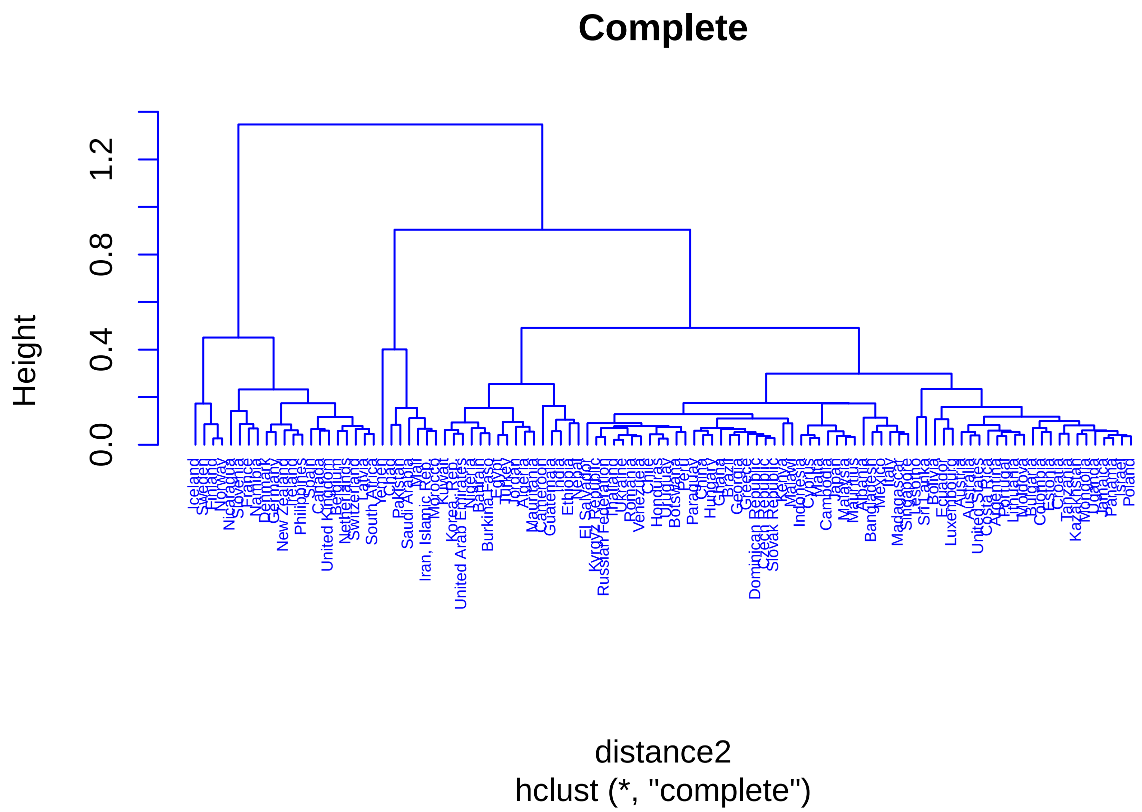
```
hce <- hclust(d = distance2, method = 'average')
hce2 <- hclust(d = distance2, method = 'complete')
hce3 <- hclust(d = distance2, method = 'single')
hce4 <- hclust(d = distance2, method = 'ward.D2')
```

2.2.3 Plotting the Dendrograms

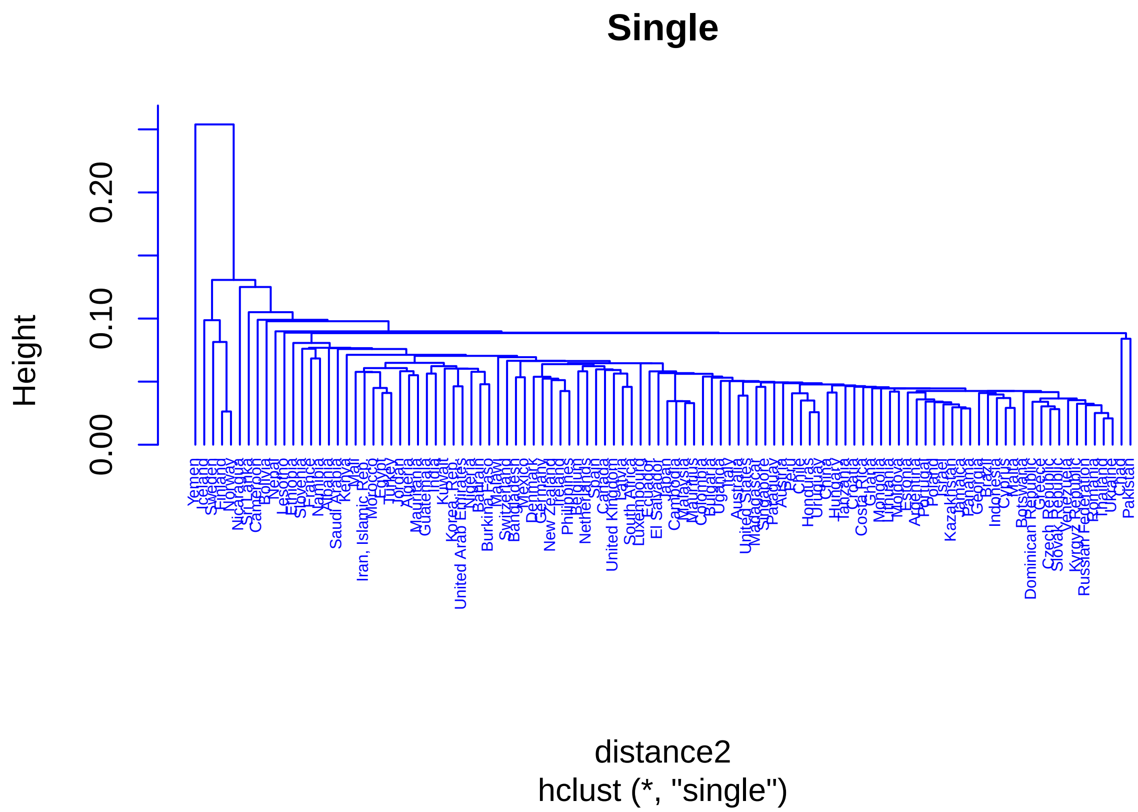
```
plot(
  hce2,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Average'
)
```



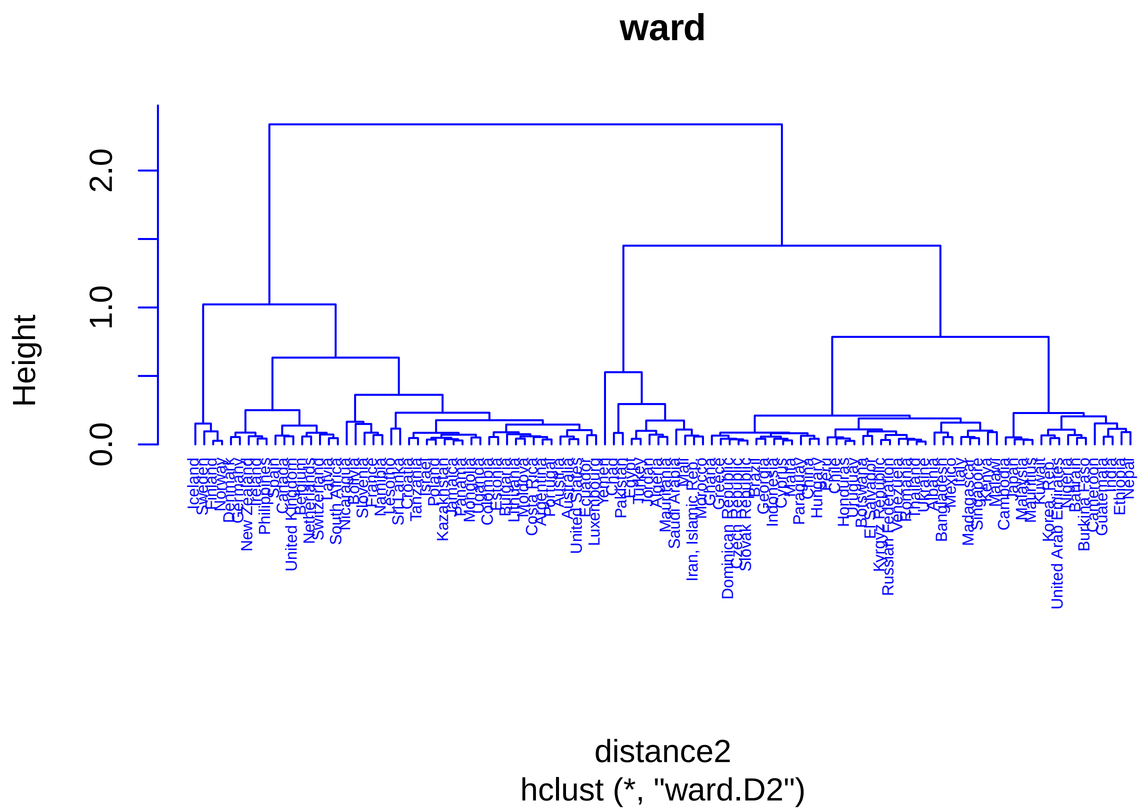

```
plot(
  hce2,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Complete'
)
```




```
plot(
  hce3,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Single'
)
```




```r
plot(
  hce4,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'ward'
)
```



2.2.4 Choice and Validation of the Optimal Number of Clusters

Elbow Method: same as the one for DTW.

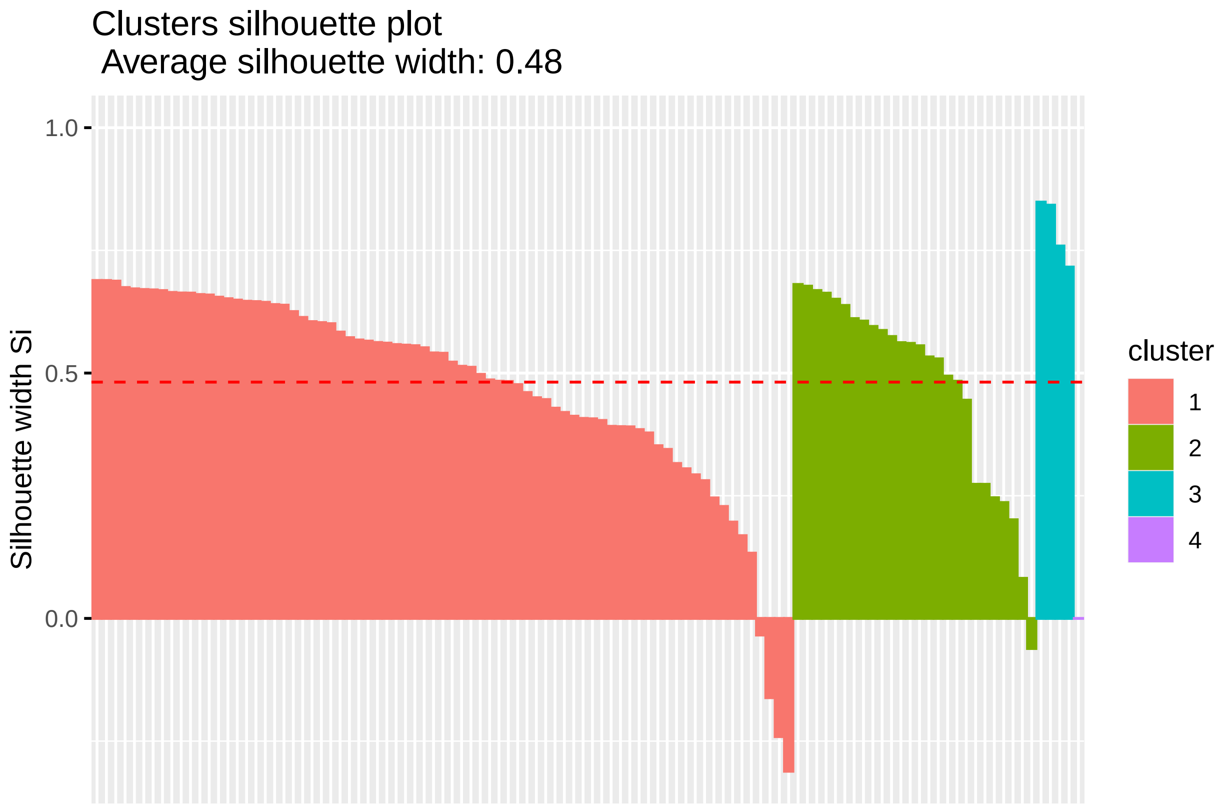
Silhouette Method:

```
 # method 1
E <-
  NbClust(
    data = D,
    diss = distance2,
    distance = NULL,
    min.nc = 2,
    max.nc = 7,
    method = 'average',
    index = 'silhouette'
  )
fviz_nbclust(a, diss = dist(D, distance2), method = "silhouette")
```

```
## Number_clusters     Value_Index
##          2.0000          0.6284
```

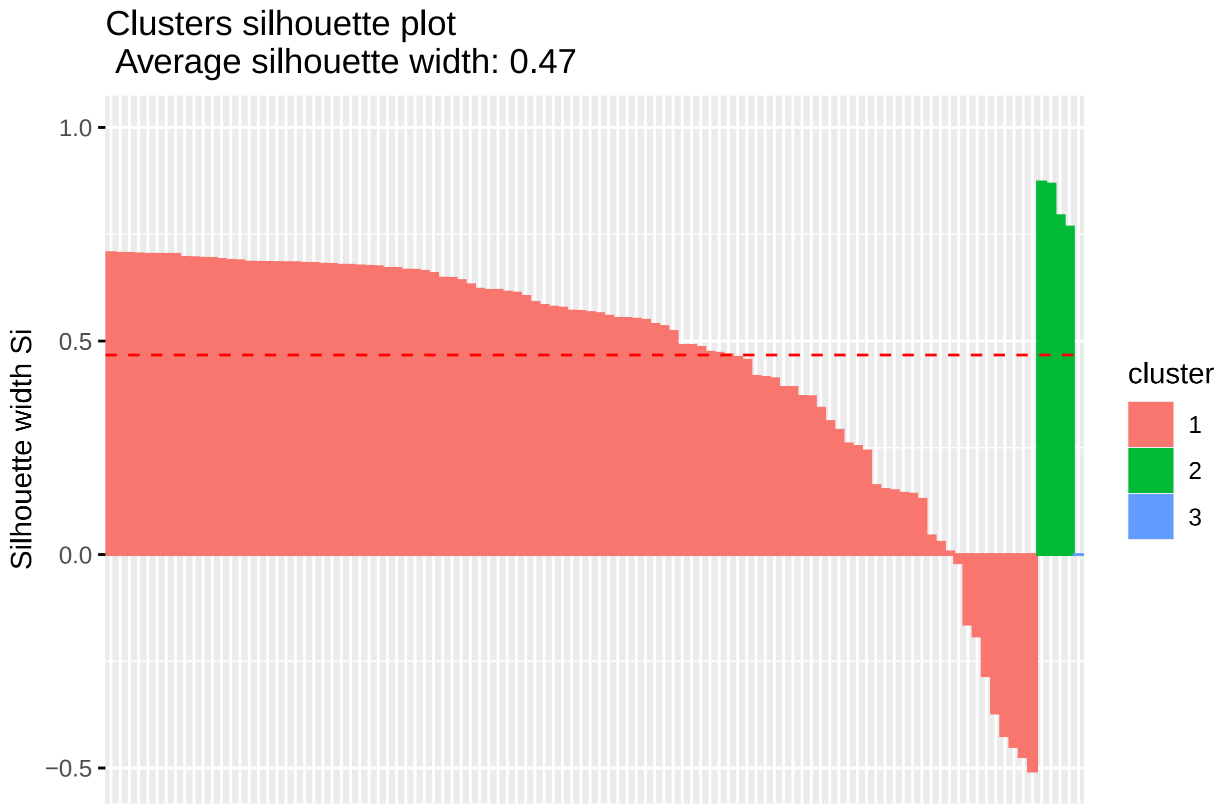
```
#method 2

hc.cute <- hcut(distance2, k = 4, hc_method = "average")
hc.cute3 <- hcut(distance2, k = 3, hc_method = "average")
par(mfrow = c(1, 2))
fviz_silhouette(hc.cute, label = F, print.summary = TRUE)
```

```
##   cluster size ave.sil.width
## 1       1   75          0.47
## 2       2   26          0.48
## 3       3    4          0.79
## 4       4    1          0.00
```

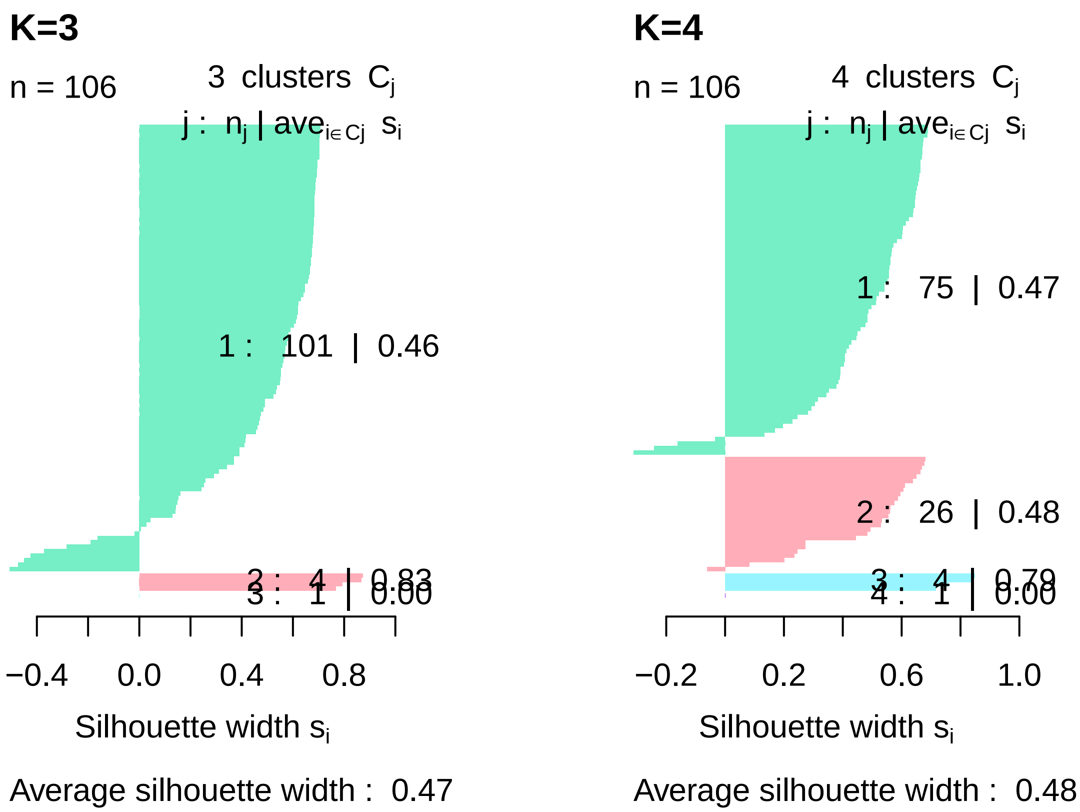



```r
fviz_silhouette(hc.cute3, label = F, print.summary = TRUE)
```

```
##   cluster size ave.sil.width
## 1       1  101          0.46
## 2       2    4          0.83
## 3       3    1          0.00
```

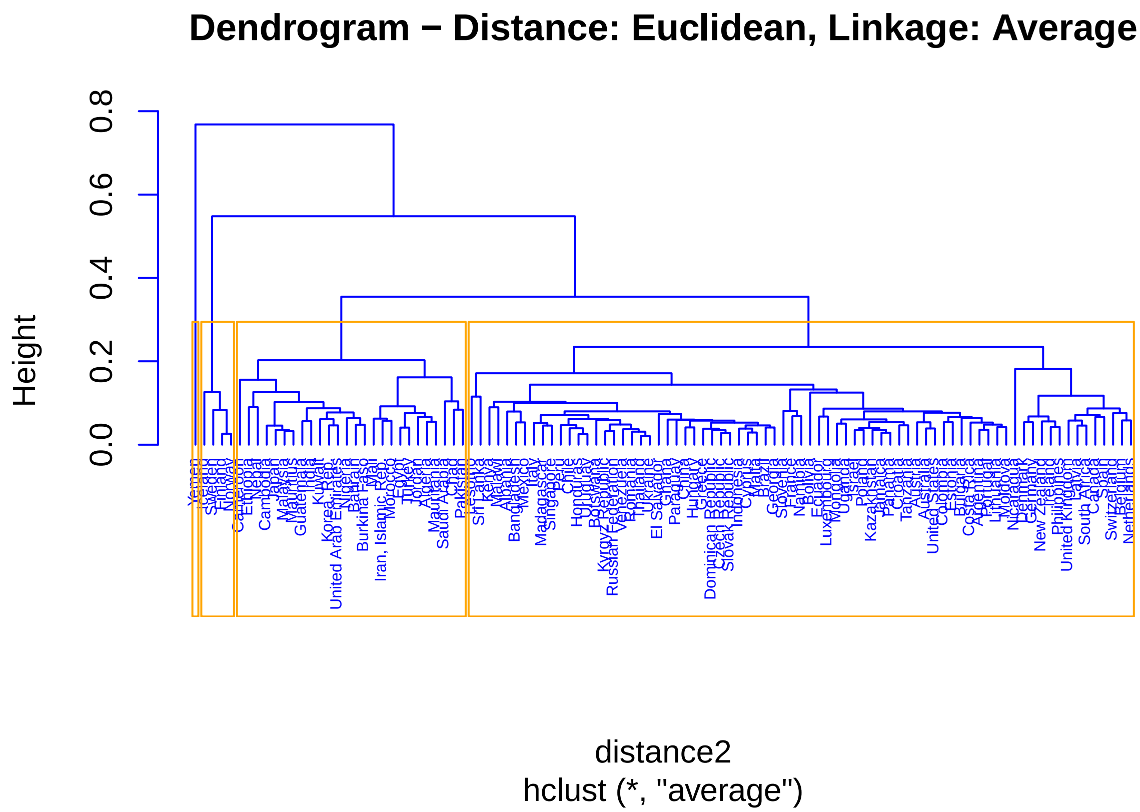



```r
#method 3
are <- agnes(distance2, method = 'average')
sie3 <- silhouette(cutree(are, k = 3),
                   distance2)
sie4 <- silhouette(cutree(are, k = 4),
                   distance2)


plot(
  sie3,
  border = NA,
  main = 'K=3',
  nmax = 80,
  cex.names = 0.7,
  col = c('aquamarine2', 'lightpink1', 'cadetblue1')
)

plot(
  sie4,
  nmax = 80,
  cex.names = 0.7,
  col = c('aquamarine2', 'lightpink1', 'cadetblue1', 'purple'),
  main = "K=4",
  border = NA
)
```



2.2.5 Cutting the Dendrogram

```r
plot(
  hce,
  cex = .5,
  hang = -1,
  col = 'blue',
  main = 'Dendrogram - Distance: Euclidean, Linkage: Average'
)
rect.hclust(hce, k = 4, border = 'orange')
```



2.2.6 Extracting the Clusters

```r
cute <- cutree(hce, k = 4)
De <- mutate(D, cluster = cute)
count(De, cluster)
```

```
## # A tibble: 4 x 2
##   cluster     n
##     <int> <int>
## 1       1    75
## 2       2    26
## 3       3     4
## 4       4     1
```

```r
rownames(De) <- rownames(D)
De <- De[15:1]
head(De)
```

```
##           cluster  2019  2018  2017  2016  2015   2014   2013   2012   2011
## Albania         1 0.769 0.734 0.728 0.704 0.701 0.6869 0.6412 0.6655 0.6748
## Algeria         2 0.634 0.629 0.629 0.642 0.632 0.6182 0.5966 0.6112 0.5991
## Argentina       1 0.746 0.733 0.732 0.735 0.734 0.7317 0.7195 0.7212 0.7236
## Australia       1 0.731 0.730 0.731 0.721 0.733 0.7409 0.7390 0.7294 0.7291
## Austria         1 0.744 0.718 0.709 0.716 0.733 0.7266 0.7437 0.7391 0.7165
## Bahrain         2 0.629 0.627 0.632 0.615 0.644 0.6261 0.6334 0.6298 0.6232
##             2010   2009   2008   2007   2006
## Albania   0.6726 0.6601 0.6591 0.6685 0.6607
```

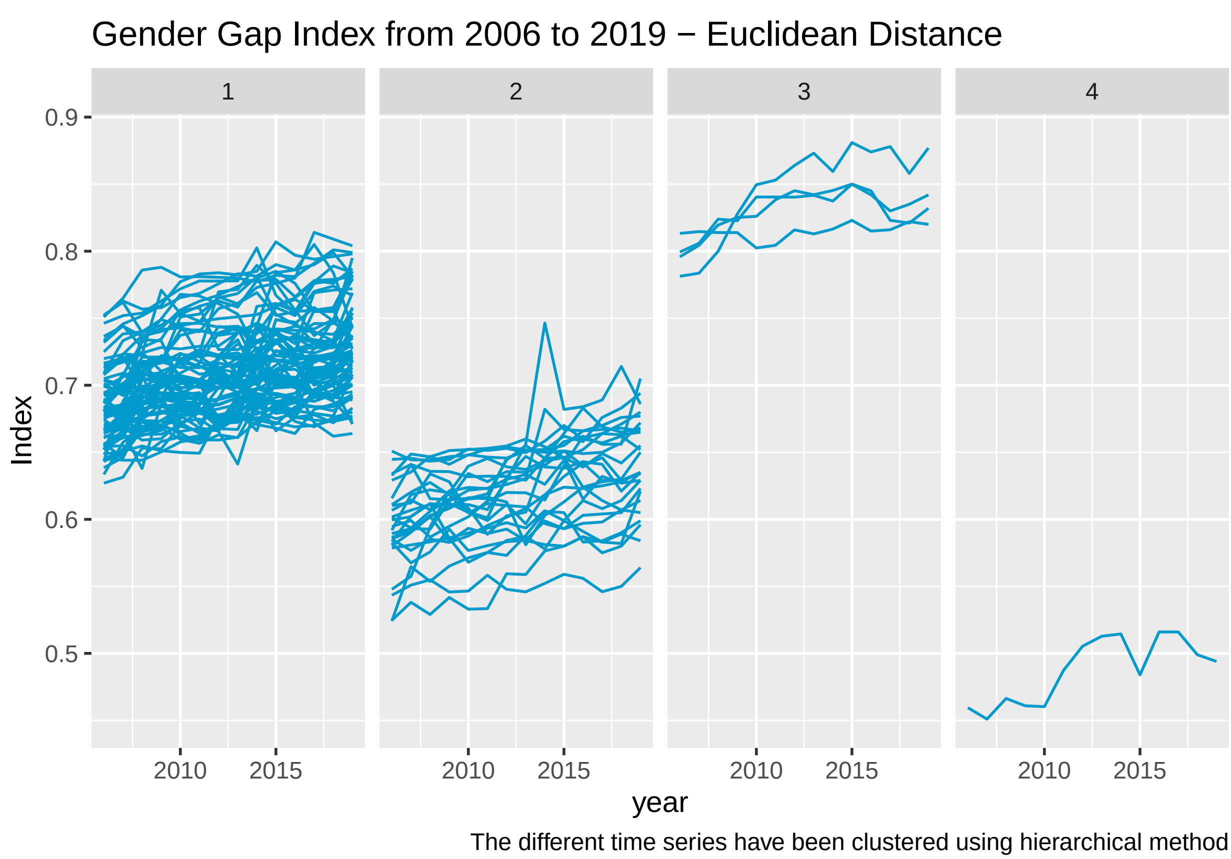
```
## Algeria   0.6052 0.6119 0.6111 0.6068 0.6018
## Argentina 0.7187 0.7211 0.7209 0.6982 0.6829
## Australia 0.7271 0.7282 0.7241 0.7204 0.7163
## Austria   0.7091 0.7031 0.7153 0.7060 0.6986
## Bahrain   0.6217 0.6136 0.5927 0.5931 0.5894
```

```r
fnte <- function(i) {
  cluster <- rownames(De)[De$cluster == i]
  return(cluster)
}
sapply(1:4, FUN = fnte)
```

```
## [[1]]
##  [1] "Albania"            "Argentina"          "Australia"
##  [4] "Austria"            "Bangladesh"         "Belgium"
##  [7] "Bolivia"            "Botswana"           "Brazil"
## [10] "Bulgaria"           "Canada"             "Chile"
## [13] "China"              "Colombia"           "Costa Rica"
## [16] "Croatia"            "Cyprus"             "Czech Republic"
## [19] "Denmark"            "Dominican Republic" "Ecuador"
## [22] "El Salvador"        "Estonia"            "France"
## [25] "Georgia"            "Germany"            "Ghana"
## [28] "Greece"             "Honduras"           "Hungary"
## [31] "Indonesia"          "Ireland"            "Israel"
## [34] "Italy"              "Jamaica"            "Kazakhstan"
## [37] "Kenya"              "Kyrgyz Republic"    "Latvia"
## [40] "Lesotho"            "Lithuania"          "Luxembourg"
## [43] "Madagascar"         "Malawi"             "Malta"
## [46] "Mexico"             "Moldova"            "Mongolia"
## [49] "Namibia"            "Netherlands"        "New Zealand"
## [52] "Nicaragua"          "Panama"             "Paraguay"
## [55] "Peru"               "Philippines"        "Poland"
## [58] "Portugal"           "Romania"            "Russian Federation"
## [61] "Singapore"          "Slovak Republic"    "Slovenia"
## [64] "South Africa"       "Spain"              "Sri Lanka"
## [67] "Switzerland"        "Tanzania"           "Thailand"
## [70] "Uganda"             "Ukraine"            "United Kingdom"
## [73] "United States"      "Uruguay"            "Venezuela"
##
## [[2]]
##  [1] "Algeria"              "Bahrain"              "Burkina Faso"
##  [4] "Cambodia"             "Cameroon"             "Chad"
##  [7] "Egypt"                "Ethiopia"             "Guatemala"
## [10] "India"                "Iran, Islamic Rep."   "Japan"
## [13] "Jordan"               "Korea, Rep."          "Kuwait"
## [16] "Malaysia"             "Mali"                 "Mauritania"
## [19] "Mauritius"            "Morocco"              "Nepal"
## [22] "Nigeria"              "Pakistan"             "Saudi Arabia"
## [25] "Turkey"               "United Arab Emirates"
##
## [[3]]
## [1] "Finland" "Iceland" "Norway"  "Sweden"
##
## [[4]]
## [1] "Yemen"
```

2.2.7 Visualizing the Clusters

```
country <- rownames(De)
De <- cbind(country, De)
Delong <- De %>%
  pivot_longer(
    cols = c(-country, -cluster),
    names_to = "year",
    values_to = "Index"
  ) %>%
  mutate(year = ymd(paste0(year, "-01-01")))

ggplot() +
  geom_line(data = Delong,
            aes(y = Index, x = year, group = country),
            colour = "deepskyblue3") +
  facet_wrap(~ cluster, nrow = 1) +
  labs(title = "Gender Gap Index from 2006 to 2019 - Euclidean Distance",
       caption = "The different time series have been clustered using hierarchical method")
```

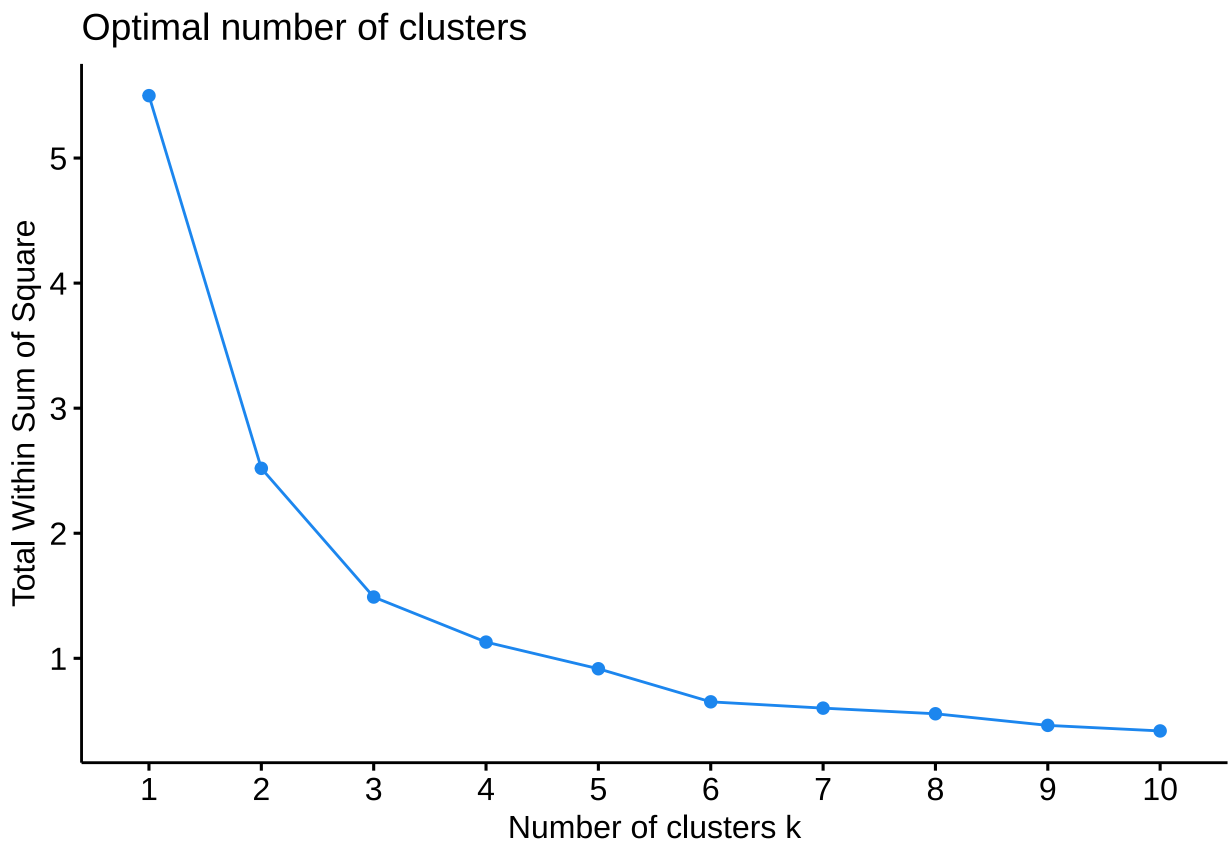


3. K-Medoids

3.1 Choosing the Number of Clusters

Elbow Method:

```r
fviz_nbclust(
  D,
  FUNcluster = pam,
  method = "wss",
  diss = NULL,
  k.max = 10,
  linecolor = 'dodgerblue2'
)
```



3.2 Applying the K-Medoids

We apply the algorithm for $k = 3$ and $k = 4$.

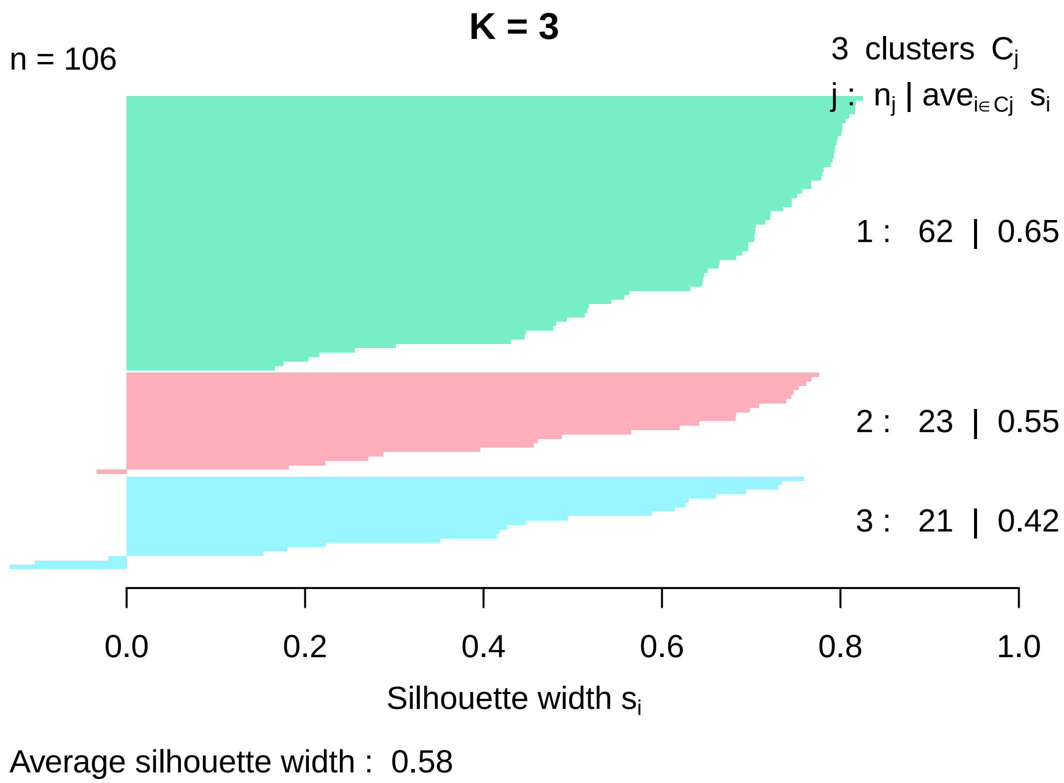
```r
set.seed(123)
pam3 <- pam(distance, 3 , stand = FALSE, diss = T)
pam4 <- pam(distance, 4 , stand = FALSE, diss = T)
```

3.3 Validating the Clusters

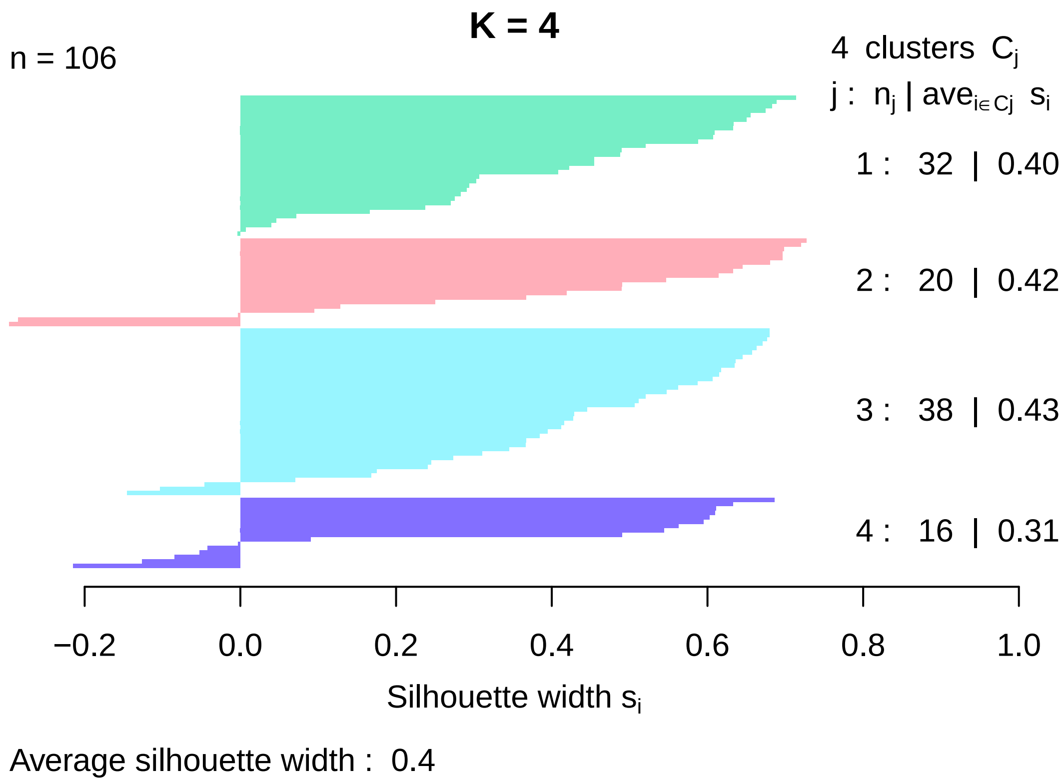
Silhouette method:

```r
plot(
  silhouette(pam3$cluster, distance),
  col = c('aquamarine2', 'lightpink1', 'cadetblue1'),
  border = NA,
  main = ''
```

```
)
title("K = 3", adj = 0.5, line = 1)
```




```
plot(
  silhouette(pam4$cluster, distance),
  col = c('aquamarine2', 'lightpink1', 'cadetblue1', 'slateblue1'),
  border = NA,
  main = ''
)
title("K = 4", adj = 0.5, line = 1)
```



3.4 Extracting the Clusters

With $k = 3$:

```
Dpam3 <- mutate(D, cluster = pam3$clustering)
rownames(Dpam3) <- rownames(D)
fnt3 <- function(i) {
  cluster <- rownames(Dpam3)[Dpam3$cluster == i]
  return(cluster)
}
sapply(1:3, FUN = fnt3)
```

```
## [[1]]
##  [1] "Albania"            "Argentina"          "Australia"
##  [4] "Austria"            "Bangladesh"         "Bolivia"
##  [7] "Botswana"           "Brazil"             "Bulgaria"
## [10] "Cambodia"           "Chile"              "China"
## [13] "Colombia"           "Costa Rica"         "Croatia"
## [16] "Cyprus"             "Czech Republic"     "Dominican Republic"
## [19] "Ecuador"            "El Salvador"        "Estonia"
## [22] "Georgia"            "Ghana"              "Greece"
## [25] "Honduras"           "Hungary"            "Indonesia"
## [28] "Israel"             "Italy"              "Jamaica"
## [31] "Japan"              "Kazakhstan"         "Kenya"
## [34] "Kyrgyz Republic"    "Lesotho"            "Lithuania"
## [37] "Luxembourg"         "Madagascar"         "Malawi"
## [40] "Malaysia"           "Malta"              "Mauritius"
```

```
## [43] "Mexico"             "Moldova"            "Mongolia"
## [46] "Panama"             "Paraguay"           "Peru"
## [49] "Poland"             "Portugal"           "Romania"
## [52] "Russian Federation" "Singapore"          "Slovak Republic"
## [55] "Sri Lanka"          "Tanzania"           "Thailand"
## [58] "Uganda"             "Ukraine"            "United States"
## [61] "Uruguay"            "Venezuela"
##
## [[2]]
##  [1] "Algeria"              "Bahrain"              "Burkina Faso"
##  [4] "Cameroon"             "Chad"                 "Egypt"
##  [7] "Ethiopia"             "Guatemala"            "India"
## [10] "Iran, Islamic Rep."   "Jordan"               "Korea, Rep."
## [13] "Kuwait"               "Mali"                 "Mauritania"
## [16] "Morocco"              "Nepal"                "Nigeria"
## [19] "Pakistan"             "Saudi Arabia"         "Turkey"
## [22] "United Arab Emirates" "Yemen"
##
## [[3]]
##  [1] "Belgium"        "Canada"         "Denmark"        "Finland"
##  [5] "France"         "Germany"        "Iceland"        "Ireland"
##  [9] "Latvia"         "Namibia"        "Netherlands"    "New Zealand"
## [13] "Nicaragua"      "Norway"         "Philippines"    "Slovenia"
## [17] "South Africa"   "Spain"          "Sweden"         "Switzerland"
## [21] "United Kingdom"
```

With $k = 4$:

```
Dpam4 <- mutate(D, cluster = pam4$clustering)
rownames(Dpam4) <- rownames(D)
fnt4 <- function(i) {
  cluster <- rownames(Dpam4)[Dpam4$cluster == i]
  return(cluster)
}
sapply(1:4, FUN = fnt4)
```

```
## [[1]]
##  [1] "Albania"            "Bangladesh"         "Brazil"
##  [4] "Cambodia"           "China"              "Cyprus"
##  [7] "Czech Republic"     "Dominican Republic" "El Salvador"
## [10] "Georgia"            "Ghana"              "Greece"
## [13] "Guatemala"          "Hungary"            "Indonesia"
## [16] "Italy"              "Japan"              "Kenya"
## [19] "Korea, Rep."        "Kuwait"             "Kyrgyz Republic"
## [22] "Madagascar"         "Malawi"             "Malaysia"
## [25] "Malta"              "Mauritius"          "Mexico"
## [28] "Paraguay"           "Peru"               "Singapore"
## [31] "Slovak Republic"    "Venezuela"
##
## [[2]]
##  [1] "Algeria"              "Bahrain"              "Burkina Faso"
##  [4] "Cameroon"             "Chad"                 "Egypt"
##  [7] "Ethiopia"             "India"                "Iran, Islamic Rep."
## [10] "Jordan"               "Mali"                 "Mauritania"
## [13] "Morocco"              "Nepal"                "Nigeria"
```

```
## [16] "Pakistan"             "Saudi Arabia"         "Turkey"
## [19] "United Arab Emirates" "Yemen"
##
## [[3]]
##  [1] "Argentina"          "Australia"          "Austria"
##  [4] "Belgium"            "Bolivia"            "Botswana"
##  [7] "Bulgaria"           "Canada"             "Chile"
## [10] "Colombia"           "Costa Rica"         "Croatia"
## [13] "Ecuador"            "Estonia"            "France"
## [16] "Honduras"           "Israel"             "Jamaica"
## [19] "Kazakhstan"         "Lesotho"            "Lithuania"
## [22] "Luxembourg"         "Moldova"            "Mongolia"
## [25] "Namibia"            "Panama"             "Poland"
## [28] "Portugal"           "Romania"            "Russian Federation"
## [31] "Slovenia"           "Sri Lanka"          "Tanzania"
## [34] "Thailand"           "Uganda"             "Ukraine"
## [37] "United States"      "Uruguay"
##
## [[4]]
##  [1] "Denmark"        "Finland"        "Germany"        "Iceland"
##  [5] "Ireland"        "Latvia"         "Netherlands"    "New Zealand"
##  [9] "Nicaragua"      "Norway"         "Philippines"    "South Africa"
## [13] "Spain"          "Sweden"         "Switzerland"    "United Kingdom"
```

3.5 Visualizing the Clusters

With $k = 3$:

```
Dpam3 <- cbind(country,Dpam3)
Dpam3_long <- Dpam3 %>%
  pivot_longer(
    cols = c(-country,-cluster),
    names_to = "year",
    values_to = "avh"
  ) %>%
  mutate(year = ymd(paste0(year, "-01-01")))
pam3$medoids
```

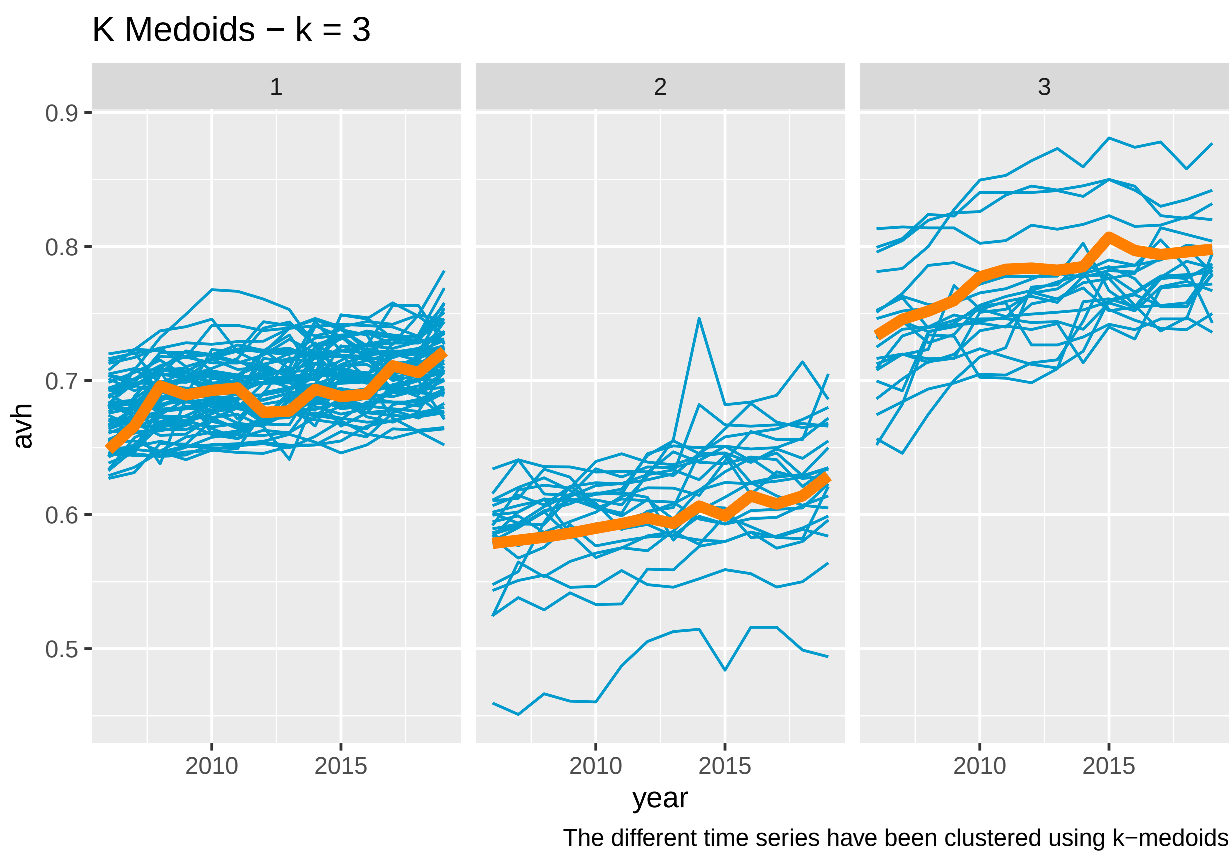
```
## [1] "Honduras" "Egypt"    "Ireland"
```

```
medoids <- Dpam3[c(39,28,45),] #extracting medoids from Dpam3 (we used grep function to
#identify position of rows)
medoids
```

```
##           country   2006   2007   2008   2009   2010   2011   2012   2013
## Honduras Honduras 0.6483 0.6661 0.6960 0.6893 0.6927 0.6945 0.6763 0.6773
## Egypt       Egypt 0.5786 0.5809 0.5832 0.5862 0.5899 0.5933 0.5975 0.5935
## Ireland   Ireland 0.7335 0.7457 0.7518 0.7597 0.7773 0.7830 0.7839 0.7823
##            2014  2015  2016  2017  2018  2019 cluster
## Honduras 0.6935 0.688 0.690 0.711 0.706 0.722       1
## Egypt    0.6064 0.599 0.614 0.608 0.614 0.629       2
## Ireland  0.7850 0.807 0.797 0.794 0.796 0.798       3
```

```
medoids_long <- medoids %>%
  pivot_longer(cols = c(-cluster,-country),
               names_to = "year",
```

```r
                values_to = "avh") %>%
  mutate(year = ymd(paste0(year, "-01-01")))

ggplot() +
  geom_line(data = Dpam3_long,
            aes(y = avh, x = year, group = country),
            colour = "deepskyblue3") +
  facet_wrap( ~ cluster, nrow = 1) +
  geom_line(
    data = medoids_long,
    aes(y = avh, x = year, group = cluster),
    col = "darkorange1",
    size = 2
  )  +
  labs(title = "K Medoids - k = 3",
       caption = "The different time series have been clustered using k-medoids")
```




```r
Dpam4 <- cbind(country,Dpam4)
Dpam4_long <- Dpam4 %>%
  pivot_longer(
    cols = c(-country,-cluster),
    names_to = "year",
    values_to = "avh"
  ) %>%
  mutate(year = ymd(paste0(year, "-01-01")))
pam4$medoids
```

```
## [1] "Malta"       "Egypt"       "Panama"      "New Zealand"
```

```
medoids2 <- Dpam4[c(64,28,79,74),] #extracting medoids from Dpam4 (we used grep function
#to identify position of rows)
medoids2
```

```
##                 country   2006   2007   2008   2009   2010   2011   2012   2013
## Malta             Malta 0.6518 0.6615 0.6634 0.6635 0.6695 0.6658 0.6666 0.6761
## Egypt             Egypt 0.5786 0.5809 0.5832 0.5862 0.5899 0.5933 0.5975 0.5935
## Panama           Panama 0.6935 0.6954 0.7095 0.7024 0.7072 0.7042 0.7122 0.7164
## New Zealand New Zealand 0.7509 0.7649 0.7859 0.7880 0.7808 0.7810 0.7805 0.7799
##               2014  2015  2016  2017  2018  2019 cluster
## Malta       0.6707 0.668 0.664 0.682 0.686 0.693       1
## Egypt       0.6064 0.599 0.614 0.608 0.614 0.629       2
## Panama      0.7195 0.722 0.721 0.722 0.722 0.730       3
## New Zealand 0.7772 0.782 0.781 0.791 0.801 0.799       4
```

```
medoids2_long <- medoids2 %>%
  pivot_longer(cols = c(-cluster,-country),
               names_to = "year",
               values_to = "avh") %>%
  mutate(year = ymd(paste0(year, "-01-01")))

ggplot() +
  geom_line(data = Dpam4_long,
            aes(y = avh, x = year, group = country),
            colour = "deepskyblue3") +
  facet_wrap( ~ cluster, nrow = 1) +
  geom_line(
    data = medoids2_long,
    aes(y = avh, x = year, group = cluster),
    col = "darkorange1",
    size = 2
  )  +
  labs(title = "K Medoids - k = 4",
      caption = "The different time series have been clustered using k-medoids")
```

The different time series have been clustered using k−medoids

4. Cluster Validity Assessment

Using the **dtwclust** package, we compute the clusters again in order to apply the **cvi** function that returns necessary indexes to compare methods.

The indexes are:

- *Sil*: Silhouette index to be maximized
- *D*: Dunn index to be maximized
- *COP*: COP index to be minimized
- *DB*: Davies-Bouldin index to be minimized
- *DB*: Modified Davies-Bouldin index to be minimized
- *CH*: Calinski-Harabasz index to be maximized
- *SF*: Score Function to be maximized

```
set.seed(123)
clust.hier3 <-
  tsclust(D,
          type = "hierarchical",
          k = 3,
          distance = "dtw") # Hierarchical - k = 3
clust.hier4 <-
  tsclust(D,
          type = "hierarchical",
          k = 4,
          distance = "dtw") # Hierarchical - k = 4
```

```r
clust.pam3 <-
  tsclust(
    D,
    type = "partitional",
    k = 3,
    distance = "dtw",
    centroid = "pam"
  ) # K-Medoids - k = 3
clust.pam4 <-
  tsclust(
    D,
    type = "partitional",
    k = 4,
    distance = "dtw",
    centroid = "pam"
  ) # K-Medoids - k = 4
```

4.1 Hierarchical k = 3 X k = 4

```r
cvi(clust.hier3)
```

```
##         Sil          SF          CH          DB      DBstar           D
##  0.46865517  0.24889812 32.51667640  0.47219043  0.50214443  0.03612254
##         COP
##  0.12654323
```

```r
cvi(clust.hier4)
```

```
##         Sil          SF          CH          DB      DBstar           D
##  0.56735157  0.23815470 59.60078794  0.64333959  0.73268366  0.04542623
##         COP
##  0.06720333
```

4.2 K-Medoids k = 3 X k = 4

```r
cvi(clust.pam3)
```

```
##         Sil          SF          CH          DB      DBstar           D
##  0.42460761  0.35359684 49.73494097  0.61407999  0.66457512  0.01499604
##         COP
##  0.12065116
```

```r
cvi(clust.pam4)
```

```
##         Sil          SF          CH          DB      DBstar           D
##  0.43221001  0.30977898 83.23514704  0.87582544  1.11646029  0.02284842
##         COP
##  0.06068568
```

4.3 Hierarchical X K-Medoids, k = 3

```r
cvi(clust.hier3)
```

```
##         Sil          SF          CH          DB      DBstar           D
##  0.46865517  0.24889812 32.51667640  0.47219043  0.50214443  0.03612254
```

```
##          COP
##   0.12654323
```

```r
cvi(clust.pam3)
```

```
##          Sil          SF          CH          DB      DBstar           D
##   0.42460761  0.35359684 49.73494097  0.61407999  0.66457512  0.01499604
##          COP
##   0.12065116
```

4.4 Hierarchical X K-Medoids, k = 4

```r
cvi(clust.hier4)
```

```
##          Sil          SF          CH          DB      DBstar           D
##   0.56735157  0.23815470 59.60078794  0.64333959  0.73268366  0.04542623
##          COP
##   0.06720333
```

```r
cvi(clust.pam4)
```

```
##          Sil          SF          CH          DB      DBstar           D
##   0.43221001  0.30977898 83.23514704  0.87582544  1.11646029  0.02284842
##          COP
##   0.06068568
```

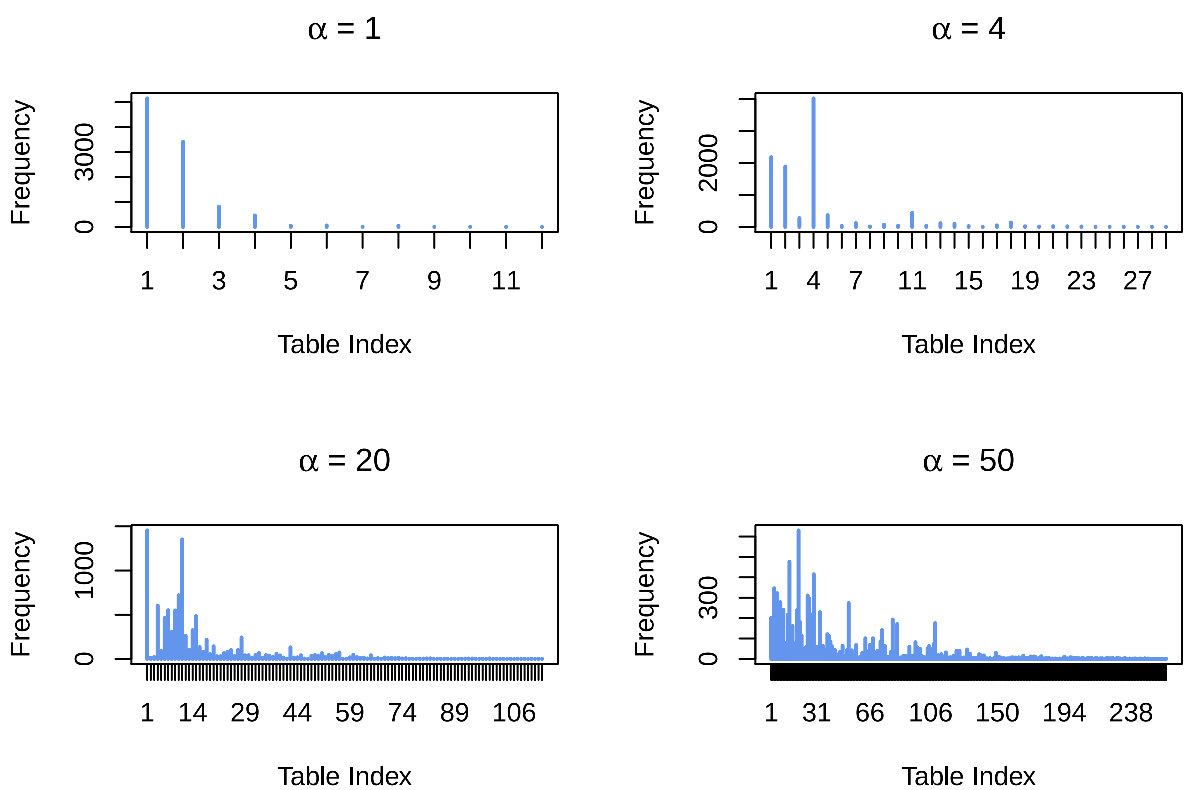
5. Non parametric clustering

5.1 CRP generator

We report the code used to produced the plots in the section about CRP.

```r
crp = function(num.customers, alpha) {
  table = c(1)
  next.table <- 2
  for (i in 1:num.customers) {
    if (runif(1, 0, 1) < alpha / (alpha + i)) {
      # the customers sits in a new table
      table <- c(table, next.table)
      next.table <- next.table + 1
    } else {
      # the customer sits in a table already occupied
      select.table <- table[sample(1:length(table), 1)]
      table <- c(table, select.table)
    }
  }
  table
}
# Plot the random partition of 10.000 customers for different values # of the concentration parameter
par(mfrow = c(2, 2))
plot(
  table(crp(10000, 1))
  ,
  xlab = "Table Index",
  ylab = "Frequency",
  col = 'cornflowerblue',
```

```
  main = expression(paste(alpha,' = 1'))
)
plot(
  table(crp(10000, 4))
  ,
  xlab = "Table Index",
  ylab = "Frequency",
  col = 'cornflowerblue',
  main = expression(paste(alpha,' = 4'))
)
plot(
  table(crp(10000, 20))
  ,
  xlab = "Table Index",
  ylab = "Frequency",
  col = 'cornflowerblue',
  main = expression(paste(alpha,' = 20'))
)
plot(
  table(crp(10000, 50))
  ,
  xlab = "Table Index",
  ylab = "Frequency",
  col = 'cornflowerblue',
  main = expression(paste(alpha,' = 50'))
)
```



5.2 BNPTSclust: Clustering algorithm

We report the code of the output of interests contained in the slides. Since the algorithm is computationally intensive (some cases took a few hours to be run completely) and the complete ouput printed is very long, we report here only the codes we used for the case of interest commented in the slides. The relevant outputs of all the algorithm runs contained in the table of the results can be found in the pdf file called *Algorithm_runs_outputs.pdf*.

It is assumed that the periods of the series appear as the row names of the file; for this reason, we need to use the transpose of the original data set (that is, *GGIts.csv*). We actually transposed our original dataset in excel, but we could have done it in R as follows, manipulating the one used above:

```
E <- t(D)
```

5.2.1. Importing and manipulating the data

```
S <- read.csv2('GGIts.csv',row.names = 1, head = T)
S <- S[ , ! apply( S , 2 , function(x) any(is.na(x)) ) ] # eliminating NA's
dim(S) # 14 years, 106 countries
```

```
## [1]  14 106
```

```
S <- S[14:1,] # sorting the years
S[1,] # first row of the transposed data set
```

```
##      Albania Algeria Argentina Australia Austria Bahrain Bangladesh Belgium
## 2006  0.6607  0.6018    0.6829    0.7163  0.6986  0.5894      0.627  0.7078
##      Bolivia Botswana Brazil Bulgaria Burkina.Faso Cambodia Cameroon Canada
## 2006  0.6335   0.6897 0.6543    0.687       0.5854   0.6291   0.5865 0.7165
##        Chad  Chile  China Colombia Costa.Rica Croatia Cyprus Czech.Republic
## 2006 0.5247 0.6455 0.6561   0.7049     0.6936  0.7145  0.643         0.6712
##      Denmark Dominican.Republic Ecuador  Egypt El.Salvador Estonia Ethiopia
## 2006  0.7462             0.6639  0.6433 0.5786      0.6837  0.6944   0.5946
##      Finland France Georgia Germany  Ghana Greece Guatemala Honduras Hungary
## 2006  0.7958  0.652    0.67  0.7524 0.6653  0.654    0.6067   0.6483  0.6698
##      Iceland  India Indonesia Iran..Islamic.Rep. Ireland Israel  Italy Jamaica
## 2006  0.7813 0.6011    0.6541             0.5803  0.7335 0.6889 0.6456  0.7014
##       Japan Jordan Kazakhstan  Kenya Korea..Rep. Kuwait Kyrgyz.Republic Latvia
## 2006 0.6447 0.6109     0.6928 0.6486      0.6157 0.6341          0.6742 0.7091
##      Lesotho Lithuania Luxembourg Madagascar Malawi Malaysia   Mali  Malta
## 2006  0.6807    0.7077     0.6671     0.6385 0.6437   0.6509 0.5996 0.6518
##      Mauritania Mauritius Mexico Moldova Mongolia Morocco Namibia  Nepal
## 2006     0.5835    0.6328 0.6462  0.7128   0.6821  0.5827  0.6864 0.5478
##      Netherlands New.Zealand Nicaragua Nigeria Norway Pakistan Panama Paraguay
## 2006       0.725      0.7509    0.6566  0.6104 0.7994   0.5434 0.6935   0.6556
##        Peru Philippines Poland Portugal Romania Russian.Federation Saudi.Arabia
## 2006 0.6619      0.7516 0.6802   0.6922  0.6797              0.677       0.5242
##      Singapore Slovak.Republic Slovenia South.Africa  Spain Sri.Lanka Sweden
## 2006     0.655          0.6757   0.6745       0.7125 0.7319    0.7199 0.8133
##      Switzerland Tanzania Thailand Turkey Uganda Ukraine United.Arab.Emirates
## 2006      0.6997   0.7038   0.6831  0.585 0.6797  0.6797               0.5919
##      United.Kingdom United.States Uruguay Venezuela  Yemen
## 2006         0.7365        0.7042  0.6549    0.6664 0.4595
```

5.2.2. Case 1 of interest

Nstable model, assuming a quadratic trend and the level of the series as criteria for clustering; $c_0 = c_1 = 0.001$. Produces 5 clusters.

```
set.seed(123)
# we are using the function for annual data
(tseriesca.out1 <- tseriesca(
    S, # data set
    maxiter = 10000, # number of iterations
    burnin = 1000, # burn in
    thinning = 5, # thinning
    level = T, # consider the level as cluster criteria
    trend = T, # consider the trend as cluster criteria
    deg = 2, # consider a quadratic trend
    scale = F, # our data are already expressed in the same unit of measure
    # variance distributions parameters
    c0eps = 0.001,
    c1eps = 0.001,
    c0beta = 0.001,
    c1beta = 0.001,
    c0alpha = 0.001,
    c1alpha = 0.001,
    # hyper prior on the parameter a of the PD
    priora = T, # we fix a prior over a
    pia = 0.5, # suggested value for pi
    q0a = 1, # suggested value for q0a
    q1a = 1, # suggested value for q1a
    # the Nstable process is the special case of PD where b = 0
    priorb = F, # we do not fix a prior over b
    b = 0, # we fix the value
    indlpml = T # we want the output to contain the LPML
  ))
```

Plotting the clusters:

```
clusterplots(tseriesca.out1, S)
```

Obtaining the diagnostic plots

```
diagplots(tseriesca.out1)
```

5.2.3. Case 2 of interest

Dirichlet model assuming only a linear trend as clustering criteria (excluding the level); $c_0 = c_1 = 0.001$. Produces 11 clusters.

```
set.seed(123)
(tseriesca.out2 <- tseriesca(
    S, # data set
    maxiter = 10000, # number of iterations
    burnin = 1000, # burn in
    thinning = 5, # thinning
    level = F, # don't consider the level as cluster criteria
    trend = T, # consider the trend as cluster criteria
    deg = 1, # consider a linear trend
    scale = F, # our data are already expressed in the same unit of measure
```

```r
    # variance distributions parameters
    c0eps = 0.001,
    c1eps = 0.001,
    c0beta = 0.001,
    c1beta = 0.001,
    c0alpha = 0.001,
    c1alpha = 0.001,
    # the Dirichlet process is a special case of PD when a=0
    priora = F, # we do not fix a prior over a
    a = 0, # we fix the value of a
    # hyper prior on the parameter b of the PD
    priorb = T, # we fix a prior over b
    q0b = 1, # suggested value for q0b
    q1b = 1, # suggested value for q1b
    b = 0.01, # for the algorithm to work, b needs to be greater
    # than a (since we set a prior over b, this argument is
    # interpreted as a starting point)
    indlpml = T # we want the output to contain the LPML
  ))
```

Plotting the clusters:

```r
clusterplots(tseriesca.out2, S)
```

Obtaining the diagnostic plots:

```r
diagplots(tseriesca.out2)
```

5.2.4. Case 3 of interest

Poisson Dirichlet process assuming only the level as clustering criteria; $c_0 = c_1 = 0.001$. Produces 53 clusters.

```r
#set.seed(123)
(tseriesca.out3 <- tseriesca(
    S, # data set
    maxiter = 10000, # number of iterations
    burnin = 1000, # burn in
    thinning = 5, # thinning
    level = T, # consider the level as cluster criteria
    trend = F, # don't consider the trend as cluster criteria
    scale = F, # our data are already expressed in the same unit of measure
    # variance distributions parameters
    c0eps = 0.001,
    c1eps = 0.001,
    c0beta = 0.001,
    c1beta = 0.001,
    c0alpha = 0.001,
    c1alpha = 0.001,
    # hyper prior on the parameter a of the PD
    priora = T, # we fix a prior over a
    pia = 0.5, # suggested value for pi
    q0a = 1, # suggested value for q0a
    q1a = 1, # suggested value for q1a
    # hyper prior on the parameter b of the PD
    priorb = T, # we fix a prior over b
    q0b = 1, # suggested value for q0b
```

```
    q1b = 1, # suggested value for q1b
    indlpml = T # we want the output to contain the LPML
  ))
```

Plotting the clusters:

```
clusterplots(tseriesca.out3, S)
```

Obtaining the diagnostic plots:

```
diagplots(tseriesca.out3)
```
