# R Instructions for the 6th seminar

A dataset Countries2.RData contains 5 variables for 30 countries. Using cluster analysis, discover which countries are similar. Use a standardized dataset.

## Step 0: Prepare your working environment

getwd() # Where are you now?

setwd("…") # Change your working directory

rm(list=ls()) # Remove all objects from previous analyses

## R Instructions for the problem 1

Standardize the data:

load("Countries2.RData")

Countries2std <- as.data.frame(scale(Countries2, center = T, scale = T))

## R Instructions for the problem 2

Get Euclidean distances:

euclid.dist <- dist(Countries2std, method = "euclidean")

# dist(Countries2std, method = "euclidean", **upper = T**) if you want to get also the upper triangle to be printed

e.g. distance between Spain and the Czech Republic= 2.2554669 means that a cluster including Spain and a cluster including the Czech Republic were merged, when the cluster distance was 2.255.

## R Instructions for the problem 3

Create dendrograms using various methods:

For various methods, choose from: **ward**, **single**, **complete** (this one is set to be default), **average**, **mcquitty**, **median**, **centroid**

countries.cluster <- hclust(euclid.dist, method = "complete")

Draw the dendrogram

plot(countries.cluster)

To see differences between various methods:

m <- c("ward", "single", "complete", "average", "mcquitty", "median", "centroid")

par(mfrow = c(2, 4)) #creates 8 panels in a graph’s plot

for (i in m) {

clust <- hclust(euclid.dist, method = i)

plot(clust)

}

## R Instructions for the problem 4

Compute the cophenetic coefficients:

# Remeber: euclid.dist is a matrix of distances between particular pairs of # countries

coph.coef <- c()

for (i in m) { # Remeber: m is a list of clustering methods

clust <- hclust(euclid.dist, method = i)

coph <- cophenetic(clust)

coph.coef <- c(coph.coef, cor(euclid.dist, coph))

}

coph.table <- data.frame(m, coph.coef)

## R Instructions for the problem 5

At which distance should you stop the clustering?

par(mfrow = c(1, 1))

clust <- hclust(euclid.dist, method = "complete") # Change the method here

s <- c(1:length(clust$height)) # List of steps

h <- sort(clust$height)

# Sort heights (some methods are designed to provide ordered heights, some # not. Check it by clust$height==h

plot(h ~ s, type = "s", xlab = "Order number", ylab = "Height")

You should stop the clustering when the line drops down too steeply. (On y-axes there are distances between clusters when merged. On x-axes arenthe order numbers which are equal to “step numbers” only for some methods (single, complete, average,ward,…). Otherwise you have to find out which step is related to that distance at which you want stop clustering.

## R Instructions for the problem 6

1. Determine the number of clusters. You may use either the results of agglomerative methods above, or use following script:

wssplot <- function(data, nc=15, seed=1234){

wss <- (nrow(data)-1)\*sum(apply(data,2,var))

for (i in 2:nc){

set.seed(seed)

wss[i] <- sum(kmeans(data, centers=i)$withinss)}

plot(1:nc, wss, type="b", xlab="Number of Clusters",

ylab="Within groups sum of squares")}

wssplot(Countries2std)

# As the number of clusters is increasing, the withingroup sum of squares

# decreases

1. Initialize the centers and number of iterations. In default, you don’t need to set up anything: R chooses *n* random rows (where n = numer of centers) as centers and 10 as a number of max. iterations.
2. Run the analysis:

set.seed(1234)

kclust = kmeans(Countries2std, centers = 4, iter.max = 25)

1. Find out in which cluster is particular country.

kclust$cluster

1. For each cluster find the vector of means of all considered variables.

kclust$centers

1. Plot a scatterplot for two observed variables and differentiate the points by thein cluster:

# For all variables in one matrix plot

plot(Countries2std, col = kclust$cluster)

points(kclust$centers, pch = 22)

# For selected variables

plot(Countries2std$Fertility ~ Countries2std$Age1Birth, col = kclust$cluster)

points(kclust$centers, pch = 22)