Exercise 7

Riccardo Petrella

2023-11-29

These are the packages to be installed and loaded in order to run the code.

```
library(fpc)
library(smacof)
library(pdfCluster)
library(cluster)
library(reshape2)
library(prabclus)
library(teigen)
library(psych)
library(dbscan)
library(clusterSim)
library(mixsmsn)
library(DescTools)
library(flexmix)
library(stringdist)
library(nomclust)
library(poLCA)
library(fda)
library(dplyr)
```

1

First the dataset from the poLCA package is uploaded. Then we pre-process the data as shown in the task and we finally compute the simple matching distance and the MDS.

```
data(election)
election12 <- election[,1:12]
electionwithna <- election12

# labeling missing values with "NA" to become another category
for (i in 1:12){
levels(electionwithna[,i]) <- c(levels(election12[,i]),"NA")
electionwithna[is.na(election12[,i]),i] <- "NA"
}

# simple matching distance
sm_elec <- sm(electionwithna)
```

```
# multi-dimensional scaling
mds_elec <- mds(sm_elec)
```

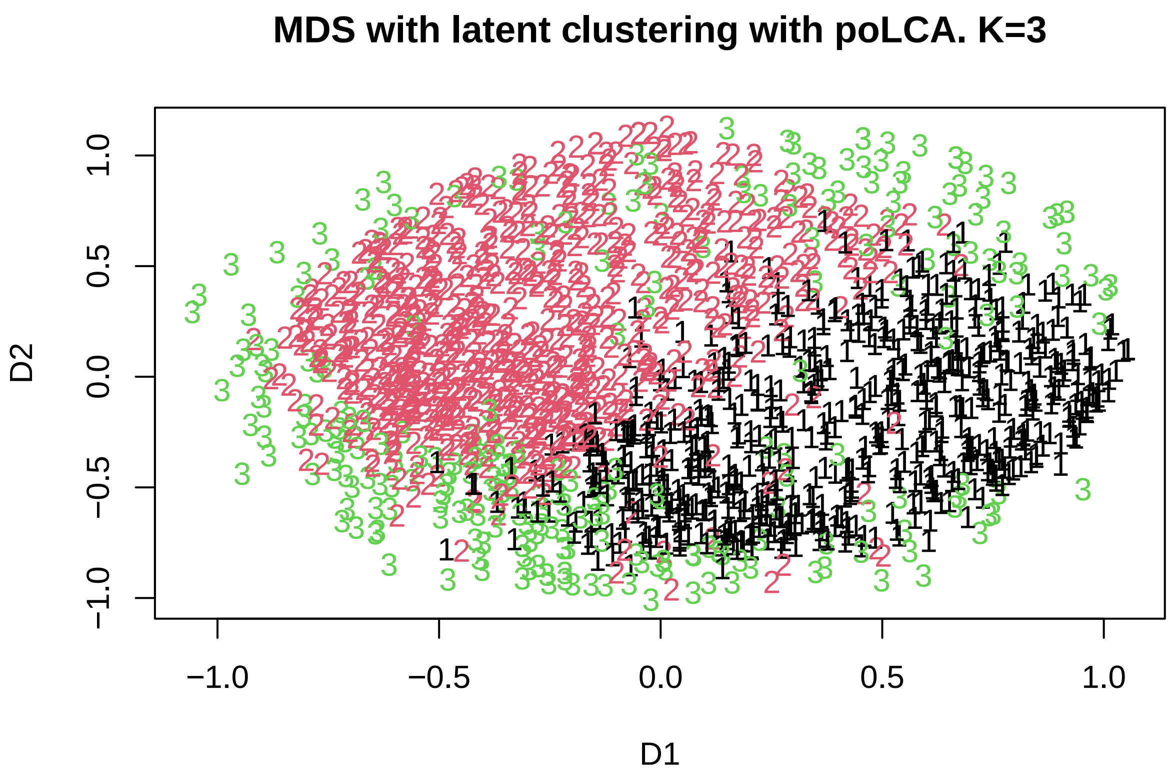
(a) Compute a latent class clustering with 3 clusters using poLCA (read its help page and decide about potentially useful parameter settings)

```
formula <- cbind(MORALG,CARESG,KNOWG,LEADG,DISHONG,INTELG,
             MORALB,CARESB,KNOWB,LEADB,DISHONB,INTELB)~1

# k=3
set.seed(080985)
elec_poLCA3 <- poLCA(formula = formula, electionwithna,
      nclass = 3, na.rm = F,  nrep = 5)
```

```
## Model 1: llik = -25990.17 ... best llik = -25990.17
## Model 2: llik = -25885.25 ... best llik = -25885.25
## Model 3: llik = -25885.25 ... best llik = -25885.25
## Model 4: llik = -25885.25 ... best llik = -25885.25
## Model 5: llik = -25891.5 ... best llik = -25885.25
## Conditional item response (column) probabilities,
##  by outcome variable, for each class (row)
##
## $MORALG
##            1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:             0.4680       0.4895          0.0286            0.0060 0.0079
## class 2:             0.1069       0.4510          0.2755            0.1321 0.0345
## class 3:             0.0576       0.4082          0.1332            0.0660 0.3351
##
## $CARESG
##            1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:             0.3371       0.5429          0.0867            0.0252 0.0081
## class 2:             0.0333       0.3201          0.4160            0.2051 0.0255
## class 3:             0.0769       0.2832          0.2050            0.1594 0.2756
##
## $KNOWG
##            1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:             0.4990       0.4824          0.0131            0.0055 0.0000
## class 2:             0.1194       0.6230          0.1978            0.0496 0.0102
## class 3:             0.0818       0.5479          0.1376            0.0513 0.1814
##
## $LEADG
##            1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:             0.3298       0.5560          0.1017            0.0049 0.0076
## class 2:             0.0282       0.3112          0.4574            0.1851 0.0179
## class 3:             0.0394       0.3353          0.2541            0.0966 0.2746
##
## $DISHONG
##            1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:             0.0247       0.0634          0.3563            0.5322 0.0235
## class 2:             0.1248       0.2687          0.3919            0.1771 0.0375
## class 3:             0.0412       0.1604          0.2142            0.1767 0.4075
```

```
##
## $INTELG
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.5060       0.4691         0.0188            0.0061 0.0000
## class 2:           0.1511       0.6238         0.1649            0.0509 0.0093
## class 3:           0.0876       0.5720         0.1144            0.0683 0.1577
##
## $MORALB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.0936       0.4307         0.3185            0.0937 0.0635
## class 2:           0.3151       0.5678         0.0997            0.0042 0.0132
## class 3:           0.0389       0.2634         0.1132            0.1183 0.4661
##
## $CARESB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.0162       0.1375         0.4631            0.3687 0.0144
## class 2:           0.1625       0.5842         0.2230            0.0213 0.0090
## class 3:           0.0253       0.1451         0.2268            0.2826 0.3203
##
## $KNOWB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.0740       0.3821         0.3875            0.1457 0.0107
## class 2:           0.2390       0.6724         0.0856            0.0009 0.0021
## class 3:           0.0836       0.4037         0.1669            0.1275 0.2183
##
## $LEADB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.0350       0.3111         0.4425            0.1792 0.0323
## class 2:           0.2759       0.6415         0.0703            0.0041 0.0082
## class 3:           0.0347       0.3396         0.1811            0.1566 0.2881
##
## $DISHONB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.0623       0.2487         0.4161            0.1935 0.0795
## class 2:           0.0180       0.0933         0.3758            0.4861 0.0269
## class 3:           0.0478       0.1546         0.2031            0.1219 0.4725
##
## $INTELB
##           1 Extremely well 2 Quite well 3 Not too well 4 Not well at all     NA
## class 1:           0.1144       0.4285         0.3229            0.1227 0.0116
## class 2:           0.2791       0.6565         0.0637            0.0007 0.0000
## class 3:           0.0591       0.4647         0.1266            0.1001 0.2495
##
## Estimated class population shares
##  0.3803 0.4735 0.1462
##
## Predicted class memberships (by modal posterior prob.)
##  0.3815 0.4756 0.1429
##
## =========================================================
## Fit for 3 latent classes:
## =========================================================
## number of observations: 1785
## number of estimated parameters: 146
## residual degrees of freedom: 1639
## maximum log-likelihood: -25885.25
##
## AIC(3): 52062.51
## BIC(3): 52863.64
## G^2(3): 25461.92 (Likelihood ratio/deviance statistic)
## X^2(3): 12552843829 (Chi-square goodness of fit)
##
```

```r
plot(mds_elec$conf,col=elec_poLCA3$predclass,
pch=clusym[elec_poLCA3$predclass])
title(main = "MDS with latent clustering with poLCA. K=3")
```



We create a list of the 12 variables of the dataset with the expression "~1" as the formula to use in poLCA (~1 means that the formula does not have covariates, since the variable "PARTY" is a potential covariate but has been removed in the data pre-processing). We have decided to use only one extra parameter shown in the example in the Help windows (i.e. nrep=5), while nclass=3 is required in the task and na.rm = F because we have already turned NA's into a new category.

From the scatterplot of the MDS, we can clearly see that the clusters do have well defined shapes and patterns, with some outliers. Cluster 1 and cluster 2 are elliptical and compact, while cluster 3 looks like a hollow circumference around the first two.

Remark: in poLCA, the lower the BIC the better the model.

(b) Compute a latent class clustering with 3 clusters using flexmixedruns.

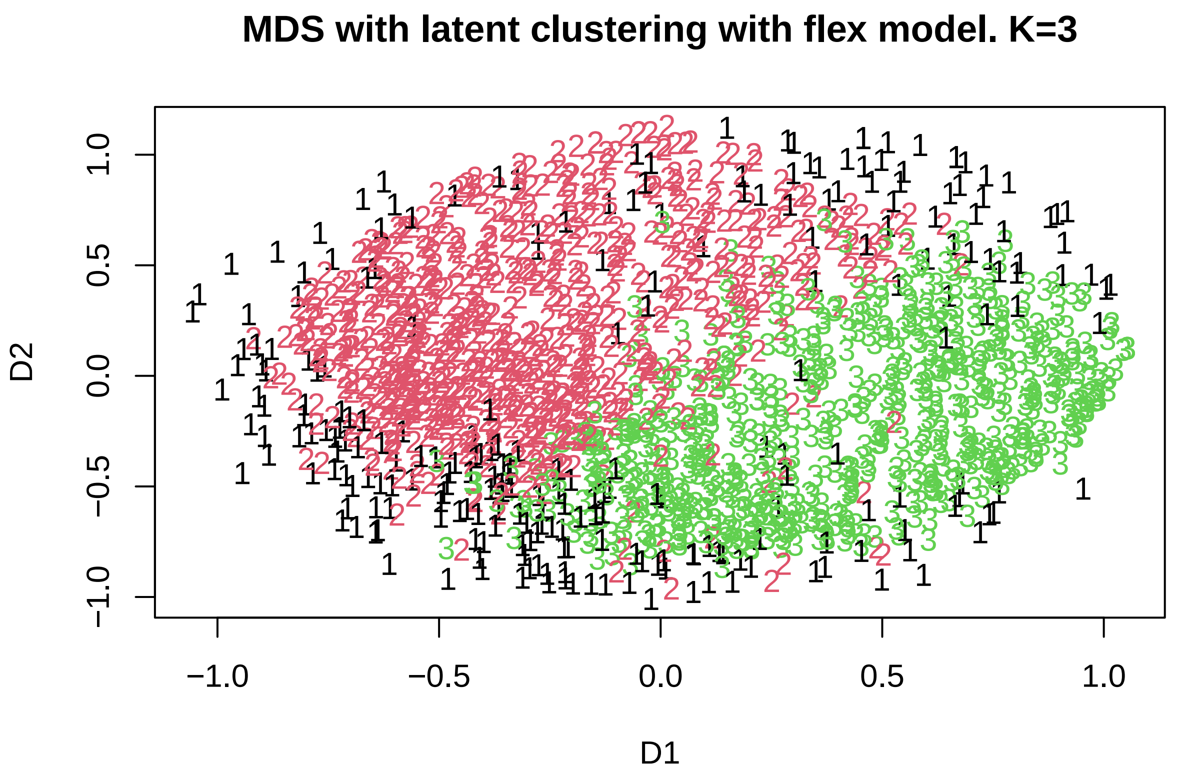
```
set.seed(887766)
flex_elec <-
flexmixedruns(electionwithna,continuous=0,discrete=12,n.cluster=3)
```

```
## k=  3  new best fit found in run  1
## Nonoptimal or repeated fit found in run  2
## Nonoptimal or repeated fit found in run  3
## k=  3  new best fit found in run  4
## Nonoptimal or repeated fit found in run  5
## Nonoptimal or repeated fit found in run  6
## Nonoptimal or repeated fit found in run  7
## k=  3  new best fit found in run  8
## Nonoptimal or repeated fit found in run  9
## k=  3  new best fit found in run  10
## Nonoptimal or repeated fit found in run  11
## Nonoptimal or repeated fit found in run  12
## Nonoptimal or repeated fit found in run  13
## Nonoptimal or repeated fit found in run  14
## Nonoptimal or repeated fit found in run  15
## Nonoptimal or repeated fit found in run  16
## Nonoptimal or repeated fit found in run  17
## Nonoptimal or repeated fit found in run  18
## Nonoptimal or repeated fit found in run  19
## Nonoptimal or repeated fit found in run  20
## k=  3  BIC=  52863.75
```

```
flex_elec$flexout[[3]]
```

```
##
## Call:
## flexmix(formula = x ~ 1, k = k, cluster = initial.cluster, model = lcmixed(continuous = continuous,
##     discrete = discrete, ppdim = ppdim, diagonal = diagonal),
##     control = control)
##
## Cluster sizes:
##   1   2   3
## 255 848 682
##
## convergence after 27 iterations
```

```
plot(mds_elec$conf,col=flex_elec$flexout[[3]]@cluster,
pch=clusym[flex_elec$flexout[[3]]@cluster])
title(main = "MDS with latent clustering with flex model. K=3")
```

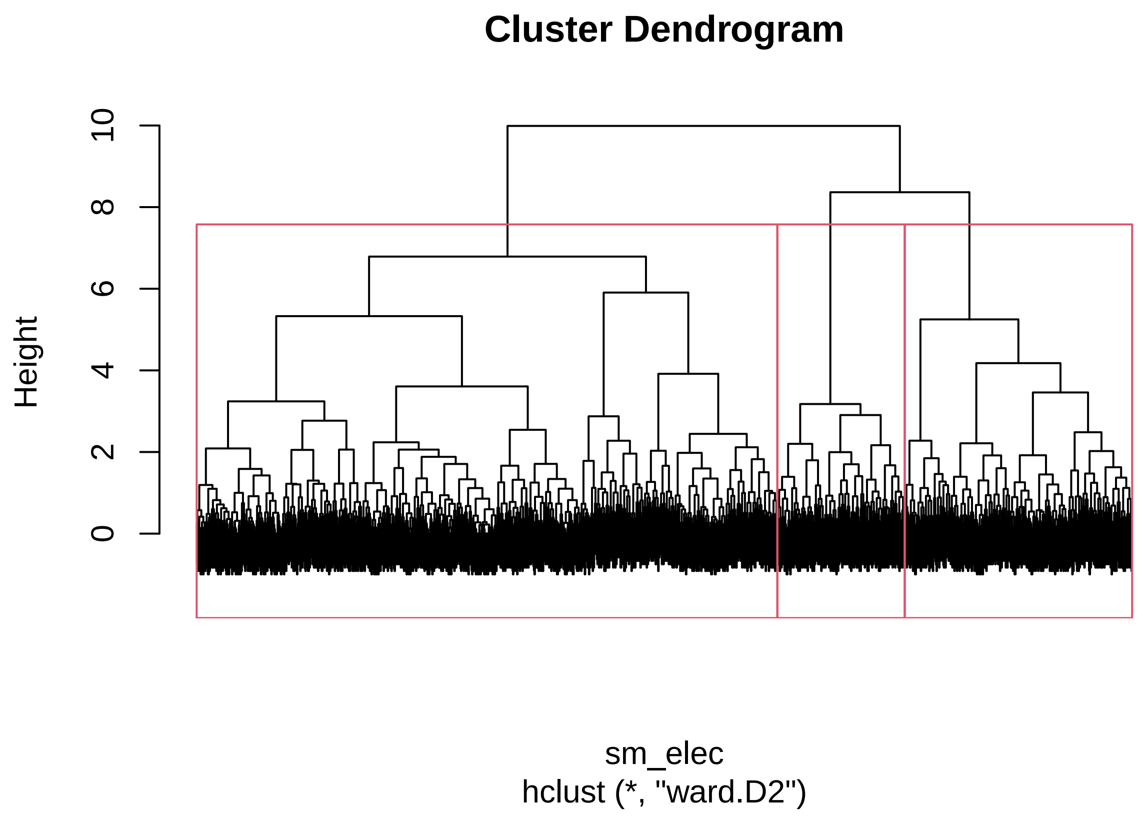


Visually, the clustering accomplished with very similar to the former, tough the numbers of clusters are inverted.

(c) Compute a distance-based clustering of your choice with 3 clusters based on the simple matching distance (you can also choose here between computing the simple matching distance on electionwithna or on election12, which will compute the distance just taking variables into account that are non-missing on both observations, using daisy as explained earlier in class - actually in general then it's a dissimilarity and not a distance as missing values can spoil the triangle inequality).

We have decided to perform hierarchical clustering with Ward's method, based on the simple matching distance and applied on the "electionwithna" dataset.

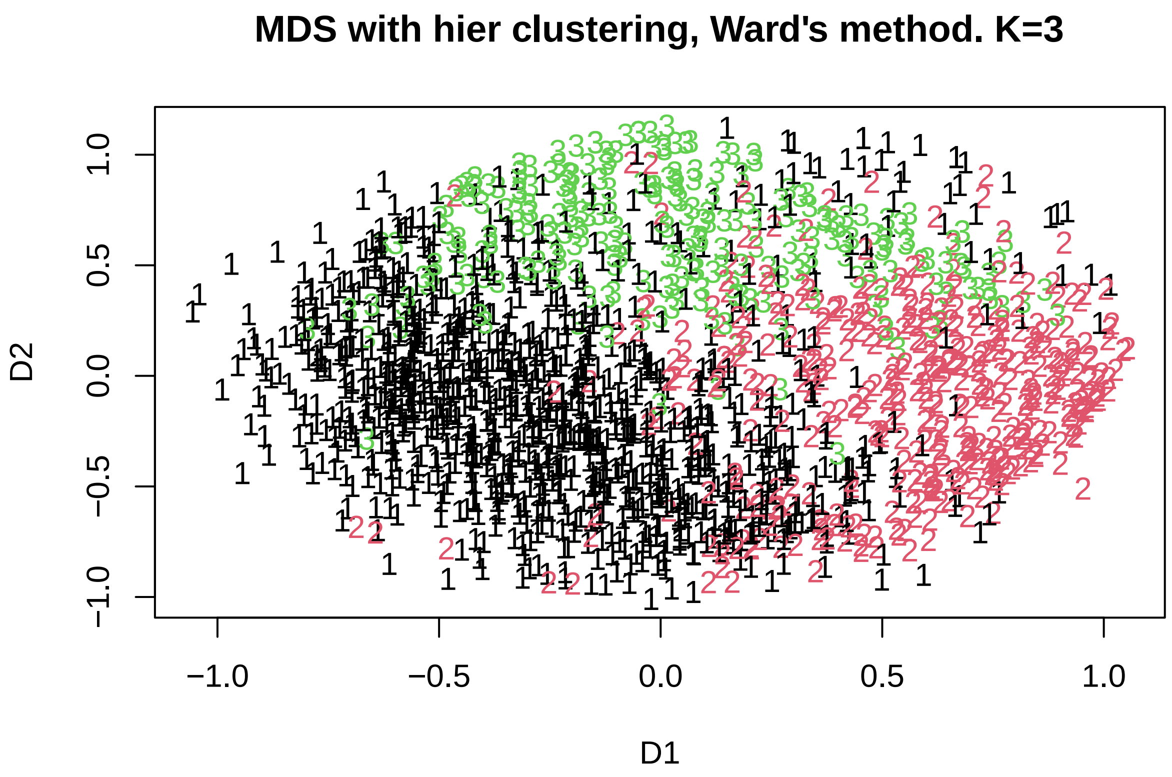
```r
hier_elec <- hclust(sm_elec,method="ward.D2")

# Plot the dendrogram:

plot(hier_elec, labels = FALSE)

rect.hclust(hier_elec, k = 3)
```



```
hclust_elec <- cutree(hier_elec, k=3)
table(hclust_elec)
```

```
## hclust_elec
##    1    2    3
## 1108  434  243
```

```
plot(mds_elec$conf,col=hclust_elec,
pch=clusym[hclust_elec])
title(main = "MDS with hier clustering, Ward's method. K=3")
```



This time the clusters look pretty diffent with respect to the first two clusterings. In facts, despite the clusters are still very intuitive to see and all three of them are elliptical and quite homogeneous, there is no cluster shaped as a hollow circumference around the others like before.

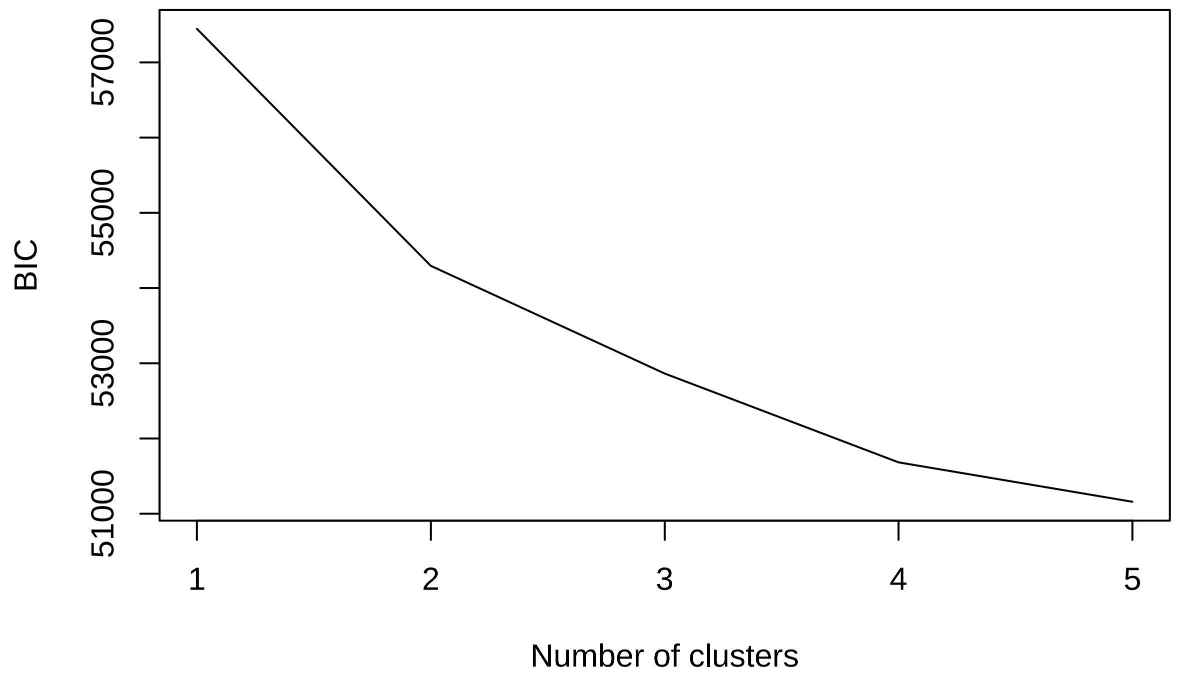
(d) Compute a latent class clustering using flexmixedruns with estimated number of clusters. (poLCA will not estimate the number of cluster automatically, although it gives out the BIC, so this could in principle be implemented easily.)

We now perform clustering with flexmixedruns but this time we want to estimate the number of clusters (from 1 to 5) with BIC.

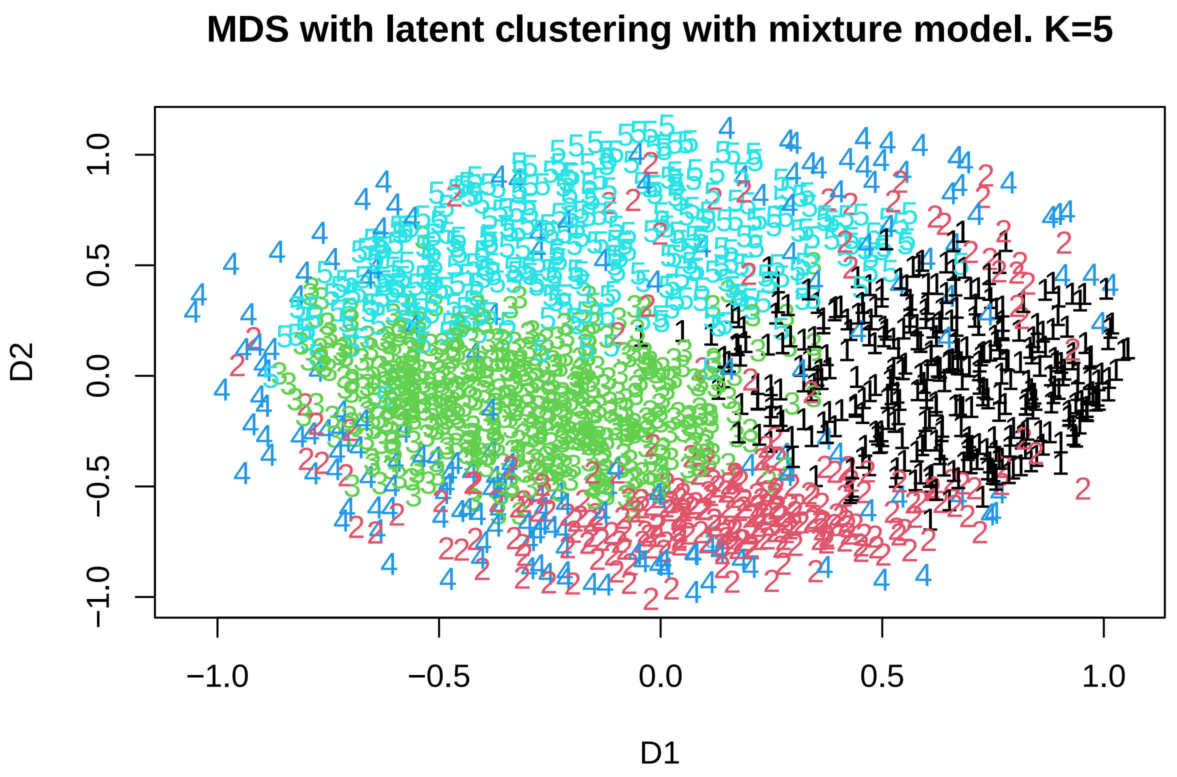
Remark: in flexmixedruns, the lower the BIC the better the model. For the sake of comparison, we have computed the clustering with poLCA with the optimal number of clusters obtained from the latent class clustering.

```
set.seed(2456)
flex_elec2 <-
flexmixedruns(electionwithna,continuous=0,discrete=12,n.cluster=1:5)

# just as example to compare the values of BIC
k <- 5
elec_poLCA <- list(NULL)
for (i in 1:k){
  elec_poLCA[i] <- poLCA(formula = formula, electionwithna,
      nclass = i, graphs = F, na.rm = F,  nrep = 5)
}
elec_poLCA
```

```
plot(1:5,flex_elec2$bicvals,typ="l",
xlab="Number of clusters",ylab="BIC")
```



```
plot(mds_elec$conf,col=flex_elec2$flexout[[5]]@cluster,
pch=clusym[flex_elec2$flexout[[5]]@cluster])
title(main = "MDS with latent clustering with mixture model. K=5")
```



The best number of clusters is 5. From the MDS plot, we notice that all the clusters look nice. There are not many outliers, and cluters 1,2,3 and 5 are elliptical-shaped with homogeneous densities of their units. Cluster 4 is a hollow circumference around the others.

(e) The original election data set also has the variables AGE and EDUC (these are the variables number 14 and 15). Define a data set election14 that has the 12 variables already used above and AGE and EDUC. Assuming that these can be treated as continuous variables (which is rather questionable at least for EDUC, but you can ignore this here), use flexmixedruns to compute a clustering based on a latent class mixture model, including estimation of the number of clusters, were the continuous variables are assumed Gaussian within clusters independently of the categorical variables. In order to use all observations for this, impute the missing values of AGE and EDUC with the mean of these variables. You can use the MDS already computed above for visualising this, but if you are curious, you can also run a new MDS for these data using the Gower coefficient.

The chunk of code below shows how to:

-1) add two columns to the original dataset

-2) turn the missing values in the first 12 columns into a category named "NA"

-3) change the NA values of the last 2 columns to the means of those columns.

-4) compute the Gower coefficient for the new dataset

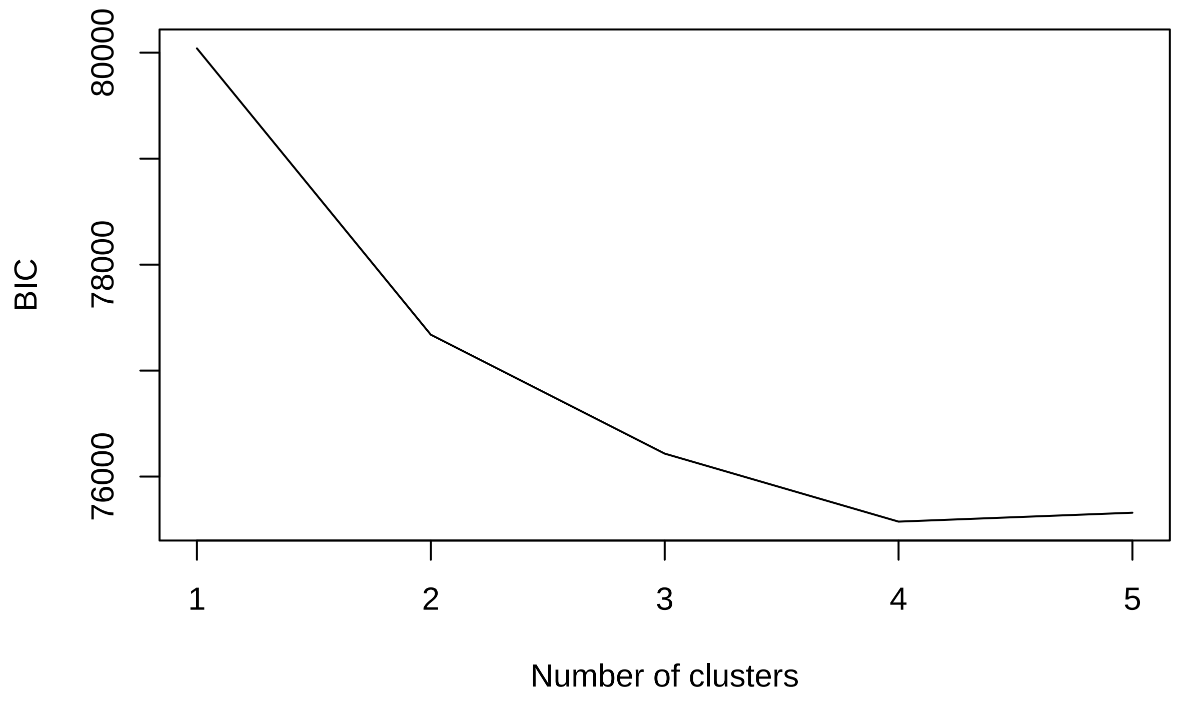
-5) compute the MDS based on the gower coeffient

```r
columns <- c(1:12, 14, 15)

# we add AGE and EDUC to the dataset
election14 <- election[,columns]


elect14 <- election14


for (i in 1:12){
levels(elect14[,i]) <- c(levels(election14[,i]),"NA")
elect14[is.na(election14[,i]),i] <- "NA"
}


# we change NA's in the last 2 columns with their column means
for (i in 1:dim(elect14)[1]){
  for (y in 13:14){
  if (is.na(elect14[i,y])){
    elect14[i,y] <- mean(elect14[,y], na.rm=T)

    }
  }
}

gower_elect <- daisy(elect14, type = list(factor=1:12))

# new mds with gower
mds_elect14 <- mds(gower_elect)
```

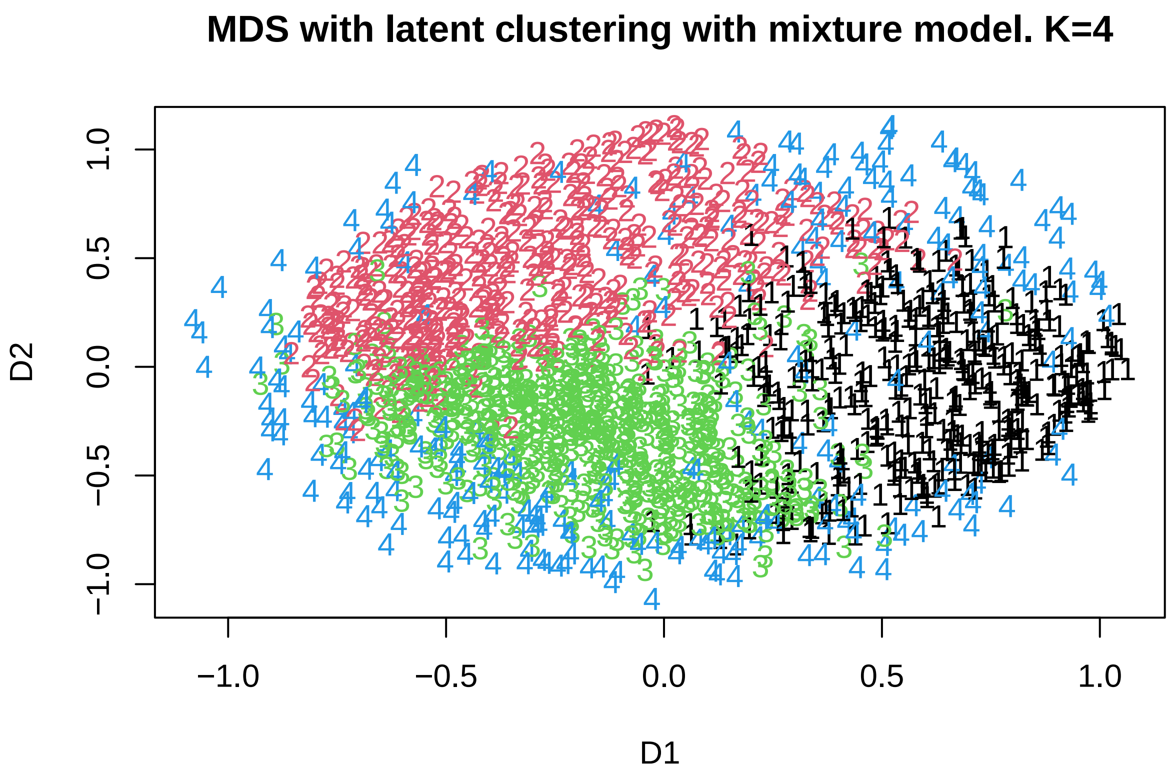
After that, we perform the same latent class clustering with flexmixedruns and we estimate the optimal number of clusters with BIC.

```r
elect14_numeric <- elect14

for (i in 1:14){
  elect14_numeric[,i] <- as.numeric(elect14_numeric[,i])
}

set.seed(825423)
flex_elec3 <- flexmixedruns(elect14_numeric, continuous = 2, discrete = 12, n.cluster = 1:5)
```

```r
plot(1:5,flex_elec3$bicvals,typ="l",
xlab="Number of clusters",ylab="BIC")
```



```r
plot(mds_elect14$conf,col=flex_elec3$flexout[[4]]@cluster,
pch=clusym[flex_elec3$flexout[[4]]@cluster])
title(main = "MDS with latent clustering with mixture model. K=4")
```



The lowest BIC is at $K = 4$, where K is the number of clusters.

In the MDS scatterplot with the new dataset (elect14_numeric), we notice that there is one main difference with the MDS scatterplot on electionwithna (with 12 variables). Cluster 3 seems to be union of cluster 2 and 3 in the previous plot.

Remark: this comparision should be taken with a grain of salt, since the plots are computed on two different MDS datasets with a different number of variables Both the last two clusterings look good, but the last one seems to have less outliers (particularly the north-east area).

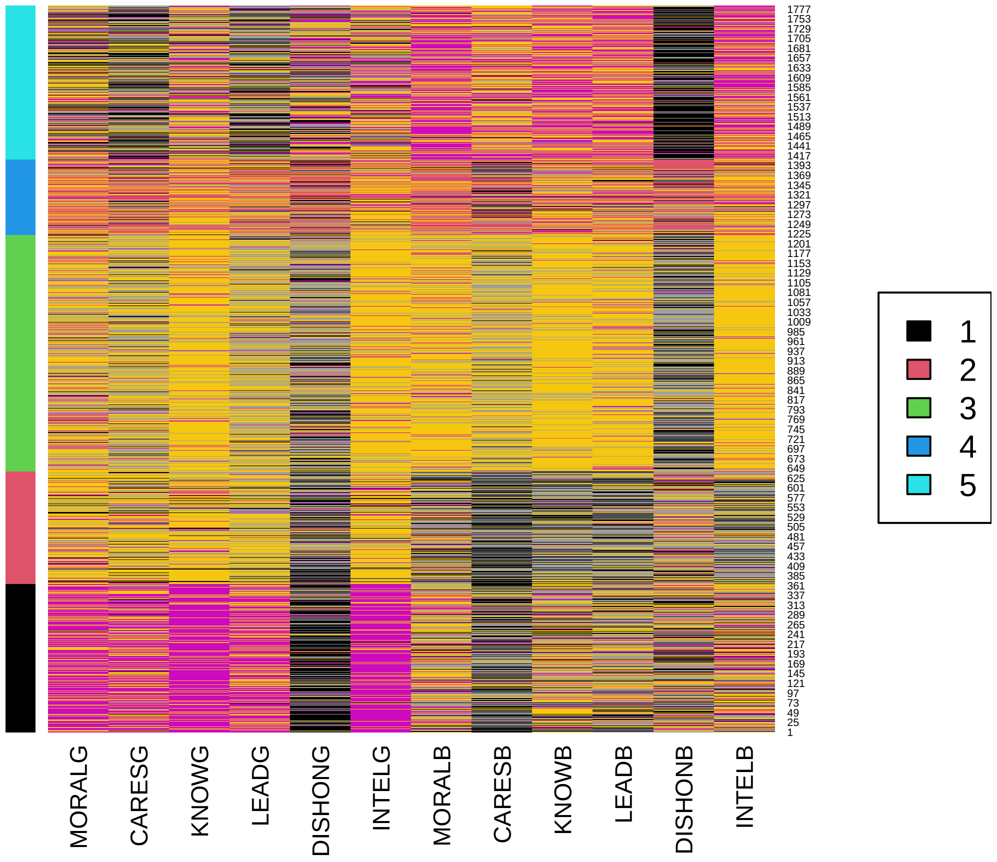
2

For one of the latent class clusterings computed in question 1 on the electionwithna-data produce a heatmap as on slide 289 of the course notes. Note that this requires a different colour choice for the heatmap "entries" than in the course notes, as the data now have more than 2 categories (it is part of the exercise to find out how to do this). Comment on the plots. Do you find the clusters convincing? Why or why not? Is there evidence against local independence?

Here we have decided to use the dataset with 12 variables and missing values labeled as "NA", applied to the second latent class model computed with flexmixedruns ($K = 5$).

```r
# heatmap, rows ordered by clusters,
elect12_numeric <- electionwithna

for (i in 1:12){
  elect12_numeric[,i] <- as.numeric(elect12_numeric[,i])
}

electionwithna_matrix <- as.matrix(elect12_numeric)

heatmap(electionwithna_matrix[order(flex_elec2$flexout[[5]]@cluster),],
  Rowv=NA, Colv = NA,
  RowSideColors=palette()[flex_elec2$flexout[[5]]@cluster]
  [order(flex_elec2$flexout[[5]]@cluster)],
  col=c(6:10),scale="none")

legend('right', legend=c(1:5), fill=c(1:5))
```

Remark: in order to create a heatmap for this dataset, we have to turn all the columns from factors to numeric variables and then turn the whole dataset into a matrix.



Let us describe the heatmap: the columns are the variables of the dataset and the rows are the observations. The vertical colored line on the left side shows the clusters, and the observations of the segment of the same color belong to the same cluster (indeed, there are 5 segments).

The interpretation of clusters is much harder from the heatmap than from a pairplot or a scatterplot with MDS. However, some patterns are still noticeable:

- The observations that belong to cluster 1 have a strong concentration of the violet color on the first 5 variables to the left and a noticeable black column on "dishong" revealing that all the observations in cluster 1 follows the same patterns.

- In cluster 2 the first 6 variables to the left are mainly yellow with once again the "dishong" variable covered with mainly black. The other variables seem to be mixed without a dominant color.
- In cluster 3 the yellow color is dominant across all variables but "dishong" and "dishonb".
- Cluster 4 has a great variety of colors and none seems dominant. However cyan and blue are uncommon (also in the other clusters).

Cluster 5 has the last 5 variables to the right dominated by the violet color except in "dishonb".

If on one hand we are happy to say that the clusters visually make sense, on the other hand there is evidence against local independence. In facts, if the same observations have the same color on a certain variable given the same color on another variable, than there may be dependence between these two variables.

For instance, the violet patters in cluster 1 and cluster 5 reveal local dependence across the variables with this color as well as "dishog" and "dishonb" probably violate local independence in cluster 3.

3

Task:

Assume a situation with 10 categorical variables. Five variables are binary, three variables have three categories, and two variables have five categories. What is the number of free parameters for:

- (a) a general categorical model that models all possible probabilities,
- (b) a latent class mixture model with 4 mixture components?

Solution (a):

In this scenario there are 5 binary variables and 5 polytomous variables (i.e. variables with more than two categories). Trivially, binary variables have just one variable parameter.

When it comes to polytomous variables, the degrees of freedom are computed by isolating one parameter from the variable expression and letting the others be "free" to vary. Hence, for each variable, the number K of free parameters can be derived as follows:

$$K = number\ of\ categories - 1$$

Then, we multiply by the number of variables with the same number of categories and finally to get the total number of free parameters, we multiply together the free parameters of each variable as follows:

- The five binary variables: $5 \times (2 - 1) = 5$
- The three variables with three categories: $3 \times (3 - 1) = 6$
- The two variables with five categories: $2 \times (5 - 1) = 8$

Hence the model (a) has:

$$5 \times 6 \times 8 = 240$$

free parameters.

Solution (b):

The same principle is used to compute the free parameters in the second model. Since the second model has 4 mixture components, we have to multiply the free parameters of each variable by 4. Moreover, the weigths of the mixture model are 4 and their sum is equal to 1. As a consequence, 3 of the 4 weigths are free parameters. The total number of free parameters is computed below:

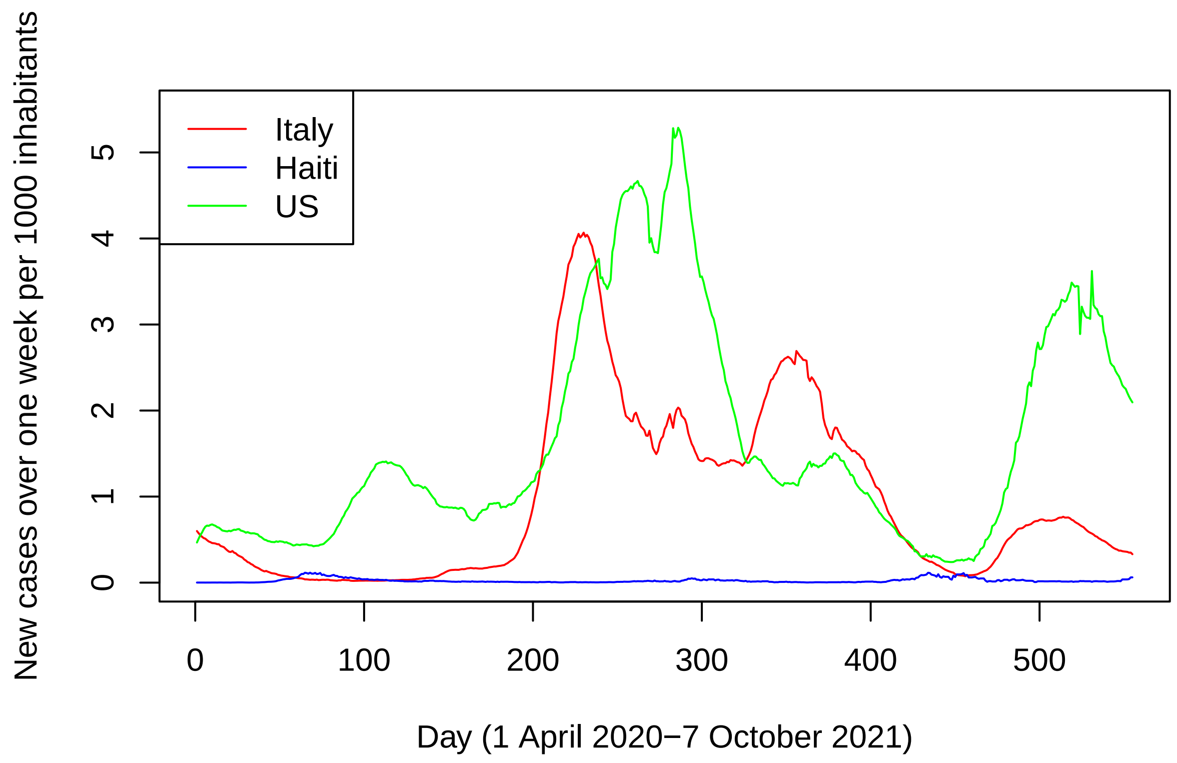
$$(5 + 6 + 8) \times 4 + 3 = 79$$

4

Consider the COVID data set analysed in Chapter 7 of the course slides.Consider Italy, Haiti, and the USA (country 79, 69, and 164). Produce residual plots, i.e., plots with the time points on the x-axis and the residuals (difference between data and fit) on the y-axis) for these countries for:

We upload the data and show the weekly new cases of the time series of the above mentioned countries.

```r
#setwd("C:/Users/...")
covid2021 <- read.table("covid2021.dat")
covid2021cl <- covid2021[,5:559] # This selects the variables for clustering
covid2021cl <- data.frame(covid2021cl)


plot(1:555, covid2021cl[79,], type='l', col = 'red',
  ylab="New cases over one week per 1000 inhabitants",
  xlab="Day (1 April 2020-7 October 2021)", ylim=c(0, 5.5))
points(1:555, covid2021cl[69,], type = 'l', col='blue')
points(1:555, covid2021cl[164,], type = 'l', col='green')
legend("topleft", legend = c("Italy","Haiti",  "US"),
  col = c("red",  "blue", "green"),
  lty = 1)
```

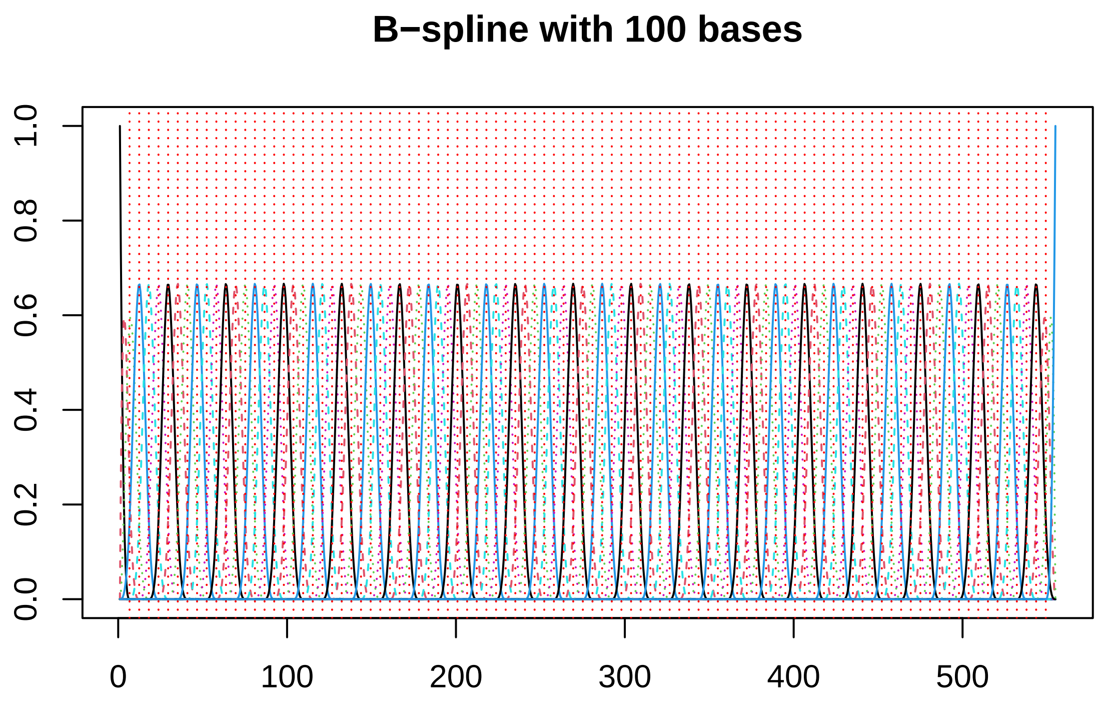


### (a) the fit by a B-spline basis with p = 100

The code below performs the B-Spline with p=100 and create the following plots:

- B-spline
- Smooth splines with smooth mean
- Residual plots.

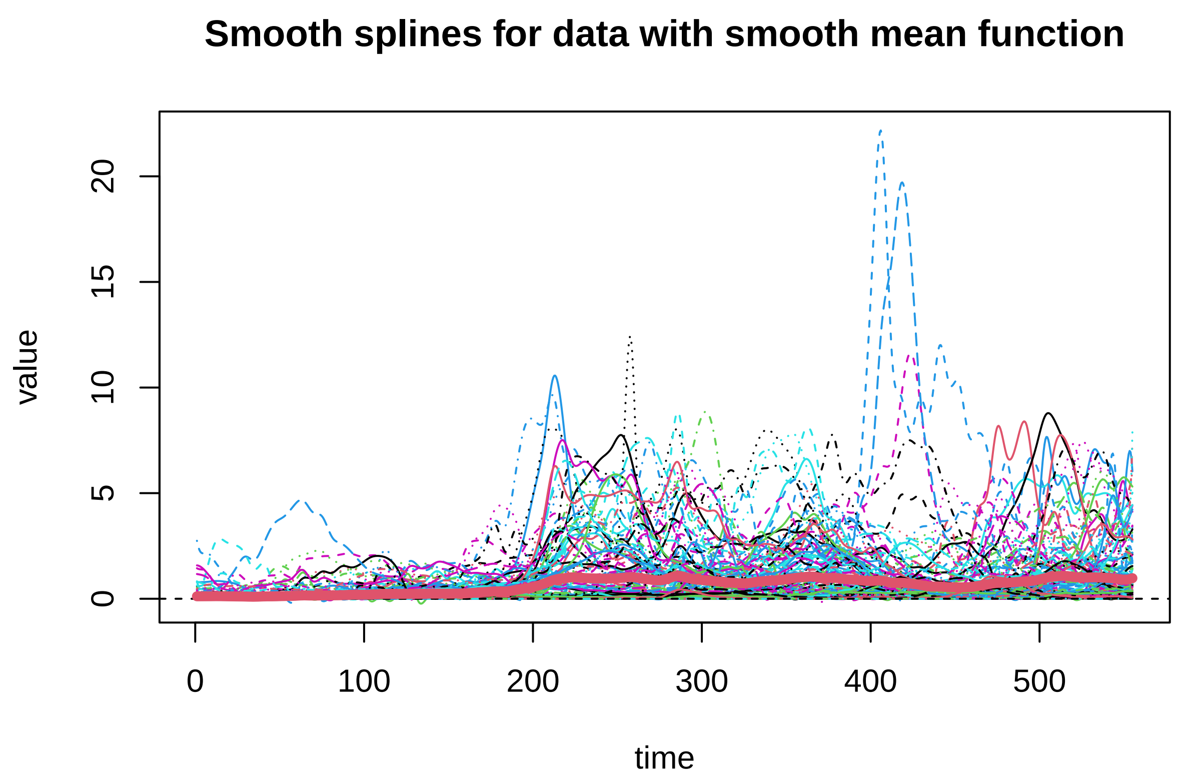
```
# Constructing B-spline basis
# Splines approximating data as linear combinations of B-spline basis

bbasis <- create.bspline.basis(c(1,555),nbasis=100) # same with p=100
fdcovid <- Data2fd(1:555,y=t(as.matrix(covid2021cl)),basisobj=bbasis)

# Plot basis
plot(bbasis)
title(main= "B-spline with 100 bases")
```



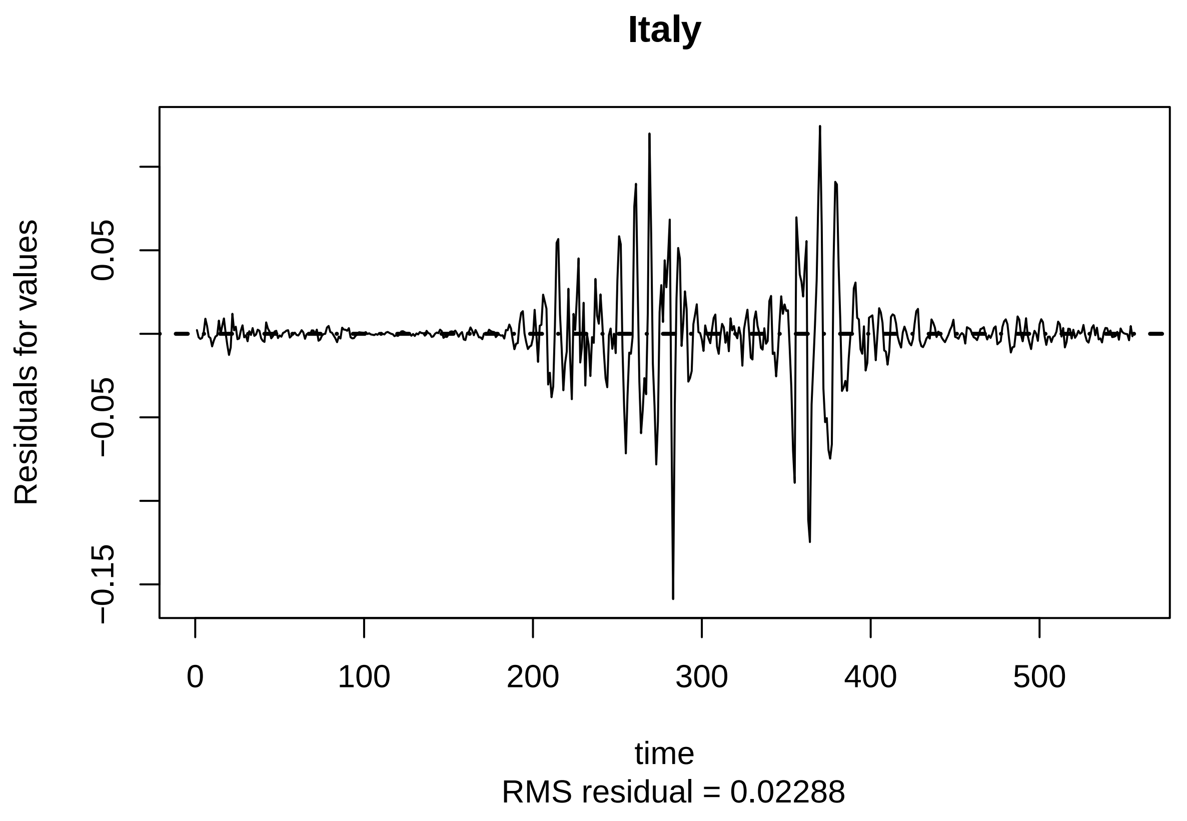
```r
plot(fdcovid)
```

```
## [1] "done"
```

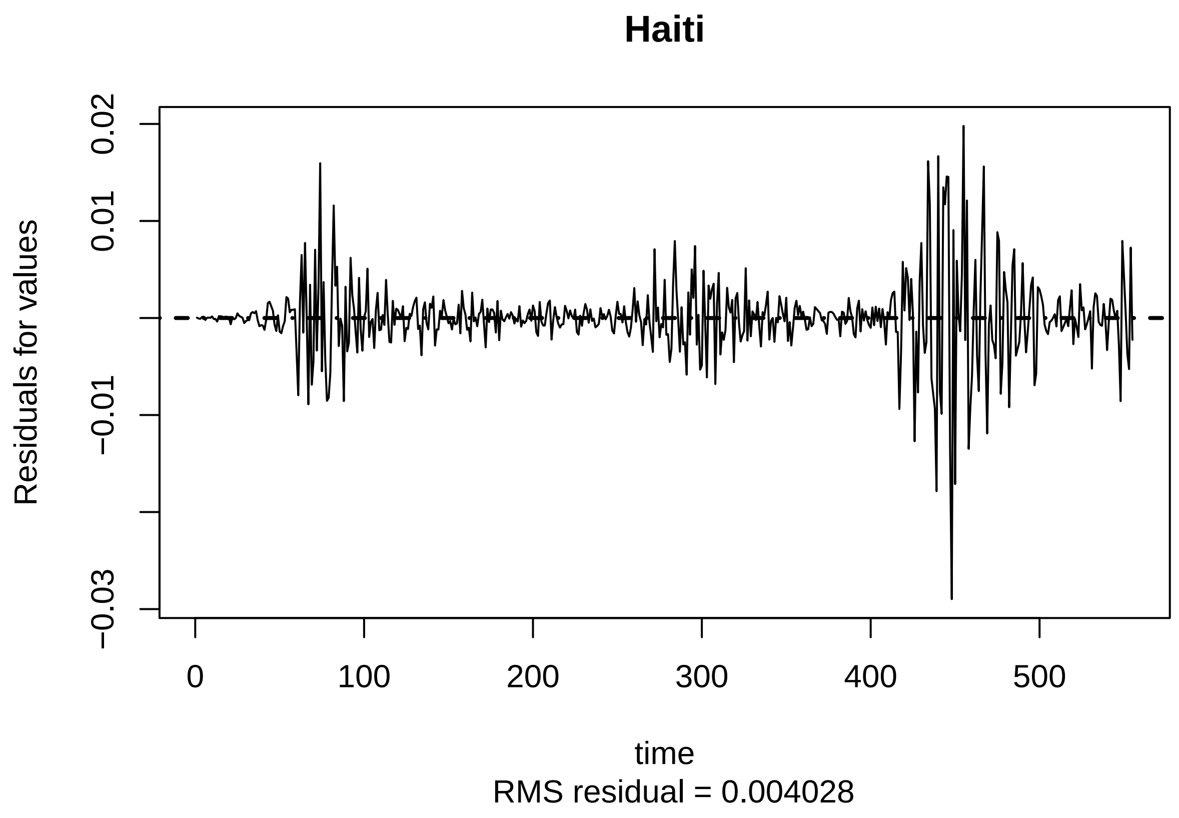
```r
mcovid <- mean.fd(fdcovid)
lines(mcovid,col=2,lwd=5)
title(main = "Smooth splines for data with smooth mean function")
```



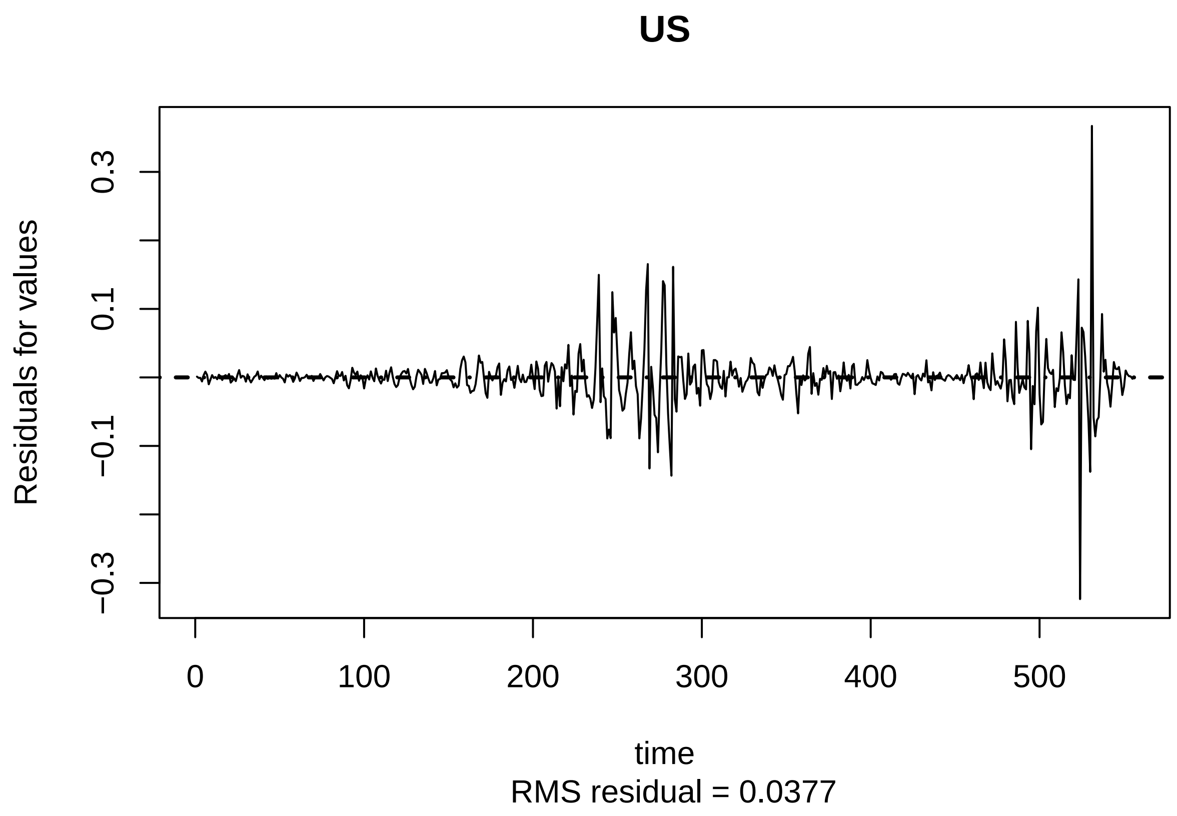
```
# Show residual plots of individual countries
plotfit.fd(t(covid2021cl),1:555,fdcovid,index=79,residual = T,cex.pch=0.5)
```



```r
plotfit.fd(t(covid2021cl),1:555,fdcovid,index=69,residual = T,cex.pch=0.5)
```



```
plotfit.fd(t(covid2021cl),1:555,fdcovid,index=164,residual = T, cex.pch=0.5)
```



The residual plots reveal that each country has experienced some peaks in the difference between real data and their fit. For example, the residuals are very big from 200 to 400 days for Italy, while Haiti has three peaks, one right before 100 days, another around 300 days and the last one between 400 and 500 days. Finally, US has two sections with big residuals, the fist one from 200 to 300 and the last one around 500 days.

Remark: I do not know what the model assumptions are but if the residuals are too big in some days, these assumptions might be violated and they must be checked.

The other plots are pretty self explainatory with their titles.

(b) the fit by a 5-dimensional principal components basis as shown on p. 310 of the slides.

Comment on potential model assumption violations from these plots. (It is part of the exercise to figure out how to produce these plots.)

Let us compute the principal components basis with 5D.

```
covidpca <- pca.fd(fdcovid, nharm = 5)


covidpca$varprop # Percentage of variance
```

```
## [1] 0.40871917 0.18555649 0.11717077 0.05581147 0.04606967
```

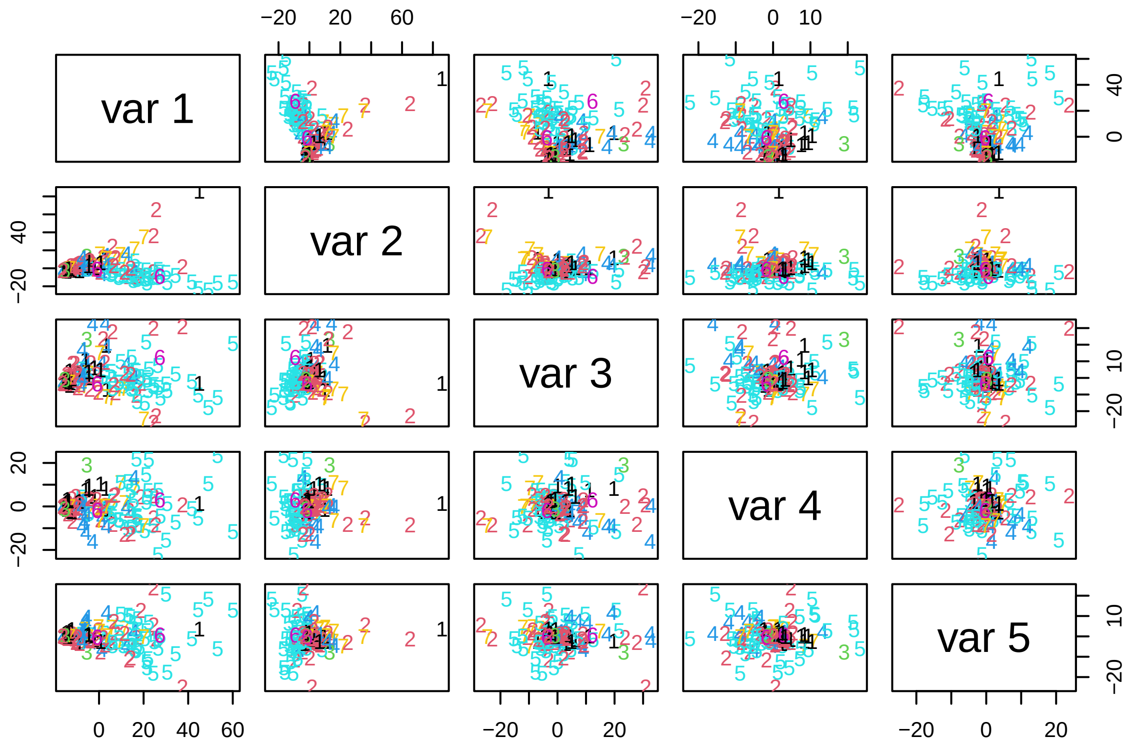
```
cumsum(covidpca$varprop) # Cumulative percentage of variance
```

```
## [1] 0.4087192 0.5942757 0.7114464 0.7672579 0.8133276
```

```
ncontinent <- as.numeric(as.factor(covid2021$continent))
levels(as.factor(covid2021$continent))
```

```
## [1] "Africa"          "Asia"            "Australia"       "Central America"
## [5] "Europe"          "North America"   "South America"
```

```
pairs(covidpca$scores,col=ncontinent,pch=clusym[ncontinent])
```



```
# 1: africa, 2: Asia, 3: Australia, 4: Central America, 5:Europe, 6: North America, 7: South America
```

We take a look at the paired scatterplots of the scores and see if the captured 80% of the total variance is enough to see some patterns. Here the time is not represented and only the scores of the PC's are compared together.

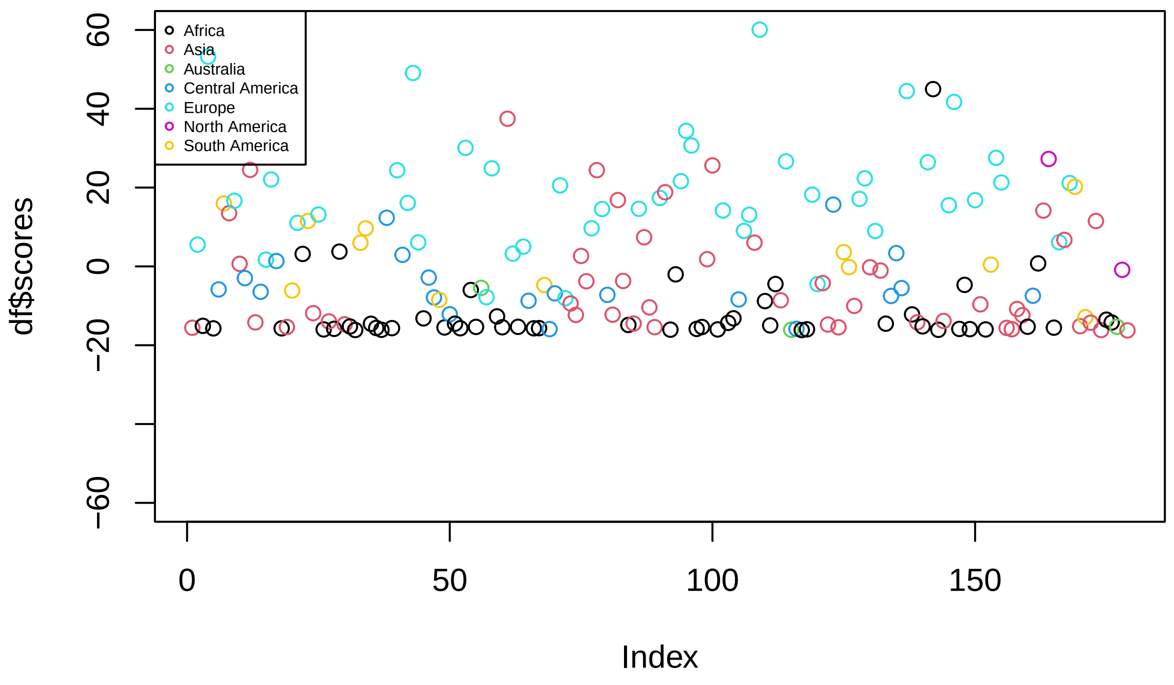
Africa (1) has few points and are usually centered close to (0,0) in most of the paired plots with an outlier as exception. Europe (5) has many observations and its points are much more sparse. Anyway, it is very difficult to detect patterns from these plots. We will find a better way in point 5.

5

Representing all countries in the COVID data set by the first functional principal component scores only, run a one-way analysis of variance to test whether there is evidence that the scores from different continents have different means. Also visualise the scores so that the continents can easily be compared, and interpret plot and result (try to figure out what larger or smaller/positive or negative scores on the first principal component actually mean).

```r
covidpca1 <- pca.fd(fdcovid, nharm = 1)


df <- data.frame(scores=covidpca1$scores, continent=as.factor(ncontinent))

plot(df$scores, col=ncontinent, ylim=c(-60,60))
legend("topleft", legend = levels(as.factor(covid2021$continent))
       , col = 1:7, pch=1, cex=0.5)
```

Important Remark: the scores of PC's can be negative because the theory used by the function "pca.fd" does allow the scores to be negative. There is no lower bound set to 0, despite it does not make sense to have negative scores for a function that can only have positive or null values.



```r
# anova
anova <- aov(scores ~ continent, data = df)
summary(anova)  # we reject the null
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)    
## continent     6  25062    4177   27.62 <2e-16 ***
## Residuals   172  26008     151                   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The one-way variance test is performed with ANOVA with the function ‘aov’. Its formula tries to explain the variance of the scores of the PC’s given the associated number of the continent. The null hypothesis claims that the scores of different continents have different means. Since the p-value is approximately 0, it is rejected as we hoped.

From the last plot, Africa has most of the scores set around -20 and Asia as well, even though some scores are more positive. Europe has the highest scores.