Clamix – Clustering Symbolic Objects Described by Discrete Distributions

Vladimir Batagelj

joint work with Nataša Kejžar and Simona Korenjak-Černe

University of Ljubljana

Workshop on Theory and Application of High-dimensional Complex and Symbolic Data Analysis in Economics and Management Science

October 27–29, 2011, Beihang University, Beijing, China

Outline

Symbolic objects

In this paper we present an approach to clustering of (very) large data sets of mixed units – units measured in different scales.
The approach is based on representation of units by *symbolic objects* (SOs) [Billard, L., Diday, E. (2006)]. The SOs can describe either single units or groups of initial units condensed into SOs in a pre-processing step.
For clustering of SOs we adapted two classical clustering methods:

- *leaders method* (a generalization of k-means method [Hartigan, J. A. (1975)], dynamic clouds [Diday, E. (1979)]).
- Ward's *hierarchical clustering method* [Ward, J. H. (1963)].

Symbolic objects described with distributions

An SO X is described by a list $X = [\mathbf{x}_i]$ of descriptions of variables V_i. In our model, each variable is described with frequency distribution (*bar chart*) of its values

$$\mathbf{f}_{xi} = [f_{xi1}, f_{xi2}, \ldots, f_{xik_i}].$$

With

$$\mathbf{x}_i = [p_{xi1}, p_{xi2}, \ldots, p_{xik_i}]$$

we denote the corresponding probability distribution.

$$\sum_{j=1}^{k_i} p_{xij} = 1, \quad i = 1, \ldots, m$$

Clustering and optimization

We approach the clustering problem as an optimization problem over the set of *feasible* clusterings Φ_k – partitions of units into k clusters. The *criterion function* has the following form

$$P(\mathbf{C}) = \sum_{C \in \mathbf{C}} p(C). \tag{1}$$

The *total error* $P(\mathbf{C})$ of the clustering $\mathbf{C}$ is a sum of *cluster errors* $p(C)$. We assume a model in which the error of a cluster is a sum of differences of its units from the cluster's *representative* T

$$p(C, T) = \sum_{X \in C} d(X, T). \tag{2}$$

Representatives

The best representative is called a *leader*

$$T_C = \underset{T}{\operatorname{argmin}}\, p(C, T). \tag{3}$$

Then we define

$$p(C) = p(C, T_C) = \min_T \sum_{X \in C} d(X, T). \tag{4}$$

The SO X is described by a list $X = [\mathbf{x}_i]$. Assume that also representatives are described in the same way $T = [\mathbf{t}_i]$, $\mathbf{t}_i = [t_{i1}, t_{i2}, \ldots, t_{ik_i}]$.

Dissimilarity between SOs

We introduce a dissimilarity measure between SOs with

$$d(X,T) = \sum_i \alpha_i d(\mathbf{x}_i, \mathbf{t}_i), \quad \alpha_i \geq 0, \quad \sum_i \alpha_i = 1, \qquad (5)$$

where

$$d(\mathbf{x}_i, \mathbf{t}_i) = \sum_{j=1}^{k_i} w_{xij}\delta(p_{xij}, t_{ij}), \quad w_{xij} \geq 0. \qquad (6)$$

The weight w_{xij} can be for the same unit X different for each variable V_i (needed in descriptions of ego-centric networks, population pyramids, etc.).

Leaders method

Leaders method is a generalization of a popular nonhierarchical clustering k-means method.
The idea is to get "optimal" clustering into a pre-specified number of clusters with the following iterative procedure:

determine an initial clustering
repeat
  determine leaders of the clusters in the current clustering;
  assign each unit to the nearest new leader – producing a new clustering
until the leaders stabilize.

Selection of the new leaders

Given a cluster C, the corresponding leader T_C is the solution of the problem

$$T_C = \operatorname{argmin}_T \sum_{X \in C} d(X, T) = \operatorname{argmin}_T \sum_{X \in C} \sum_i \alpha_i d(\mathbf{x}_i, \mathbf{t}_i)$$

$$= \operatorname{argmin}_T \sum_i \alpha_i \sum_{X \in C} d(\mathbf{x}_i, \mathbf{t}_i) = \left[\operatorname{argmin}_{\mathbf{t}_i} \sum_{X \in C} d(\mathbf{x}_i, \mathbf{t}_i) \right]_{i=1}^{m}$$

Therefore $T_C = [\mathbf{t}_i^*]$ and $\mathbf{t}_i^* = \operatorname{argmin}_{\mathbf{t}_i} \sum_{X \in C} d(\mathbf{x}_i, \mathbf{t}_i)$

Leaders

To simplify the notation we omit the index i.

$$\mathbf{t}^* = \underset{\mathbf{t}}{\operatorname{argmin}} \sum_{X \in C} d(\mathbf{x}, \mathbf{t}) = \underset{\mathbf{t}}{\operatorname{argmin}} \sum_{X \in C} \sum_{j=1}^{k} w_{xj} \delta(p_{xj}, t_j)$$

$$= \underset{\mathbf{t}}{\operatorname{argmin}} \sum_{j=1}^{k} \sum_{X \in C} w_{xj} \delta(p_{xj}, t_j)$$

$$= \Big[\underset{t_j \in \mathbb{R}}{\operatorname{argmin}} \sum_{X \in C} w_{xj} \delta(p_{xj}, t_j) \Big]_{j=1}^{k}$$

Leaders

Again we omit the index j

$$t^* = \underset{t \in \mathbb{R}}{\operatorname{argmin}} \sum_{x \in C} w_x \delta(p_x, t)$$

This is a standard optimization problem with one real variable. The solution has to satisfy the condition

$$\frac{\partial}{\partial t} \sum_{x \in C} w_x \delta(p_x, t) = 0$$

or

$$\sum_{x \in C} w_x \frac{\partial \delta(p_x, t)}{\partial t} = 0 \tag{7}$$

Dissimilarities δ

	$\delta(x,t)$	t^*_{ij}
1	$(p_x - t)^2$	$\frac{P_{ij}}{A_{ij}}$
2	$(\frac{p_x - t}{t})^2$	$\frac{Q_{ij}}{P_{ij}}$
3	$\frac{(p_x - t)^2}{t}$	$\sqrt{\frac{Q_{ij}}{A_{ij}}}$
4	$(\frac{p_x - t}{p_x})^2$	$\frac{H_{ij}}{F_{ij}}$
5	$\frac{(p_x - t)^2}{p_x}$	$\frac{A_{ij}}{H_{ij}}$
6	$\frac{(p_x - t)^2}{p_x t}$	$\sqrt{\frac{P_{ij}}{H_{ij}}}$

$$A_{ij} = \sum_{X \in C} w_{Xij} \qquad P_{ij} = \sum_{X \in C} w_{Xij} p_{Xij} \qquad Q_{ij} = \sum_{X \in C} w_{Xij} p^2_{Xij}$$

$$H_{ij} = \sum_{X \in C} \frac{w_{Xij}}{p_{Xij}} \qquad F_{ij} = \sum_{X \in C} \frac{w_{Xij}}{p^2_{Xij}}$$

Leaders

For $\delta_1(p_x, t) = (p_x - t)^2$ we get from (8)

$$0 = \sum_{x \in C} w_x \frac{\partial}{\partial t}(p_x - t)^2 = -2 \sum_{x \in C} w_x (p_x - t)$$

Therefore

$$t^* = \frac{\sum_{x \in C} w_x p_x}{\sum_{x \in C} w_x} = \frac{P}{A}$$

Leaders for δ_1

Let $w_{xij} = w_{xi}$ then for each $i = 1, \ldots, m$:

$$\sum_{j=1}^{k_i} t^*_{ij} = \frac{1}{A_i} \sum_{j=1}^{k_i} \sum_{X \in C} w_{xi} p_{xij} = \frac{1}{A_i} \sum_{X \in C} w_{xi} \sum_{j=1}^{k_i} p_{xij} = 1$$

The leaders' components are *distributions*.
Let further $w_{xij} = n_{xi}$ then for each $i = 1, \ldots, m$:

$$t^*_{Cij} = \frac{\sum_{X \in C} n_{xi} p_{xij}}{\sum_{X \in C} n_{xi}} = \frac{\sum_{X \in C} f_{xij}}{\sum_{X \in C} n_{xi}} = \frac{f_{Cij}}{n_{Ci}} = p_{Cij}$$

The leader of a union of clusters is again its distribution!

Determining the new clustering

Given leaders $\mathbf{T}$ the corresponding optimal clustering $\mathbf{C}^*$ is determined from

$$P(\mathbf{C}^*) = \sum_{X \in \mathcal{U}} \min_{T \in \mathbf{T}} d(X, T) = \sum_{X \in \mathcal{U}} d(X, T_{c^*(X)}) \qquad (8)$$

where

$$c^*(X) = \operatorname*{argmin}_k d(X, T_k)$$

We assign each unit X to the closest leader $T_k \in \mathbf{T}$.

Hierarchical agglomerative clustering

The hierarchical agglomerative clustering procedure is based on a step-by-step merging of the two closest clusters.

each unit forms a cluster: $\mathbf{C}_n = \{\{X\}: X \in \mathcal{U}\}$;
they are at level 0: $h(\{X\}) = 0,\ X \in \mathcal{U}$;
for $k = n-1$ **to** 1 **do**
  determine the closest pair of clusters
    $(u, v) = \text{argmin}_{i,j:\ i \neq j}\{D(C_i, C_j): C_i, C_j \in \mathbf{C}_{k+1}\}$;
  join the closest pair of clusters $C_{(uv)} = C_u \cup C_v$
    $\mathbf{C}_k = (\mathbf{C}_{k+1} \setminus \{C_u, C_v\}) \cup \{C_{(uv)}\}$;
    $h(C_{(uv)}) = D(C_u, C_v)$
    determine the dissimilarities $D(C_{(uv)}, C_s), C_s \in \mathbf{C}_k$
endfor

$\mathbf{C}_k$ is a partition of the finite set of units $\mathcal{U}$ into k clusters. The level $h(C)$ of the cluster $C_{(uv)} = C_u \cup C_v$.

Generalized Ward's relation

To obtain the compatibility with the adapted leaders method, we define the dissimilarity between clusters C_u and C_v, $C_u \cap C_v = \emptyset$, as

$$D(C_u, C_v) = p(C_u \cup C_v) - p(C_u) - p(C_v).$$

$\mathbf{u}_i$ and $\mathbf{v}_i$ are components of the leaders of clusters C_u and C_v, and $\mathbf{z}_i$ is a component of the leader of the cluster $C_u \cup C_v$. After some computation we get for δ_1:

$$z = \frac{A_u u + A_v v}{A_u + A_v}.$$

and

$$D(C_u, C_v) = \sum_i \alpha_i \sum_j \frac{A_{uij} \cdot A_{vij}}{A_{uij} + A_{vij}} (u_{ij} - v_{ij})^2 \qquad (9)$$

a *generalized Ward's relation*.

Other dissimilarities

Relations similar to Ward's can be derived for other dissimilarity measures $\delta(x,t)$ (see [Kejžar, N. et al. (2011)]).

δ	$D(C_u, C_v)$	z
δ_1	$\frac{A_u \cdot A_v}{A_u + A_v}(u-v)^2$	$\frac{A_u u + A_v v}{A_u + A_v}$
δ_2	$\frac{(u-z)^2}{uz^2}P_u + \frac{(v-z)^2}{vz^2}P_v$	$\frac{uP_u + vP_v}{P_u + P_v}$
δ_3	$2(zA_z - uA_u - vA_v)$	$\sqrt{\frac{u^2A_u + v^2A_v}{A_u + A_v}}$
δ_4	$(u-z)^2F_u + (v-z)^2F_v$	$\frac{H_u + H_v}{1/uH_u + 1/vH_v}$
δ_5	$\frac{(u-z)^2}{uz^2}A_u + \frac{(v-z)^2}{vz^2}A_v$	$\frac{A_u + A_v}{H_u + H_v}$
δ_6	$\frac{(u-z)^2}{u^2z}P_u + \frac{(v-z)^2}{v^2z}P_v$	$\frac{P_u + P_v}{H_u + H_v}$

The proposed approach is partially implemented in the R-packages `Clustddist` and `Clamix`.

Sheme of analysis

raw data

⇓

ENCODE

⇓

unified data

⇓

MAKE SOs

⇓

SOs - lists of distributions

⇓

`leaderSO`

⇓

clustering and cluster leaders

⇓

`hclustSO`

⇓

hierarchy and cluster leaders

⇓

ANALYSIS

⇓

dendrogram, reports

Encoding of numerical variables


```
# make "encWater" encoding

encWater <- list(
  "[0]"            = function(x) x<=0,
  "(0,5.65]"       = function(x) x<=5.65,
  "(5.65,29.5]"    = function(x) x<=29.5,
  "(29.5,53.9)"    = function(x) x< 53.9,
  "[53.9,62.4)"    = function(x) x< 62.4,
  "[62.4,70.75)"   = function(x) x< 70.75,
  "[70.75,78.05]"  = function(x) x<=78.05,
  "(78.05,88)"     = function(x) x< 88,
  "[88,100]"       = function(x) x<=100,
  "NA"             = function(x) TRUE )

encodeSO <- function(x,encoding,codeNA){
  if(is.na(x)) return(codeNA)
  for(i in 1:length(encoding)) if(encoding[i](x)) return(i)
}
```

SR22

Encoding of numerical variables


```
> source("C:\\...\\sr14\\encFood.R")
> b <- read.delim("./sr14/abbrev.txt",dec=",",row.names=2)
> colnames(b)
 [1] "NDB_No"      "Water"       "Energ_Kcal"  "Protein"     "Tot_Lipid"
 [6] "Carbohydrt"  "Fiber_TD"    "Ash"         "Calcium"     "Phosphorus"
[11] "Iron"        "Sodium"      "Potassium"   "Magnesium"   "Zinc"
[16] "Copper"      "Manganese"   "Selenium"    "Vit_A"       "Vit_E"
[21] "Thiamin"     "Ribolflavin" "Niacin"      "Panto_Acid"  "Vit_B6"
[26] "Folate"      "Vit_B12"     "Vit_C"       "FA_Sat"      "FA_Mono"
[31] "FA_Poly"     "Cholestrl"   "GmWt_1"      "GmWt_Desc1"  "GmWt_2"
[36] "GmWt_Desc2"  "Refuse_Pct"
> water <- b[[2]]
> water[1:10]
 [1] 15.87 15.87  0.24 42.41 41.11 48.42 51.80 39.28 36.75 37.65
> names(encWater)
 [1] "[0]"          "(0,5.65]"     "(5.65,29.5]"  "(29.5,53.9)"
 [5] "[53.9,62.4)"  "[62.4,70.75)" "[70.75,78.05]" "(78.05,88)"
 [9] "[88,100]"     "NA"
> wat <- sapply(water,function(x) encodeSO(x,encWater,10))
> wat[1:10]
 [1] 3 3 2 4 4 4 4 4 4 4
> varCats <- vector("list",31)
> varCats[[1]]  <- names(encWater)
  ...
> varCats[[31]] <- names(encCholestr)
> names(varCats) <- colnames(b)[2:32]
> foodSO <- vector("list",nVar)
> foodSO[[1]] <- wat
 ...
> foodSO[[31]] <- cho
```

Encoding of ordinal or nominal variables


```
cf <- read.table("./cars/cars.csv",header=TRUE,
  dec=".",row.names=1)
levType <- c("LI","KL","EN","KA","KB","RO","TE","KU")
typ <- factor(cf$type,levels=levType)
carsSO <- vector("list",26)
names(carsSO) <- colnames(cf)
carsSO$type <- as.integer(typ)
carsCats <- vector("list",26)
names(carsCats) <- colnames(cf)
carsCats$type <- c(levType,NA)
```

Data and metadata


```
setwd("C:/..../clamix.R")
source("C:\\....\\clamix.R\\clamix.R")
source("C:\\....\\clamix.R\\sr14\\encFood.R")
nVar <- 31
varCats <- vector("list",nVar)
varCats[[1]]  <- names(encWater)
varCats[[2]]  <- names(encEnergKC)
varCats[[3]]  <- names(encProtein)
varCats[[4]]  <- names(encTotLipi)
varCats[[5]]  <- names(encCarbohyd)
.......
varCats[[28]] <- names(encFaSat)
varCats[[29]] <- names(encFaMono)
varCats[[30]] <- names(encFaPoly)
varCats[[31]] <- names(encCholestr)
nVarP <- nVar+1
nCats <- sapply(varCats,length)
so <- emptySO(nCats)
b <- read.delim("./sr14/abbrev.txt",dec=",",row.names=2)
numSO <- nrow(b)
names(varCats) <- colnames(b)[2:32]
long <- rownames(b)
namedSO <- so; names(namedSO) <- c(names(varCats),"num")
for(i in 1:nVar) names(namedSO[[i]]) <- varCats[[i]]
save(nVar,nVarP,so,namedSO,numSO,long,varCats,file="./sr14/sr14.meta")
SOs <- vector("list",numSO)
for(i in 1:numSO){
  st <- so
  for(j in 1:nVar) st[[j]][foodSO[i,j]] <- 1
  st$num <- 1
  names(SOs)[[i]] <- rownames(foodSO)[[i]]
  SOs[[i]] <- st
}
save(nVar,nVarP,so,numSO,SOs,file="./sr14/sr14.so")
```

Transformation — classes

A `list` with a number of `data.frames` that consist of one variable described with distributions each can be transformed into a `symData` object. A *`symData object`* is an R class for description of histogram symbolic data. The class is represented as a list of:

`SOs`	a vector of symbolic objects (of *class `symObject`*); the transformed data set
`so`	an empty `symObject`
`namedSO`	an empty `symObject` with names for categories of each variable
`alpha`	a vector of weights α_j
`type`	a type of `symData`.

A `symObject` is an R class for description of a histogram symbolic object (one unit). It is represented as a list of variables. Each variable is represented as a vector. The last two components of the vector correspond to the number of `NA`s and the size (`num`) of the variable items.
The transformation of the data set is done via function `create.symData`.

Transformation — example


```
data(popul06f)
data(popul06m)
datalist <- list("M"=popul06f,"F"=popul06m)
dataset <- create.symData(datalist,"fDist")
summary(dataset) # description of the class

summary symData :

Dimension (units x variables): 224 x 2
Type of distributions: fDist
Outlook of symObject:
symObject :
Number of variables: 2
M
  0-4   5-9 10-14 15-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54
    0     0     0     0     0     0     0     0     0     0     0
55-59 60-64 65-69 70-74 75-79   80+    NA   num
    0     0     0     0     0     0     0     0
F
  0-4   5-9 10-14 15-19 20-24 25-29 30-34 35-39 40-44 45-49 50-54
    0     0     0     0     0     0     0     0     0     0     0
55-59 60-64 65-69 70-74 75-79   80+    NA   num
    0     0     0     0     0     0     0     0
```

distSO


```
# computes weighted squared Euclidean dissimilarity between SOs

distSO <- function(X,Y,nVar,alpha){
  d <- numeric(nVar)
  for(i in 1:nVar) {
    ln <- length(X[[i]]);
    nX <- as.double(X[[i]][ln]); nY <- as.double(Y[[i]][ln])
    d[i] <- nX*sum((X[[i]][-ln]/nX-Y[[i]][-ln]/nY)**2)
  }
  dis <- as.numeric(d %*% alpha)/2
  if (is.na(dis)) dis <- Inf
  return(dis)
}
```

leaderSO


```
leaderSO <- function(dataset,maxL){
  #attach(dataset)
  so <- dataset$so; alpha <- dataset$alpha
  SOs <- dataset$SOs; nVar <- length(so)
  numSO <- length(SOs)
  L <- vector("list",maxL); Ro <- numeric(maxL)
# random partition into maxL clusters
  clust <- sample(1:maxL,numSO,replace=TRUE)
  tim <- 1; step <- 0
  repeat {
    step <- step+1
  # new leaders - determine the leaders of clusters in current partition
    for(k in 1:maxL){L[[k]] <- so; names(L)[[k]] <- paste("L",k,sep="")}
    for(i in 1:numSO){j <- clust[i];
      for(k in 1:nVar) L[[j]][[k]] <- L[[j]][[k]] + SOs[[i]][[k]] }
  # new partition - assign each unit to the nearest new leader
    clust <- integer(numSO)
    R <- numeric(maxL); p <- double(maxL)
    for(i in 1:numSO){d <- double(maxL)
      for(k in 1:maxL){d[k] <- distSO(SOs[[i]],L[[k]],nVar,alpha)}
      r <- min(d); j <- which(d==r)
      if(length(j)==0){
        cat("unit",i,"\n",d,"\n"); flush.console(); print(SOs[[i]]); flush.console()
        u <- which(is.na(d))[[1]]; cat("leader",u,"\n"); print(L[[u]])
        stop()}
      j <- which(d==r)[[1]];
      clust[i] <- j
      p[j] <- p[j] + r; if(R[j]<r) R[j]<- r
    }
```

leaderSO


```
  # report
    cat("\nStep",step,"\n")
    print(table(clust)); print(R); print(Ro-R); Ro <- R;
    print(p); print(sum(p)); flush.console()
    tim <- tim-1
    if(tim<1){
      ans <- readline("Times repeat = ")
      tim <- as.integer(ans); if (tim < 1)  break
    }
  # in the case of empty cluster put in the most distant SO
    repeat{
      t <- table(clust); em <- setdiff(1:maxL,as.integer(names(t)))
      if(length(em)==0) break
      j <- em[[1]]; rmax <- 0; imax <- 0
      for(i in 1:numSO){d <- double(maxL)
        for(k in 1:maxL){d[[k]] <- distSO(SOs[[i]],L[[k]],nVar,alpha)}
        r <- max(d);  if(rmax<r) {rmax <- r; imax <- i}
      }
      clust[[imax]] <- j; L[[j]] <- SOs[[imax]]
      cat("*** empty cluster",j,"- SO",imax,"transfered, rmax =",rmax,"\n")
      flush.console()
    }
  }
  #detach(dataset)
  leaders <- dataset
  leaders$SOs <- L
  return (list(clust=clust,leaders_symData=leaders,R=R,p=p))
}
```

hclustSO


```
hclustSO <- function(dataset){
# compute order of dendrogram
  orDendro <- function(i){if(i<0) return(-i)
    return(c(orDendro(m[i,1]),orDendro(m[i,2])))}
  #attach(dataset)
  so <- dataset$so; alpha <- dataset$alpha
  L <- dataset$SOs; nVar <- length(so)
  numL <- length(L); numLm <- numL-1
# each unit is a cluster; compute inter-cluster dissimilarity matrix
  D <- matrix(nrow=numL,ncol=numL); diag(D) <- Inf
  for(i in 1:numLm) for(j in (i+1):numL) {
      D[i,j] <- distCl(L[[i]],L[[j]],nVar,alpha); D[j,i] <- D[i,j]
  }
  active <- 1:numL; m <- matrix(nrow=numLm,ncol=2)
  node <- rep(0,numL); h <- numeric(numLm); LL <- vector("list",numLm)
  for(k in 1:numLm) LL[[k]] <- so
  for(k in 1:numLm){
  # determine the closest pair of clusters (p,q)
    ind <- active[sapply(active,function(i) which.min(D[i,active]))]
    dd <- sapply(active,function(i) min(D[i,active]))
    pq <- which.min(dd)
  # join the closest pair of clusters
    p<-active[pq]; q <- ind[pq]; h[k] <- D[p,q]
    if(node[p]==0){m[k,1] <- -p; Lp <- L[[p]]
      } else {m[k,1] <- node[p]; Lp <- LL[[node[p]]]}
    if(node[q]==0){m[k,2] <- -q; Lq <- L[[q]]
      } else {m[k,2] <- node[q]; Lq <- LL[[node[q]]]}
    for(t in 1:nVar) LL[[k]][[t]] <- Lp[[t]] + Lq[[t]]
    active <- setdiff(active,p)
    Lpq <- LL[[k]]
```

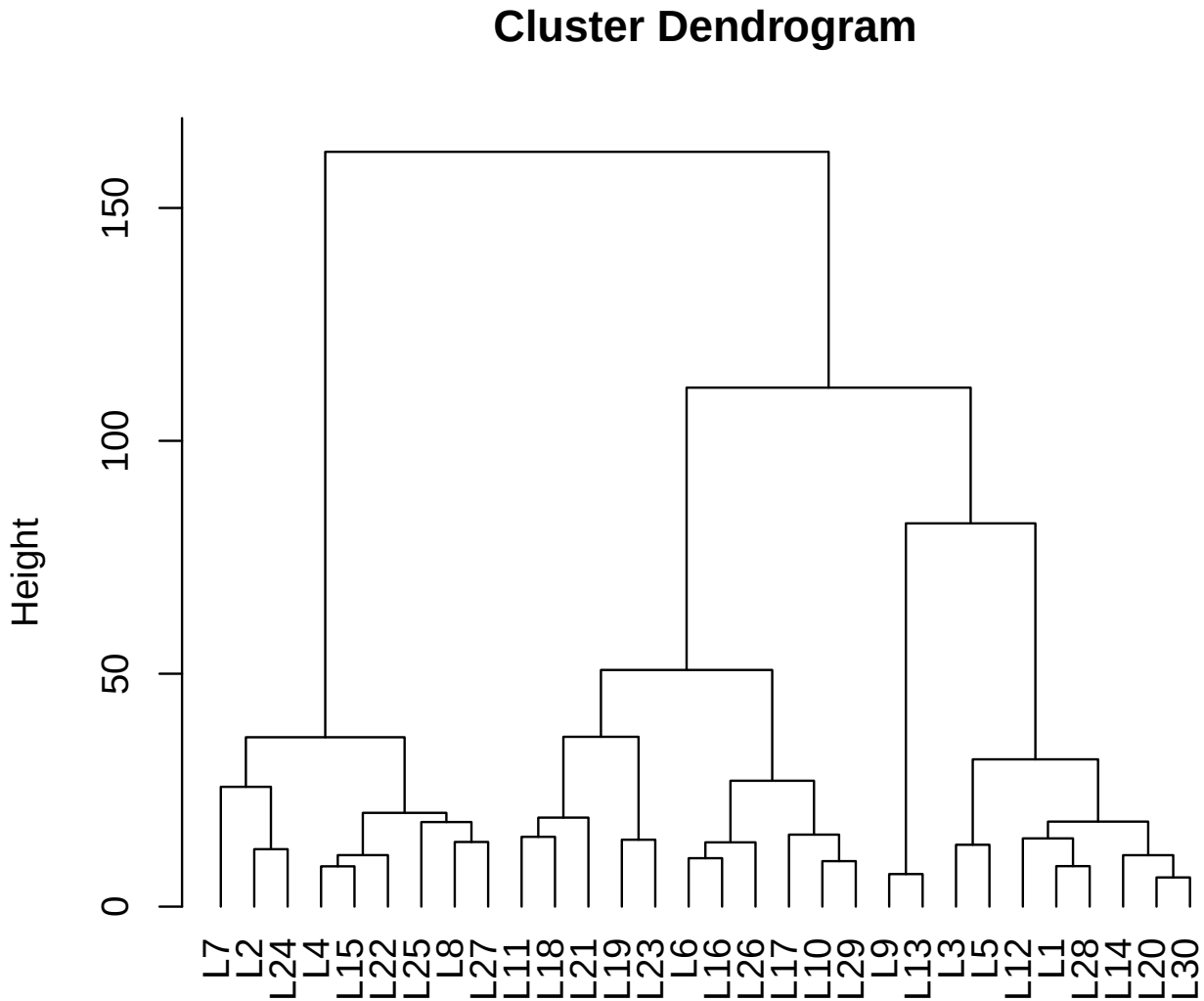
hclustSO


```
  # determine dissimilarities to the new cluster
    for(s in setdiff(active,q)){
      if(node[s]==0){Ls <- L[[s]]} else {Ls <- LL[[node[s]]]}
      D[q,s] <- distCl(Lpq,Ls,nVar,alpha); D[s,q] <- D[q,s]
    }
    node[q] <- k
  }
  hc <- list(merge=m,height=h,order=orDendro(numLm),labels=names(L),
    method="adapted ward",call=match.call(),dist.method="squared euclidean",leaders=LL)
  class(hc) <- "hclust"
  #detach(dataset)
  return(hc)
}
```

Application


```
> setwd("C:/.../clamix/clamix.R"); source("C:\\...\\clamix.R\\Clamix.R")
> load("./sr14/sr14.so"); objects()
[1] "dat"      "distSO"   "emptySO"  "encodeOS" "numSO"    "nVar"     "nVarP"
[8] "so"       "SOs"
> w <- rep(1/nVar,nVar)
> rez <- leaderSO(SOs,30,w,so)

Step 1
clust
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24
132 244 232 263 249 589 752 193 273  57 220 235 139  80  81 530  32 201 357  36 185  74 124 205
 28  29  30
178  25 148
 [1] 0.4466406 0.4443925 0.4568250 0.4571872 0.4660656 0.4721637 0.4488782 0.4399954 0.4803863 0
.......
[21]  77.52889  31.03541  50.96077  80.89709  41.74524   8.06992  35.82662  73.81336  10.05692
[1] 2483.908
Times repeat = 3
........

Step 42
clust
  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24
116 179 367 215 299 186 275 122 324 255 179 165 165 168 206 276 221  57 308 105  73 161 290 126
 28  29  30
200 185  93
 [1] 0.3961897 0.4226399 0.4669776 0.4373342 0.4699742 0.4520931 0.4155581 0.3473073 0.4438606 0
[11] 0.4576676 0.4220990 0.3943517 0.4682425 0.4087065 0.4355381 0.4578319 0.3822814 0.4319199 0
[21] 0.2938638 0.4417081 0.4029236 0.2948455 0.4391676 0.4188643 0.4375495 0.4436210 0.4508061 0
 [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 [1]  27.622636  37.895837 138.264393  64.037659 106.219657  61.910683  56.140880  28.080116  82
[10]  74.473877  47.658137  44.289150  41.319062  53.270737  67.129032  95.597943  71.012699   6
[19]  79.517072  27.804608   5.580203  55.304348  69.345717  19.494624  77.636606  37.387824 112
[28]  59.985484  59.407323  26.572320
[1] 1735.163
Times repeat = 0
> save(rez,file="sr14.rez")
> hc <- hclustSO(rez$leaders,w)
> plot(hc, hang = -1)
```

Dendrogram



Analysis of results

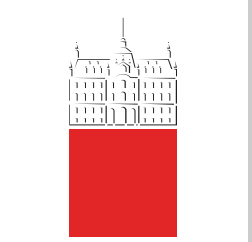

```
> long[rez$clust==7]
  [1] "LUNCHEON MEAT,BF,THIN SLICED"
  [2] "BEEF,COMP OF RTL CUTS,LN,1/4\"FAT,ALL GRDS,CKD"
  [3] "BEEF,COMP OF RTL CUTS,LN,1/4\"FAT,CHOIC,CKD"
  [4] "BEEF,COMP OF RTL CUTS,LN,1/4\"FAT,SEL,CKD"
  [5] "BEEF,COMP OF RTL CUTS,LN,1/4\"FAT,PRIME,CKD"
  [6] "BEEF,BRISKET,WHL,LN,1/4\"FAT,ALL GRDS,CKD,BRSD"
  [7] "BEEF,BRISKET,FLAT HALF,LN,1/4\"FAT,ALL GRDS,CKD,BRSD"
  [8] "BEEF,BRISKET,POINT HALF,LN,1/4\"FAT,ALL GRDS,CKD,BRSD"
  [9] "BEEF,CHUCK,ARM POT RST,LN,1/4\"FAT,ALL GRDS,CKD,BRSD"
 [10] "BEEF,CHUCK,ARM POT RST,LN,1/4\"FAT,CHOIC,CKD,BRSD"
  ....
 [43] "BEEF,RIB,SML END (RIBS 10-12),LN,1/4\"FAT,PRIME,CKD,RSTD"
 [44] "BEEF,RIB,SHORTRIBS,LN,CHOIC,CKD,BRSD"
 [45] "BEEF,RND,FULL CUT,LN&FAT,1/4\"FAT,CHOIC,CKD,BRLD"
 [46] "BEEF,RND,FULL CUT,LN&FAT,1/4\"FAT,SEL,CKD,BRLD"
  ..........
 [79] "BEEF,SHANK CROSSCUTS,LN,1/4\"FAT,CHOIC,CKD,SIMMRD"
 [80] "BEEF,SHRT LOIN,PRTRHS STEAK,LN,1/4\"FAT,CHOIC,CKD,BRLD"
 [81] "BEEF,SHRT LOIN,T-BONE STEAK,LN,1/4\"FAT,CHOIC,CKD,BRLD"
 [82] "BEEF,TENDERLOIN,LN&FAT,1/4\"FAT,CHOIC,CKD,RSTD"
 [83] "BEEF,TENDERLOIN,LN&FAT,1/4\"FAT,SEL,CKD,BRLD"
  .....
[102] "BEEF,TOP SIRLOIN,LN,1/4\"FAT,CHOIC,CKD,BRLD"
[103] "BEEF,TOP SIRLOIN,LN,1/4\"FAT,CHOIC,CKD,PAN-FRIED"
[104] "BEEF,TOP SIRLOIN,LN,1/4\"FAT,SEL,CKD,BRLD"
[105] "BEEF,GROUND,EX LN,CKD,BKD,WELL DONE"
[106] "BEEF,GROUND,EX LN,CKD,BRLD,MED"
[107] "BEEF,GROUND,EX LN,CKD,BRLD,WELL DONE"
  .........
[271] "BEEF,TOP SIRLOIN,LN&FAT,1/8\"FAT,CHOIC,CKD,PAN-FRIED"
[272] "BEEF,TOP SIRLOIN,LN&FAT,1/8\"FAT,SEL,CKD,BRLD"
[273] "LAMB,DOM,COMP OF RTL CUTS,LN,1/4\"FAT,CHOIC,CKD"
[274] "BEEF,SHRT LOIN,T-BONE STEAK,LN&FAT,1/8\"FAT,ALL GRDS,CKD,BRLD"
[275] "BEEF,SHRT LOIN,T-BONE STEAK,LN&FAT,1/8\"FAT,SEL,CKD,BRLD"
```

Analysis of results

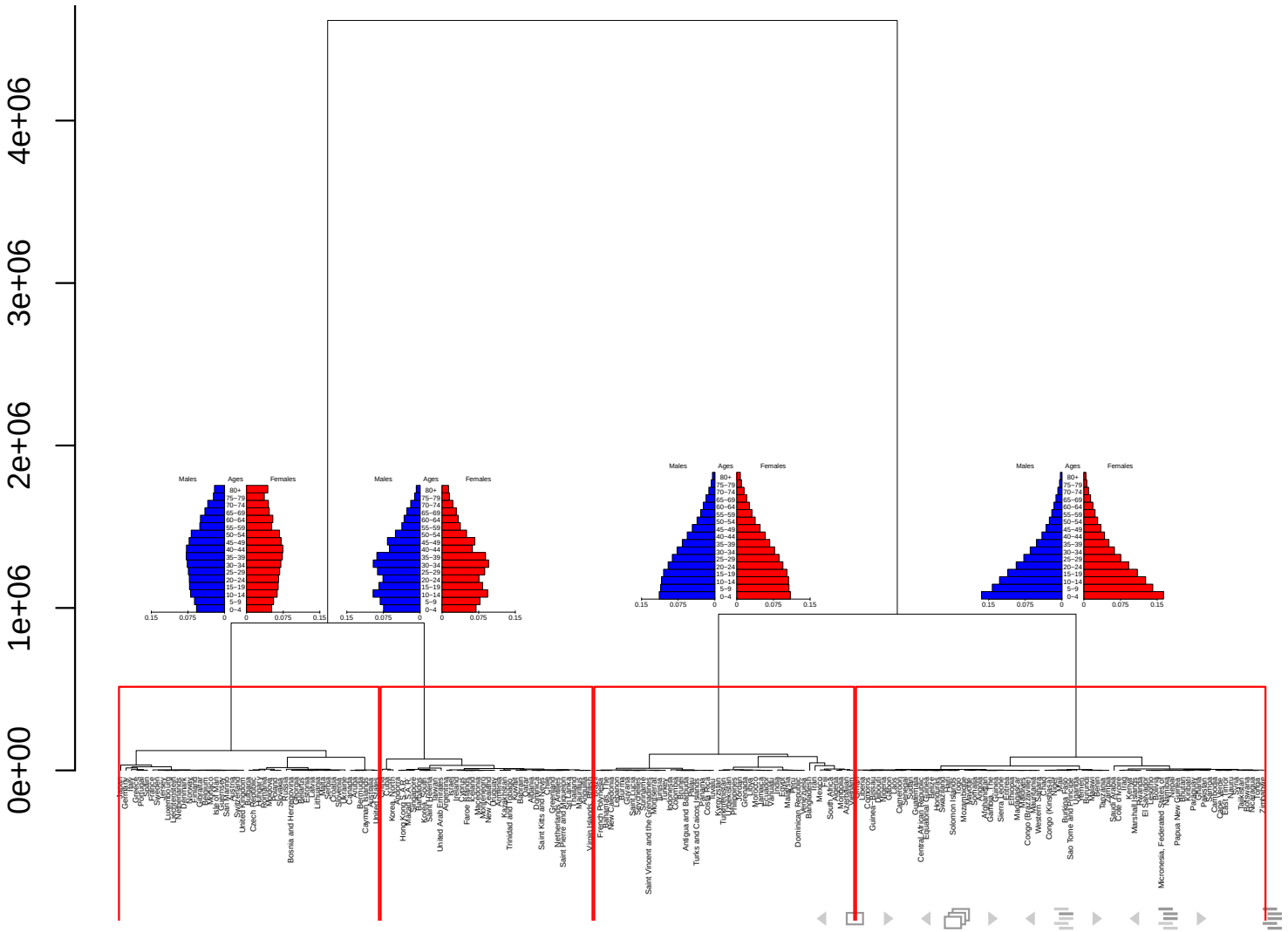

```
total <- namedSO
for(i in 1:numL) for(j in 1:nVarP) total[j] <- total[j] + L[[i]][[j]]


testSOvarP <- function(SO,var,total,a){
  pt <- total[var]/total$num
  pj <- SO[[var]]/SO$num
  te <- qnorm(1-a/2)*sqrt(pt*(1-pt)/SO$num)
  dif <- pj - pt; test <- abs(dif) > te
  for(k in 1:length(pt)) if(test[k]) {q <- pj[k]/pt[k]
    if(q > 1) cat(format(names(test)[k],width=15,justify="right"),
      format(c(pj[k],pt[k],q),
      width=12,justify="left",digits=5,nsmall=7),"\n")
  }
}
```

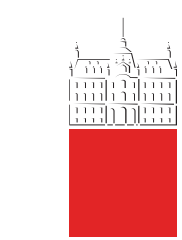
Analysis of results


```
> for(v in 1:nVar) {
+  cat("\n",v,". ",names(total)[[v]],"\n",sep="")
+  testSOvarP(L[[7]],v,total,0.001)}

1. Water
    [53.9,62.4)    0.7563636    0.1240272    6.0983712
2. Energ_Kcal
      (160,232]    0.5709091    0.1425733    4.0043206
      (232,312]    0.4036364    0.1400894    2.8812766
3. Protein
     [23.05,37]    1.0000000    0.1599603    6.2515528
4. Tot_Lipid
         [3.8,8)    0.2109091    0.1390959    1.5162857
        [8,13.7)    0.4654545    0.1394271    3.3383373
    [13.7,23.5)    0.3054545    0.1392615    2.1933888
5. Carbohydrt
             [0]    0.9890909    0.2702434    3.6600000
6. Fiber_TD
             [0]    1.0000000    0.3957609    2.5267782

  .........
30. FA_Poly
  (0.115,0.335]    0.3309091    0.1563173    2.1169068
  (0.335,0.685]    0.5854545    0.1549925    3.7773077
31. Cholestrl
        (66,80]    0.3018182    0.0816360    3.6971197
        (80,94]    0.4363636    0.0864382    5.0482759
       (94,550]    0.1818182    0.0796489    2.2827443
```

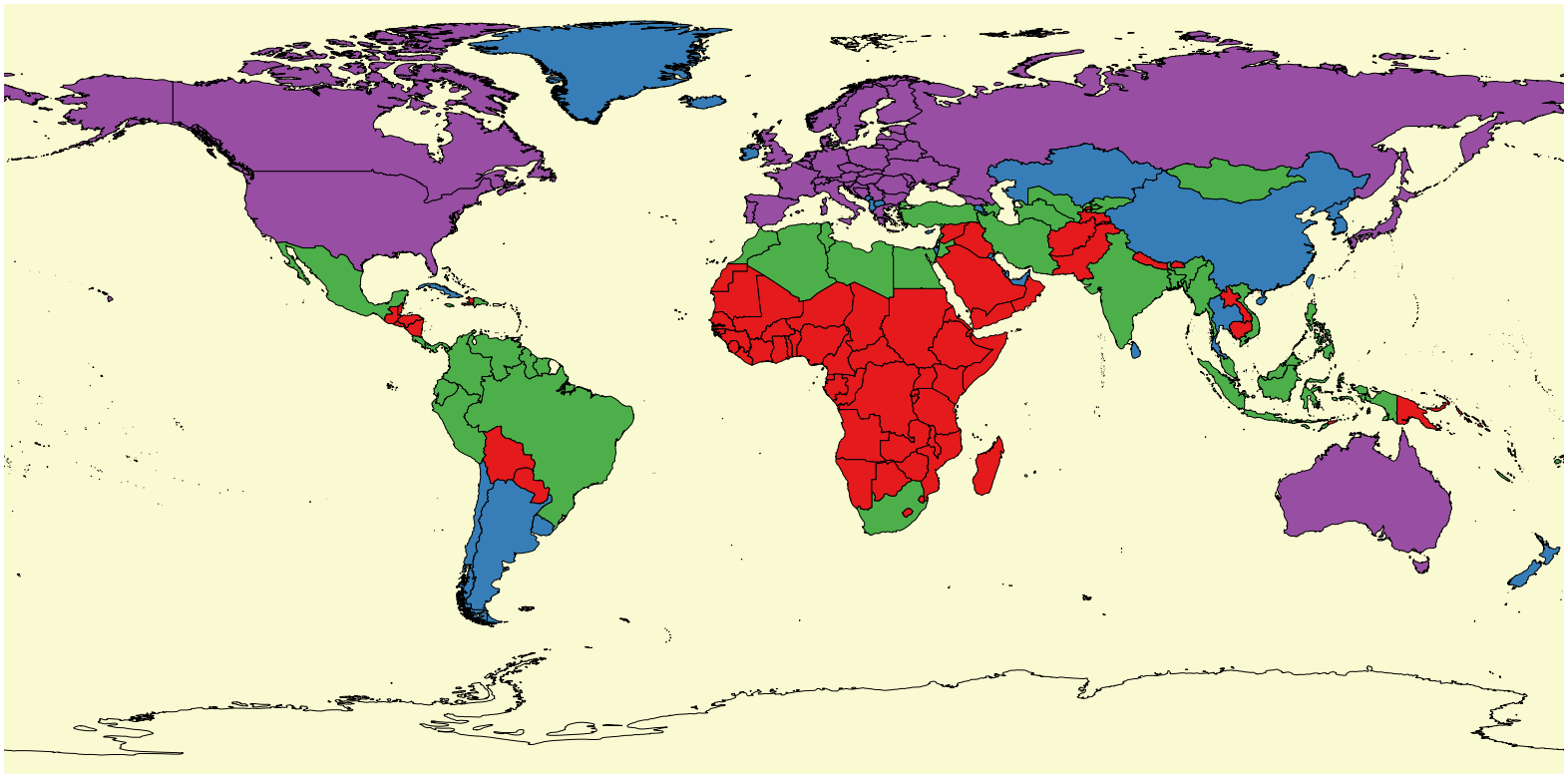


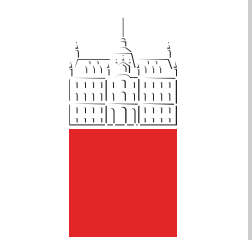
Population pyramids / World 2001



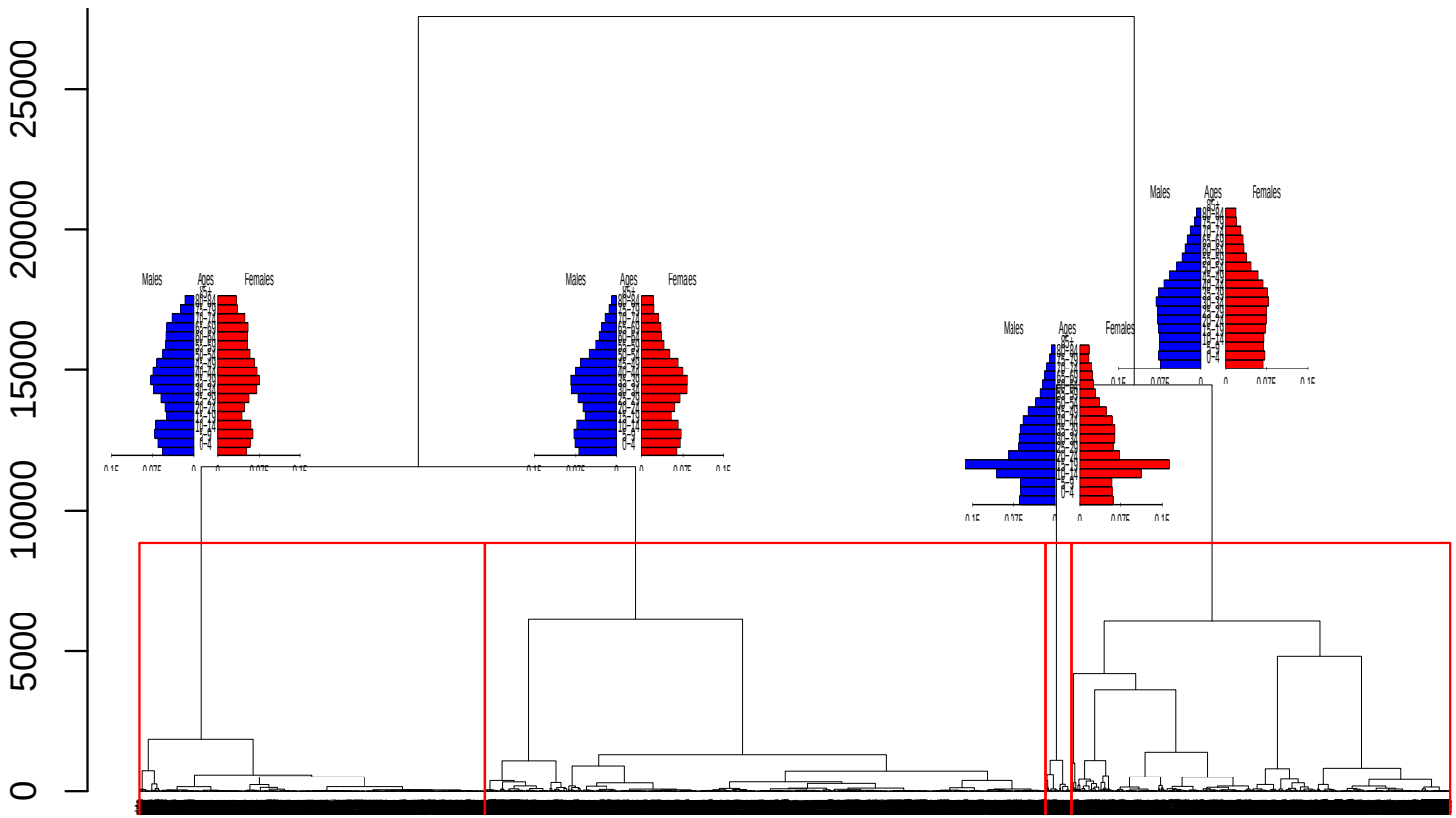


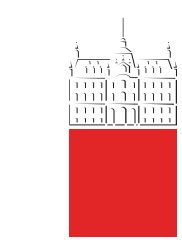
World 2001 / 4 clusters on the map



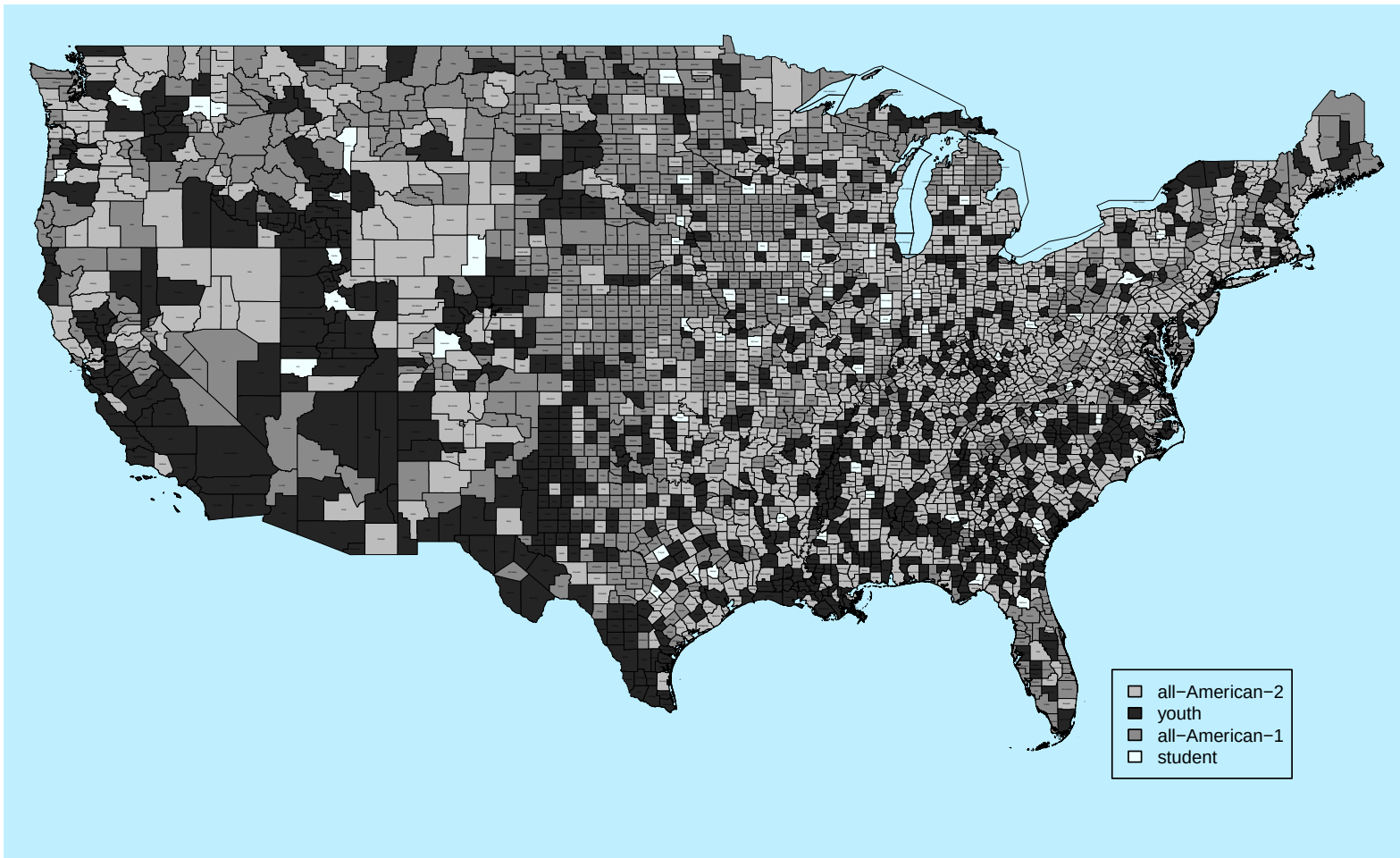


Population pyramids / US counties

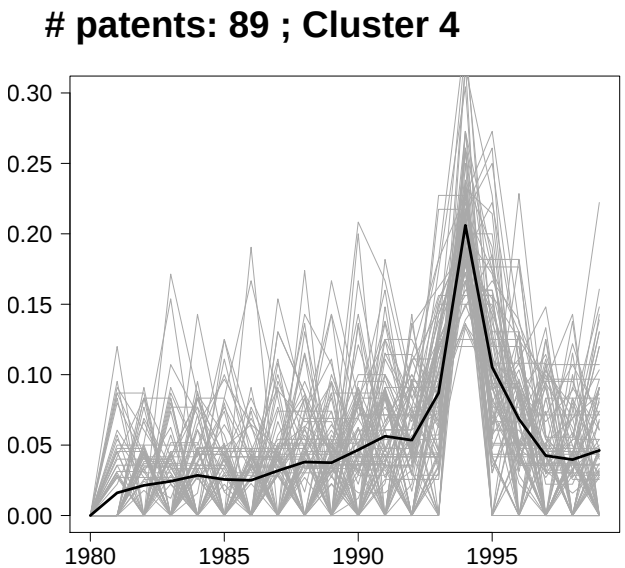


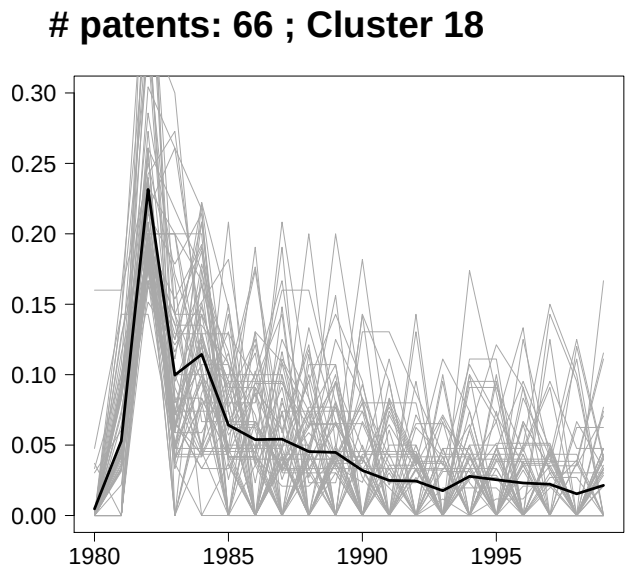


US counties map



k-means clusters of patents with patents' temporal distributions and cluster leaders





Patents clusters for δ_3

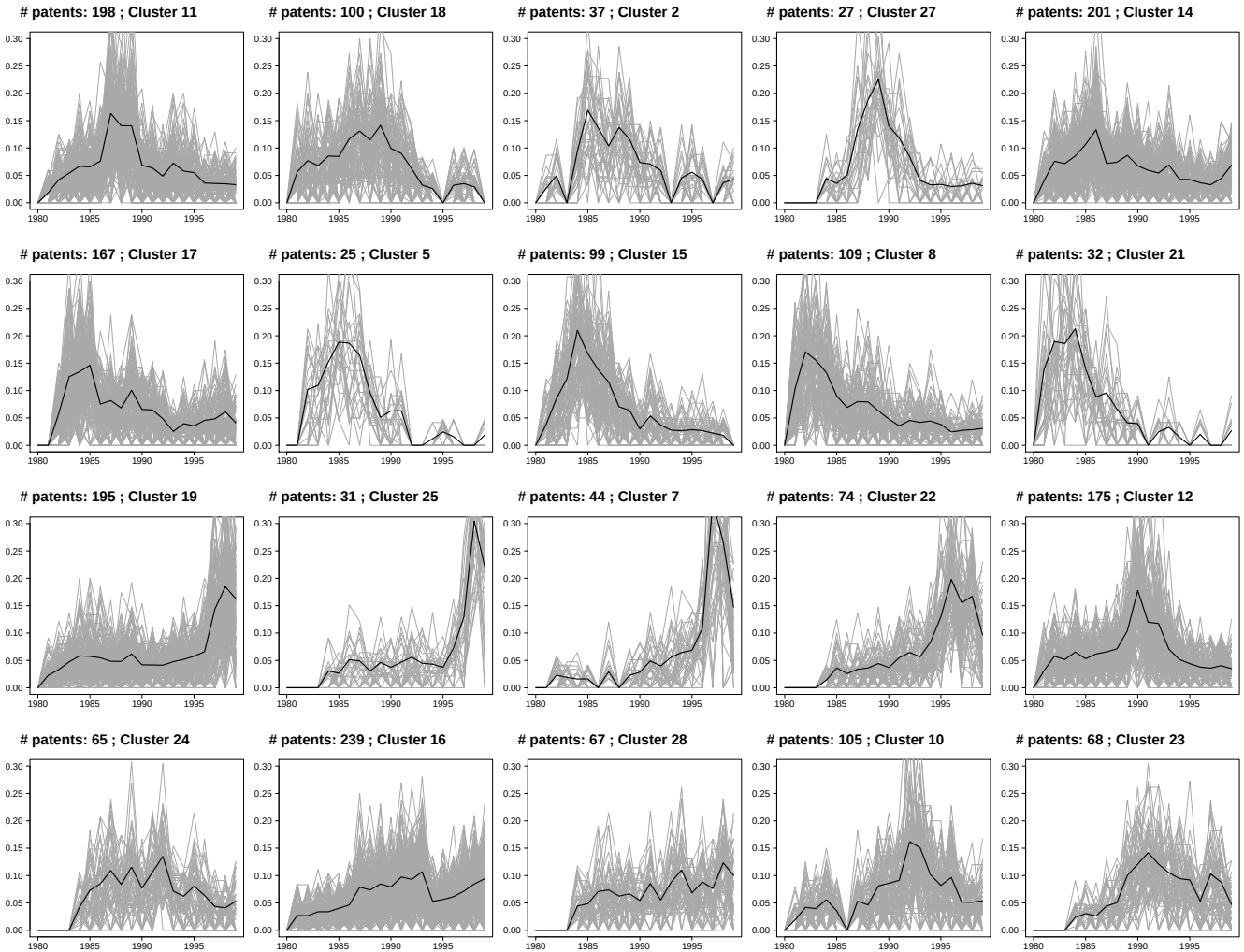
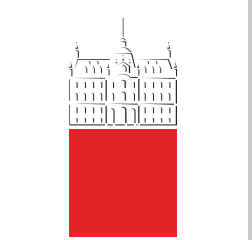
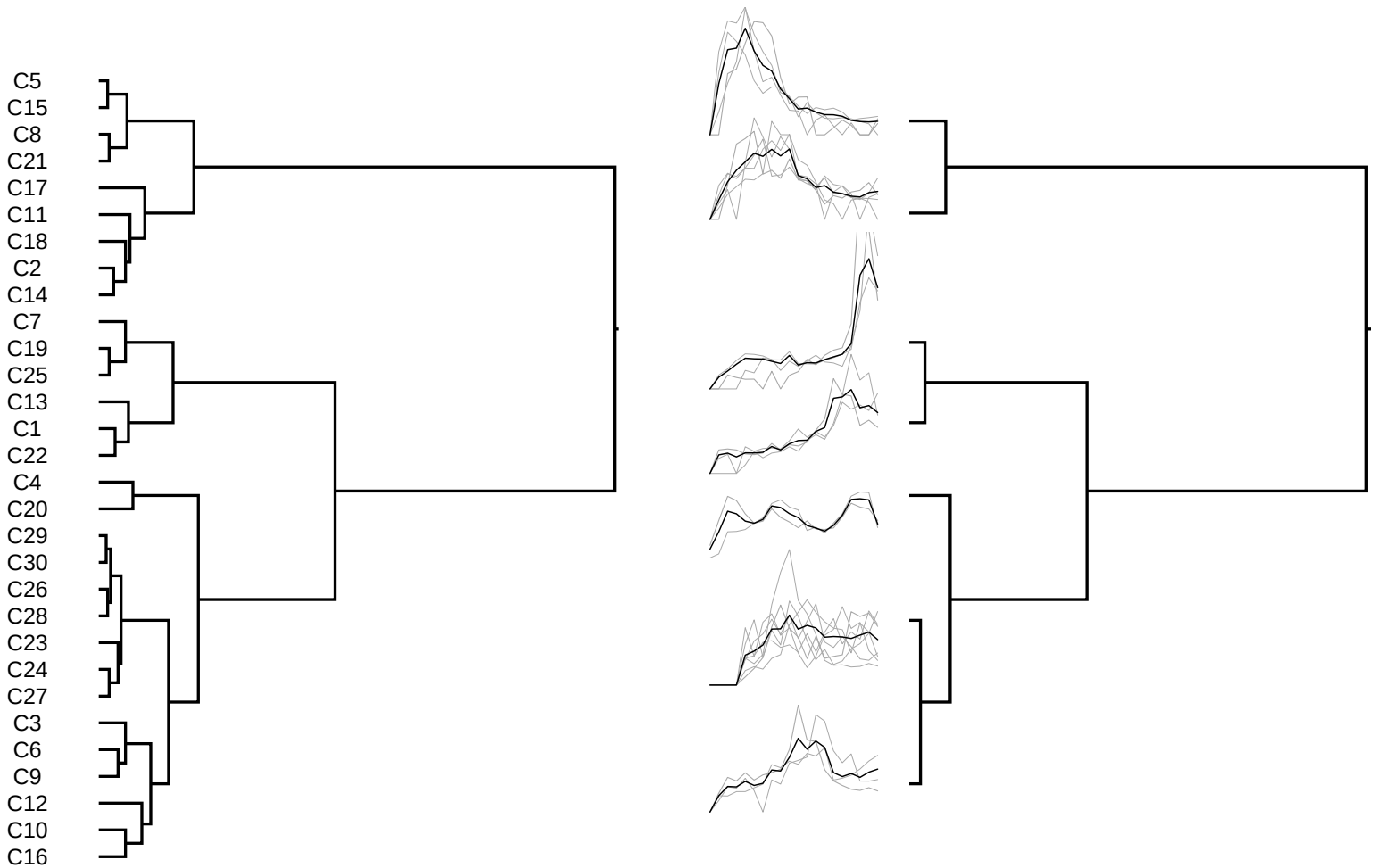


Fig. 2. Citations 20+, leaders for δ_4 - 1st part



Patents / clustering of leaders



References I

Anderberg, M. R. (1973). *Cluster Analysis for Applications*. Academic Press: New York.

Batagelj, V. (1988). *Generalized Ward and Related Clustering Problems*. Classification and Related Methods of Data Analysis. H.H. Bock (editor). North-Holland, Amsterdam, 1988. p. 67-74.

Billard, L., Diday, E. (2006). *Symbolic data analysis. Conceptual statistics and data mining*. Wiley: New York.

Diday, E. (1979). *Optimisation en classification automatique*, Tome 1.,2.. INRIA, Rocquencourt (in French).

Gan, G., Ma, C., Wu, J. (2007). *Data Clustering: Theory, Algorithms, and Applications*. SIAM - Society for Industrial Mathematics: Philadelphia.

Hartigan, J. A. (1975). *Clustering algorithms*, Wiley-Interscience: New York.

Jain, A.K., Murty, M.N., Flynn, P.J. (1999). "Data clustering: a review". *ACM Computing surveys*, 31, 264–323.

References II

Japelj Pavešić, B., Korenjak-Černe, S. (2004). "Differences in teaching and learning mathematics in classes over the world : the application of adapted leaders clustering method". In *Proceedings of the IRC - 2004 : IEA International Research Conference*. Nicosia: University of Cyprus, Department of Education, 2004, p. 85–101.

Kaufman, L., Rousseeuw, P. (1990). *Finding groups in data: an introduction to cluster analysis*. John Wiley & Sons: New York.

Kejžar, N., Korenjak-Černe, S., Batagelj, V. (2011). "Clustering of discrete distributions: A case of patent citations". J. classif., vol. 28, no. 2, p. 156-183.

Korenjak-Černe, S., Kejžar, N., Batagelj, V. (2008). Clustering of population pyramids. Informatica, vol. 32, no. 2, p. 157-167,

Korenjak-Černe, S., Batagelj, V., Japelj Pavešić, B. (2011). Clustering large data sets described with discrete distributions and its application on TIMSS data set. Stat. anal. data min., vol. 4, iss. 2, p. 199-215.

References III

Korenjak-Černe, S., Kogovšek, T., Batagelj, V. (2000). Clustering ego-centered networks. In: Blasius, Jörg (ed.). Social science methodology in the new millennium: proceedings of the Fifth International Conference on Logic and Methodology. Cologne: TT-Publikaties, 4 pages.

Korenjak-Černe, S., Batagelj, V. (1998). Clustering large datasets of mixed units. In: Rizzi, A., Vichi, M., Bock, H-H. (eds.). 6th Conference of the International Federation of Classification Societies (IFCS-98) Universitá "La Sapienza", Rome, 21-24 July, 1998. Advances in data science and classification. Berlin: Springer, p. 43-48.

Korenjak-Černe, S., Batagelj, V. (2002). Symbolic data analysis approach to clustering large datasets. In: Jajuga, K., Sokołowski, A., Bock, H-H. (eds.). 8th Conference of the International Federation of Classification Societies, July 16-19, 2002, Cracow, Poland. Classification, clustering and data analysis. Berlin: Springer, p. 319-327.

Ward, J. H. (1963). "Hierarchical grouping to optimize an objective function", *Journal of the American Statistical Association, 58*, 236–244.