Grouping techniques for facing Volume and Velocity in Big Data

How to do it using HistDAWass package for clustering Histogram-valued data

Antonio Irpino, PhD

University of Campania "L. Vanvitelli"
Dept. of Mathematics and Physics
Caserta, Italy
antonio.irpino@unicampania.it

June, 4th, 2018

A very short introduction on some aspects of Big Data

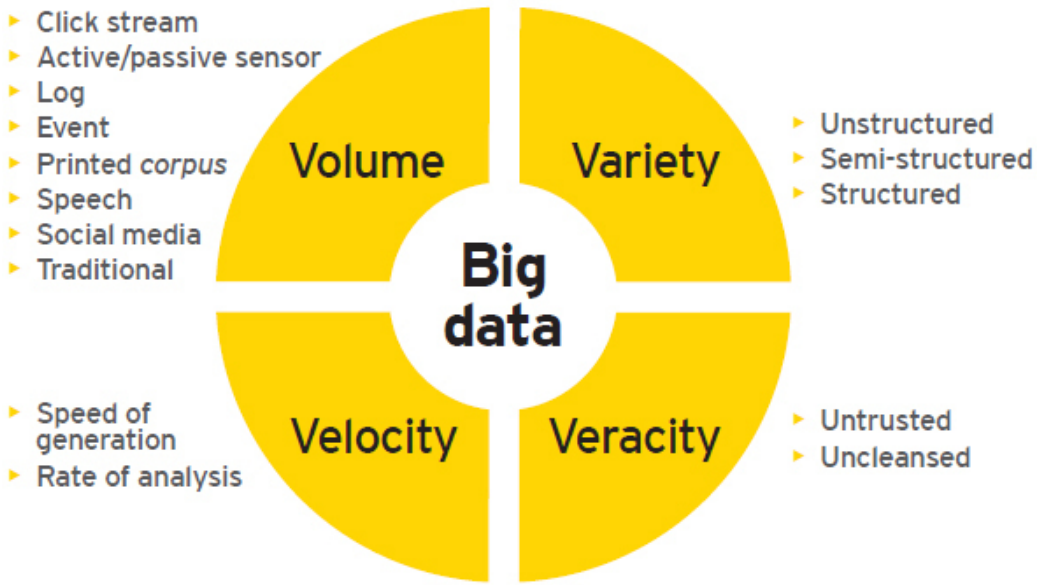
Some Big data properties

From Wikipedia:

"Big data is data sets that are so voluminous and complex that traditional data-processing application software are inadequate to deal with them."

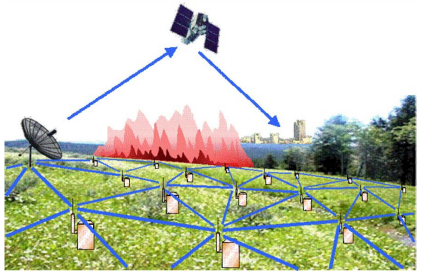
Big data can be described by the following characteristics:

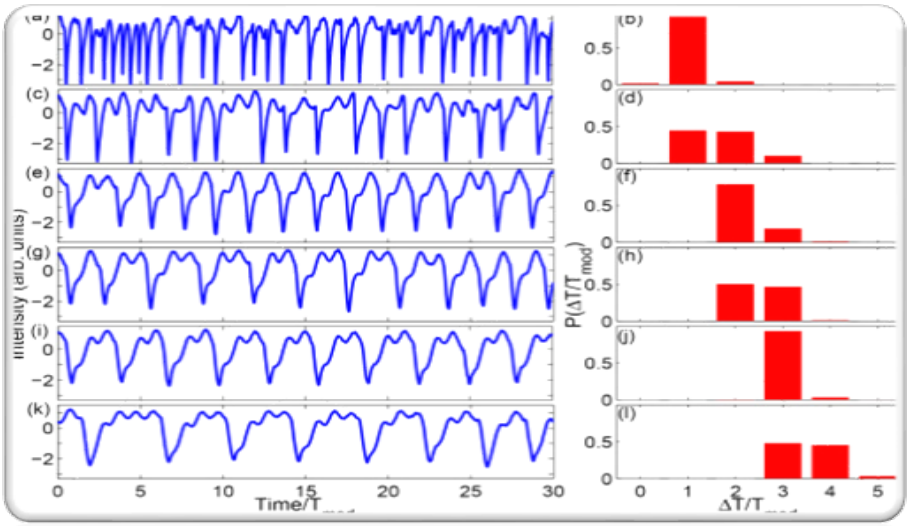
- **Volume** The quantity of generated and stored data. The size of the data determines the value and potential insight, and whether it can be considered big data or not.
- **Variety** The type and nature of the data. This helps people who analyze it to effectively use the resulting insight. Big data draws from text, images, audio, video; plus it completes missing pieces through data fusion.
- **Velocity** In this context, the speed at which the data is generated and processed to meet the demands and challenges that lie in the path of growth and development. Big data is often available in real-time.
- **Veracity** The data quality of captured data can vary greatly, affecting the accurate analysis.



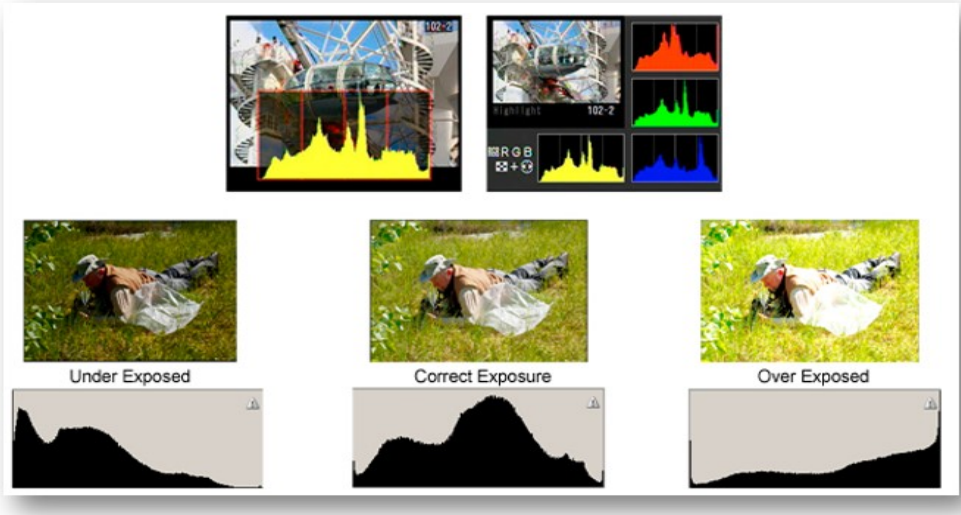
Facing Volume and Velocity

Example 1: a network of wireless sensors collecting and sharing data.





Example 2: features extracted by an image database.



A suggestion for analysing big data

Mizuta (2016) suggests to use **Mini Data** for the analysis of **Big Data**.

Mini Data

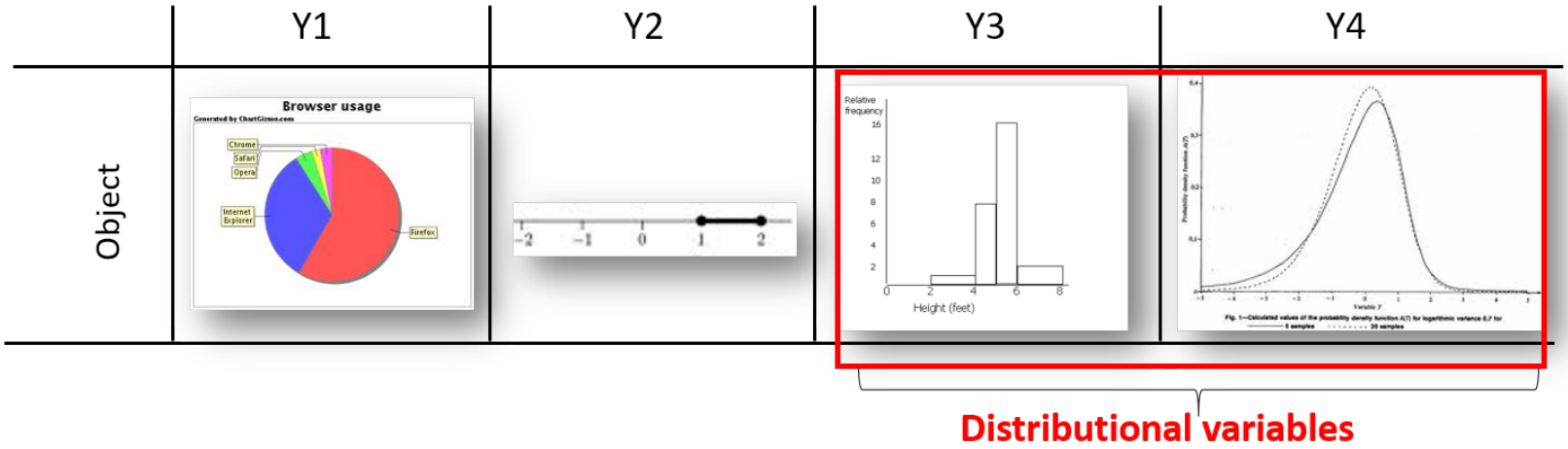
Mini data of big data are defined as data set which contains an important information about the big data, but its size and/or structure are realistic to deal with. For building Mini data some tools can be used: Sampling, Variable Selection, Dimension Reduction, Feature extraction and ...

Symbolization Symbolic Data Analysis (SDA) was proposed. Symbolic data are descripted with interval valued, distribution valued, combinations of them, or other complex structured values. The target object that are analyzed are called concepts. The concepts are typical examples of mini data.

A proposal for describing such new objects:

Symbolic Data Analysis and distributional data (Bock and Diday 2000)

The measurement done on an object for a variable may have several values: namely, data are, or might be, multi-valued.
Especially, if an object is an higher order statistical unit, namely, generalizes a set of individual measurements (a Region, a City, a market segment, a typology,. . .). But, it is not only this!



Concurrent approaches

- Functional data analysis (Data are functions!)
- Compositional data analysis (Compositions obey the Aitchison geometry!)
- Object oriented data analysis (Data live in particular spaces, which are not always Euclidean!)

A very short intro to clustering

What is clustering?

A **Clustering method** is an exploratory tool that looks for groups in data!

Clustering is widely used (Hennig 2015) for

- delimitation of species of plants or animals in biology,
- medical classification of diseases,
- discovery and segmentation of settlements and periods in archaeology,
- image segmentation and object recognition,
- social stratification,
- market segmentation,
- efficient organization of data bases for search queries.

There are also quite general tasks for which clustering is applied in many subject areas:

- exploratory data analysis looking for "interesting patterns" without prescribing any specific interpretation, potentially creating new research questions and hypotheses,
- information reduction and structuring of sets of entities from any subject area for simplification, effective communication, or effective access/action such as complexity reduction for further data analysis, or classification systems,
- investigating the correspondence of a clustering in specific data with other groupings or characteristics, either hypothesized or derived from other data

WOW! but... what is a **cluster**?

What are "true clusters"?

Hennig (2015) lists a set of ideal properties while doing (or validating) clustering:

1. Within-cluster dissimilarities should be small.
2. Between-cluster dissimilarities should be large.
3. Clusters should be fitted well by certain homogeneous probability models such as the Gaussian or a uniform distribution on a convex set, or by linear, time series or spatial process models.
4. Members of a cluster should be well represented by its centroid.
5. The dissimilarity matrix of the data should be well represented by the clustering (i.e., by the ultrametric induced by a dendrogram, or by defining a binary metric "in same cluster/in different clusters").
6. Clusters should be stable.
7. Clusters should correspond to connected areas in data space with high density.
8. The areas in data space corresponding to clusters should have certain characteristics (such as being convex or linear).
9. It should be possible to characterize the clusters using a small number of variables.
10. Clusters should correspond well to an externally given partition or values of one or more variables that were not used for computing the clustering.
11. Features should be approximately independent within clusters.
12. All clusters should have roughly the same size.
13. The number of clusters should be low

Types of clusterings

Considering the obtained partition:

1. Hard clustering (an object must belong to a single group)
2. Fuzzy or possibilistic clustering (an object belongs to a cluster accordingly to a membership degree)

Considering how data are aggregated

1. Partitive clustering
   1. K-means, K medoids, Dynamic clustering
   2. Density based clustering
   3. Model based clustering (Latent class modeling: e.g. Gaussian Mixtures Models)
2. Hierarchical clustering
   1. bottom-up (aggregating recursively objects)
   2. top-down (dividing the whole set recursively)

The most part of algorithms are based on the choice of a similarity/dissimilarity/distance between data

Distances for distributions

Abbreviation	Metric
D	Discrepancy
H	Hellinger distance
I	Relative entropy (or Kullback-Leibler divergence)
K	Kolmogorov (or Uniform) metric
L	Levi metric
P	Prokhorov metric
S	Separation distance
W	**Wasserstein** (or Kantorovich) metric
χ^2	χ^2 distance

The L_2^2 Wassertein distance is: $d_W^2(Y_i, Y_j) = \int_0^1 [Q_i(t) - Q_j(t)]^2 dt$

Where $Q_i(t)$ is a *quantile function* (namely, the inverse of th *Cumulative distribution function*). It has some nice properties in clustering (R. Verde and Irpino 2007) and basic statistics have been developed (Irpino and Verde 2015). Methods have been implemented in R in a package called HistDAWass (Histogram Data Analysis with Wasserstein distance).

Wasserstein distance: a nice property

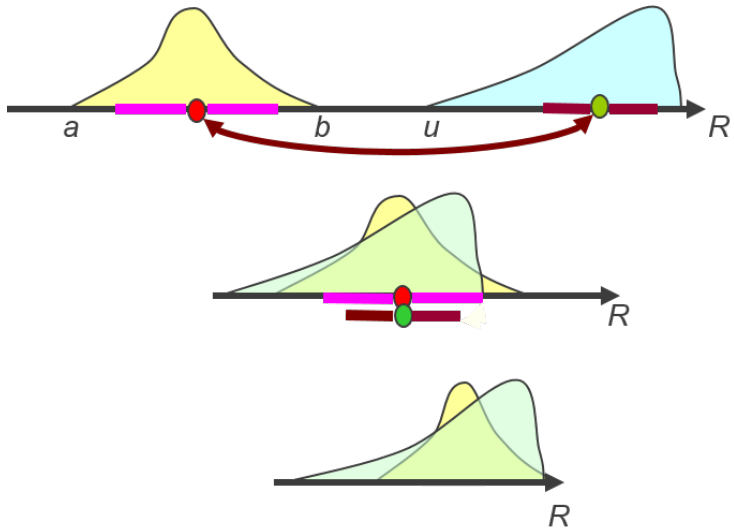
Total Distance = Position + Internal Variability
Internal Variability = Size + Shape

$$d_W^2(Y_i, Y_j) = \int_0^1 [Q_i(t) - Q_j(t)]^2 dt$$

$$= (\mu_i - \mu_j)^2 +$$

$$+ (\sigma_i - \sigma_j)^2 +$$

$$+ 2\sigma_i\sigma_j [1 - Corr_{QQ}(Y(i), Y(j))]$$



Hard-partitive algorithms

Dynamic clustering (a generalization of k-means algorithm)

The dynamic clustering algorithm: after initialization, a two-step algorithm looks for the best partition into k classes and the best representation of clusters.

We assume that the prototype of the cluster C_k $(k = 1, \ldots, K)$ is also represented by a vector $\mathbf{g}_k = (g_{k1}, \ldots, g_{kp})$, where g_{kj} is a histogram. DCA looks for the partition $P = (C_1, \ldots, C_k)$ of E in K clusters. The corresponding set of K prototypes $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_K)$ such that the following adequacy criterion of best fitting between the clusters and their prototypes is locally minimized:

$$\Delta(\mathbf{G}, P) = \sum_{k=1}^{K} \sum_{i \in C_k} d_W^2(\mathbf{y}_i, \mathbf{g}_k). \tag{1}$$

The Algorithm

The DCA algorithm

1. **Initialize the algorithm**
   1. Set a number k of clusters
   2. Set $T = 0$
   3. Generate a random partition of the objects $P(0)$
   4. Compute the criterion (the Within-cluster sum of Squares), $CRIT(0)$
2. **Representation step**
   1. Set $T = T + 1$
   2. Compute the prototypes of each cluster using $P(T-1)$
3. **Allocation step**
   1. Allocate objects to the nearest prototype obtaining the partition $P(T)$
   2. Compute $CRIT(T)$
4. **STOP CONDITION**
   - If $CRIT(T) < CRIT(T-1)$ goto **step 2**, else return results.

The WH_kmeans function

The function uses L_2 Wasserstein-based statistics

```
results=WH_kmeans(x,                    # A MatH object
                  k,                    # The number of required clusters
                rep=5,                # How many time itis initialized
                simplify=FALSE,       # A flag for speeding up,
                                      # approximating data
                qua=10,               # If symplify=TRUE how many quantiles
                                      # are used for approximating the
                                      # distributions
                standardize=FALSE)    # Do you need to standardize variables?
```

The output of `WH_kmeans` function

- **`results`** A list. It contains the best solution among the repetitions, i.e. the one having the minimum criterion.
  - **`results$IDX`** A vector. The clusters at which the objects are assigned.
  - **`results$cardinality`** A vector. The size of each final cluster.
  - **`results$centers`** A MatH object with the description of centers.
  - **`results$Crit`** A number. The criterion (Within-cluster Sum of squared distances from the centers).
  - **`results$quality`** A number. The percentage of Total SS explained by the model. (The higher the better)

Adaptive distances-based dynamic clustering (A. Irpino, Verde, and De Carvalho 2014)

A system of weights are calculated for the variables, for their components, cluster-wise or globally. The system of weights is useful if data are clustered into non-spherical classes.

We assume that the prototype of the cluster C_k $(k = 1, \ldots, K)$ is also represented by a vector $\mathbf{g}_k = (g_{k1}, \ldots, g_{kp})$, where g_{kj} is a histogram. As in the standard adaptive DCA, the proposed methods look for the partition $P = (C_1, \ldots, C_k)$ of E in K clusters. The corresponding set of K prototypes $\mathbf{G} = (\mathbf{g}_1, \ldots, \mathbf{g}_K)$ and a set of K different adaptive distances $d = (d_1, \ldots, d_K)$ depend on a set Λ of positive weights associated with the clusters, such that the following adequacy criterion of the best fitting between the clusters and their prototypes is locally minimized:

$$\Delta(\mathbf{G}, \Lambda, P) = \sum_{k=1}^{K} \sum_{i \in C_k} d(\mathbf{y}_i, \mathbf{g}_k | \Lambda). \tag{2}$$

The adaptive distances

One weight for each variable

$$d(\mathbf{y}_i, \mathbf{g}_k|\Lambda) = \sum_{j=1}^{p} \lambda_j d_W^2(y_{ij}, g_{kj}) \tag{3}$$

Two weights for each variable (one for each component of the distance)

$$d(\mathbf{y}_i, \mathbf{g}_k|\Lambda) = \sum_{j=1}^{p} \lambda_{j,\bar{y}}(\bar{y}_{ij} - \bar{y}_{g_{kj}})^2 + \sum_{j=1}^{p} \lambda_{j,Disp} d_W^2(y_{ij}^c, g_{kj}^c) \tag{4}$$

One weight for each variable and each cluster

$$d(\mathbf{y}_i, \mathbf{g}_k|\Lambda) = \sum_{j=1}^{p} \lambda_j^k d_W^2(y_{ij}, g_{kj}) \tag{5}$$

Two weights for each variable and each cluster

$$d(\mathbf{y}_i, \mathbf{g}_k|\Lambda) = \sum_{j=1}^{p} \lambda_{j,\bar{y}}^k(\bar{y}_{ij} - \bar{y}_{g_{kj}})^2 + \sum_{j=1}^{p} \lambda_{j,Disp}^k d_W^2(y_{ij}^c, g_{kj}^c) \tag{6}$$

Two possible functions for computing the weights and four possible combinations of weights

The system of weights may be

- Multiplicative: the product of weights is fixed (generally equal to one)
- Additive: the sum of weights is fixed (generally equal to one)

Ways for assigning weights.

1. A weight for **each variable**
2. A weight for **each variable** and **each cluster**
3. A weight for **each component** of a distributional variable (we mean the *position* and the *variability* component related to the decomposition of the L_2 Wasserstein distance)
4. A weight for **each component** and **each cluster**

The algorithm

The Adaptive DCA algorithm

1. **Initialize the algorithm**
   1. Set $T = 0$, a number k of clusters, initialize weights $W(0)$.
   2. Generate a random partition of the objects $P(0)$
   3. Compute the criterion (the Within-cluster sum of Squares), $CRIT(0)$
2. **Representation step** (Fix the Partition and the Weights)
   1. Set $T = T + 1$. Compute the prototypes $G(T)$ of each cluster using $P(T-1)$ and $W(T-1)$.
3. **Weighting step** (Fix the Prototypes and the Weights)
   1. Compute the weight system $W(T)$ using $G(T)$ and $P(T-1)$
4. **Allocation step** (Fix the Weights and Prototypes)
   1. Assign objects to the nearest prototype in $G(T)$ using $W(T)$, obtaining the partition $P(T)$
   2. Compute $CRIT(T)$
5. **STOP CONDITION** If $CRIT(T) < CRIT(T-1)$ goto **step 2**, else return results.

The WH_adaptive_kmeans function

```
results= WH_adaptive.kmeans(x, k,
                    schema = 1, init, rep,
                    simplify = FALSE, qua = 10, standardize = FALSE,
                    weight.sys = "PROD", theta = 2, init.weights = "EQUAL")
```

Parameter	Description
x	A MatH object (a matrix of distributionH).
k	An integer, the number of groups.
schema	a number from 1 to 4: 1 A weight for each variable (default) 2 A weight for the average and the dispersion component of each variable 3 Same as 1 but a different set of weights for each cluster 4 Same as 2 but a different set of weights for each cluster
init	(optional, do not use) initialization for partitioning the data default is 'RPART'
rep	An integer, maximum number of repetitions of the algorithm (default rep=5).
simplify	A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
qua	An integer, if simplify=TRUE is the number of quantiles used for re-coding the histograms.
standardize	A logic value (default is FALSE). If TRUE, histogram-valued data are standardized,variable by variable, using the Wasserstein based standard deviation.
weight.sys	a string. Weights may add to one ('SUM') or their product is equal to 1 ('PROD', default).
theta	a number. A parameter if weight.sys='SUM', default is 2.
init.weights	a string how to initialize weights: 'EQUAL' (default), all weights are the same, 'RANDOM', weights are initialized at random.

The output

Name	description
results	A list.Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares criterion.
results$IDX	A vector. The final clusters labels of the objects.
results$cardinality	A vector. The cardinality of each final cluster.
results$proto	A MatH object with the description of centers.
results$weights	A matrix of weights for each component of each variable and each cluster.
results$Crit	A number. The criterion (Weighted Within-cluster SS) value at the end of the run.
results$TOTSSQ	The total SSQ computed with the system of weights.
results$BSQ	The Between-clusters SSQ computed with the system of weights.
results$WSQ	The Within-clusters SSQ computed with the system of weights.
results$quality	A number. The proportion of TSS explained by the model. (The higher the better)

An application on a temperature dataset of USA

In this example, we use data of mean monthly temperatures observed in 48 states of US. Raw data are free available at the National Climatic Data Center website of US (http://www1.ncdc.noaa.gov/pub/data/cirs/). The original dataset `drd964x.tmpst.txt` contains the sequential *Time Biased Corrected* state climatic division monthly Average Temperatures recorded in the 48 (Hawaii and Alaska are not present in the dataset) states of US from 1895 to 2014.

R code about this example is available here

First of all you can access to data and R scripts from this link. USA_GIS (https://www.dropbox.com/sh/c21stseobdroub7/AAABVZzDR0k2ZPvT2eSleNova?dl=0)

A sketch of the data

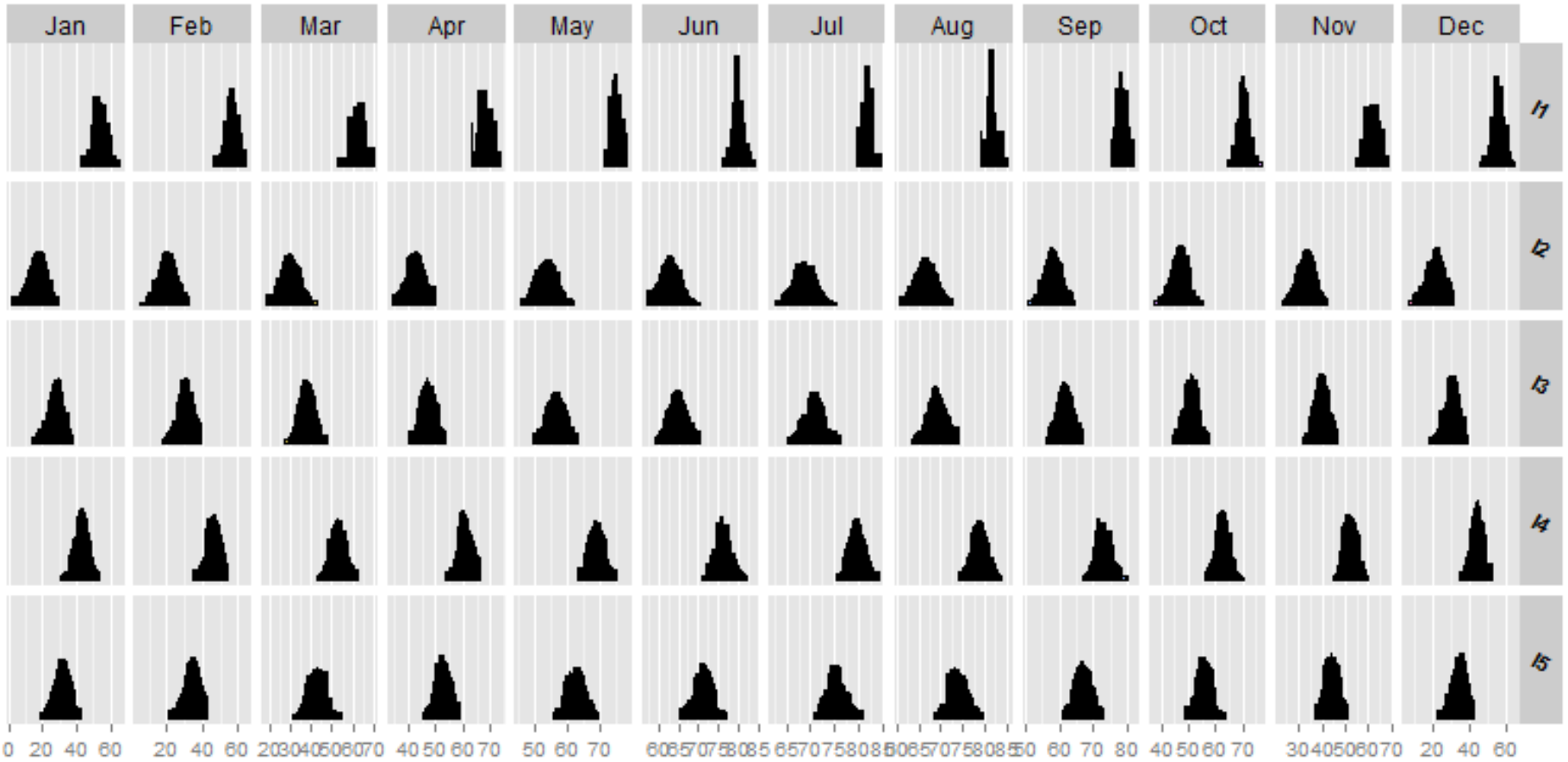
```
load("USA_TMP.RData")
plot(USA_TMP_MAT)
```

Performing a DCA (Partitive model)

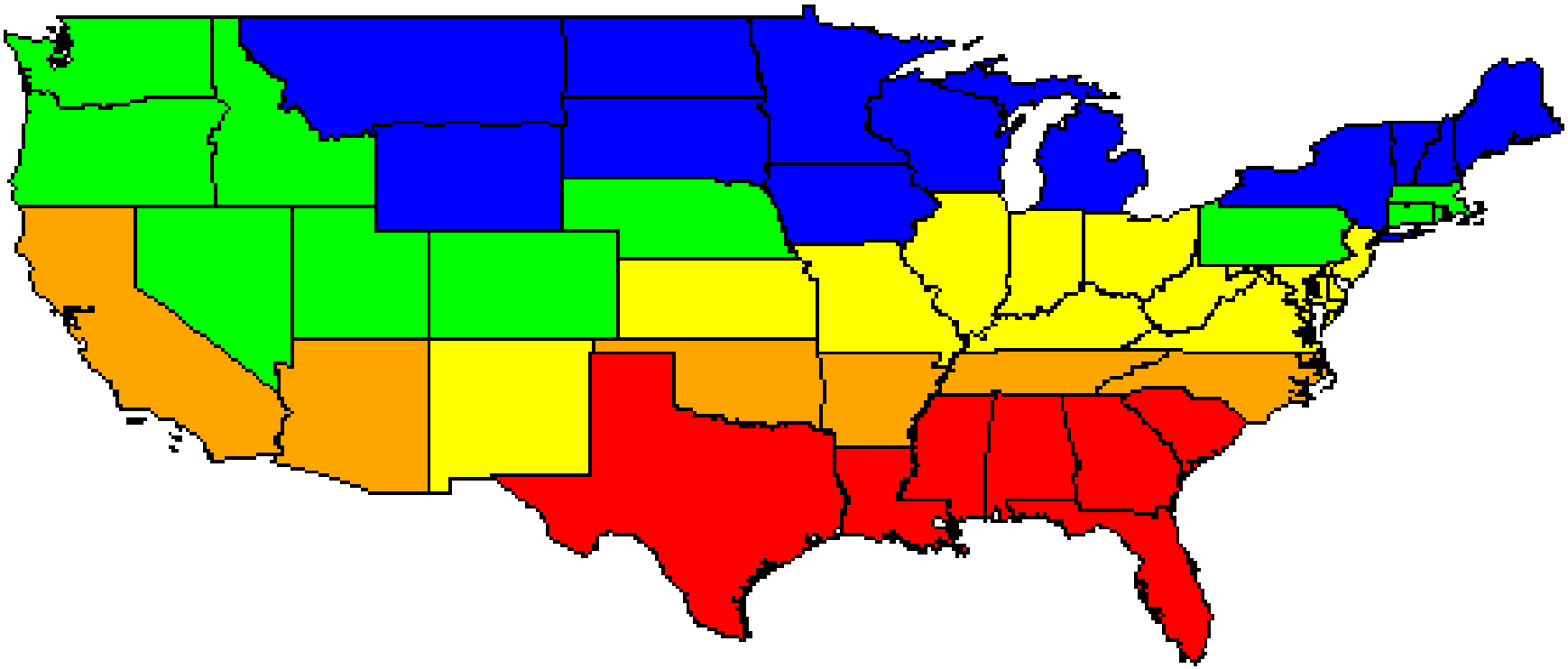
```
DCA.5k.resu=WH_kmeans(USA_TMP_MAT,k = 5,rep = 20)
# we consider the best result among 20 runs
$solution$cardinality
IDX
Cl.1 Cl.2 Cl.3 Cl.4 Cl.5
   6   11   12    7   12
$solution$centers
a matrix of distributions
 12  variables  5  rows
....
$solution$Crit
[1] 3860.546
$quality
[1] 0.9015709
```

DCA on USA_TMP_MAT: the prototypes

```
Cl.1 Cl.2 Cl.3 Cl.4 Cl.5
   6   11   12    7   12
$solution$Crit  $quality
3860.546        0.9015709
```



DCA on USA k=5: the map



What is the best (or a suitable) number of clusters

The Kalinski-Harabastz index (pseurdo F score)

- n number of objects
- k number of clusters

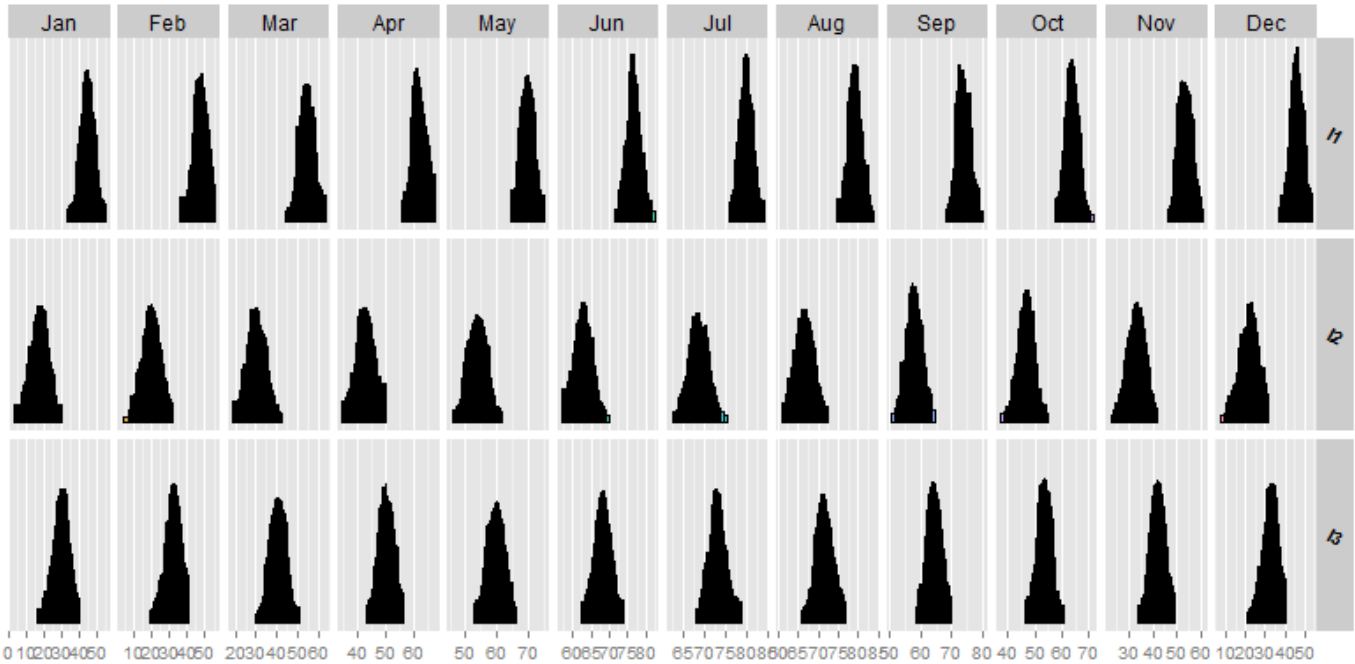
$$CH(k) = \frac{BSS/(n-k)}{WSS/(k-1)}$$

The highest the best!

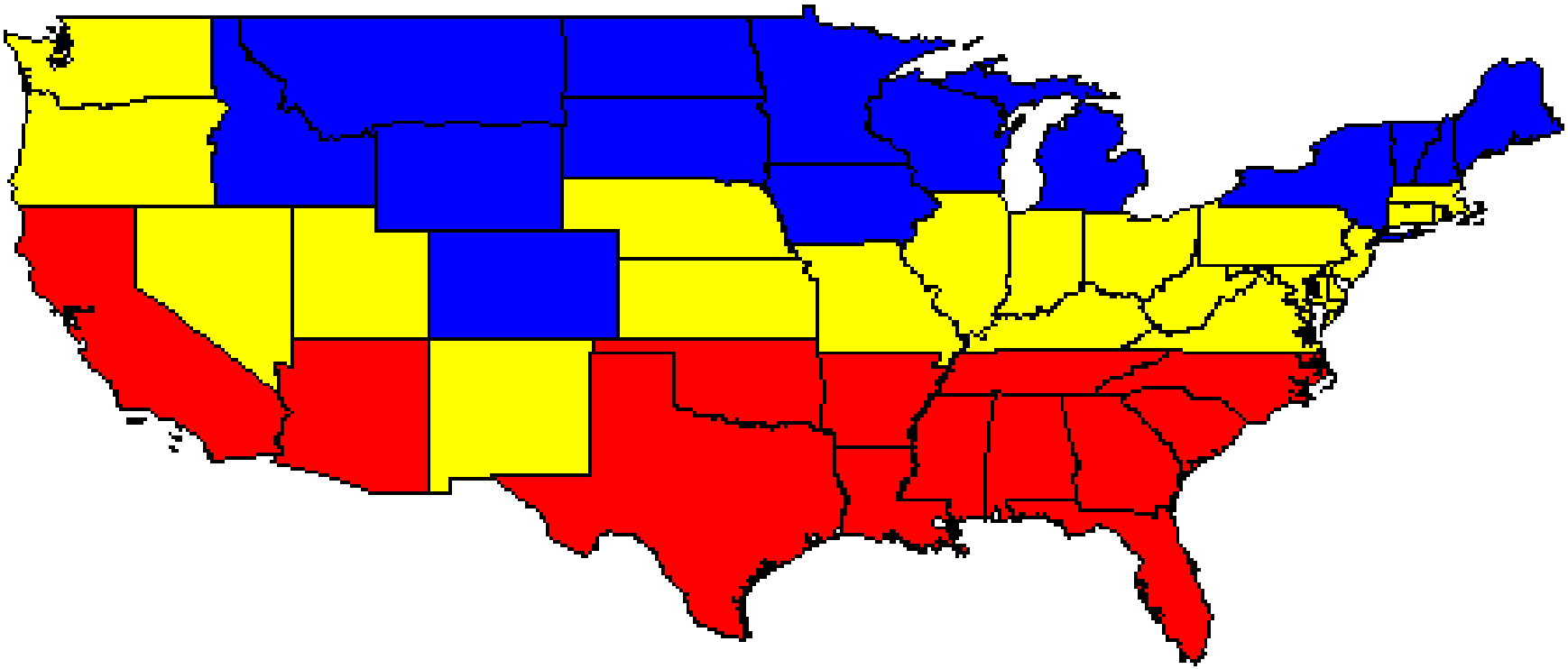
No_of_k	Crit	Qual	CH_Index
2	13368.39	0.66	90.78
3	6787.43	0.83	**109.27**
4	5046.14	0.87	100.87
5	3860.55	0.90	99.94
6	3371.69	0.92	90.63
7	3048.76	0.92	82.26
8	2577.18	0.94	82.42

DCA_results for k=3 prototypes

```
$solution$cardinality IDX
Cl.1 Cl.2 Cl.3
  13   14   21
$solution$centers
a matrix of distributions  12  variables  3  rows
$Crit       $quality
6787.425   0.8269466
```



DCA on USA k=3: the map



DCA interpretation using QPI

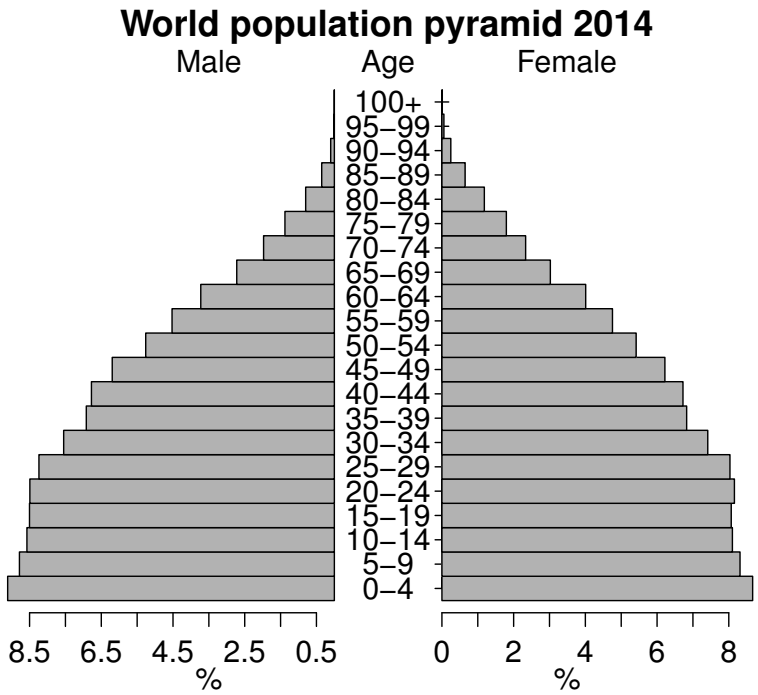
	TSS	WSS	BSS	Perc. of TSS $\frac{TSS(i)}{TSS}$	Perc. of quality $\frac{BSS(i)}{TSS(i)}$
Jan	6140.30	966.83	5173.47	15.66	84.25
Feb	5989.12	922.01	5067.11	15.27	84.61
Mar	4576.23	617.61	3958.62	11.67	86.50
Apr	3029.37	469.43	2559.94	7.72	84.50
May	2232.27	491.71	1740.56	5.69	77.97
Jun	1861.36	537.50	1323.87	4.75	71.12
Jul	1200.60	340.22	860.38	3.06	71.66
Aug	1451.14	347.46	1103.68	3.70	76.06
Sep	2128.07	383.84	1744.23	5.43	81.96
Oct	2401.76	397.58	2004.17	6.12	83.45
Nov	3379.04	561.03	2818.02	8.62	83.40
Dec	4832.32	752.21	4080.10	12.32	84.43
Total	39221.58	6787.43	32434.15	100.00	82.69

DCA position and variability components

	TSS	WSS	BSS	BSSc	BSSv	% of q. $\frac{BSS(i)}{TSS(i)}$	pos. comp.	var. comp.
Jan	6140.30	966.83	5173.47	5161.36	12.11	84.25	99.77	0.23
Feb	5989.12	922.01	5067.11	5054.33	12.78	84.61	99.75	0.25
Mar	4576.23	617.61	3958.62	3951.77	6.84	86.50	99.83	0.17
Apr	3029.37	469.43	2559.94	2555.28	4.66	84.50	99.82	0.18
May	2232.27	491.71	1740.56	1735.77	4.79	77.97	99.73	0.27
Jun	1861.36	537.50	1323.87	1321.33	2.53	71.12	99.81	0.19
Jul	1200.60	340.22	860.38	855.84	4.54	71.66	99.47	0.53
Aug	1451.14	347.46	1103.68	1100.55	3.13	76.06	99.72	0.28
Sep	2128.07	383.84	1744.23	1742.92	1.31	81.96	99.92	0.08
Oct	2401.76	397.58	2004.17	2001.44	2.73	83.45	99.86	0.14
Nov	3379.04	561.03	2818.02	2811.37	6.65	83.40	99.76	0.24
Dec	4832.32	752.21	4080.10	4064.92	15.18	84.43	99.63	0.37
Total	39221.58	6787.43	32434.15	32356.89	77.26	82.69	99.76	0.24

An example on poulation pyramids

We considered population age-sex pyramids data collected by the Census Bureau of USA in 2014.
A population pyramid is a common way to represent jointly the distribution of sex and age of people living in a given administrative unit (city, region or country, for instance).
In this dataset (available in the `HistDAWass` package with the name `Age_Pyramids_2014`), each country (228 countries) is represented by two histograms describing the age distribution for the male and the female population. Both distributions are represented by vertically juxtaposing, and the representation is similar to a pyramid. The shape of pyramids varies according to the distribution of the age in the population and it is related to the development of a country.



DCA with adaptive distances

In this example, we use the `Age_pyramids` dataset. We fix $k = 4$.

Adaptive DCA weights for each schema

A weight for each variable (Schema 1)

	Weights
Male.pop	0.99957
Fem.pop	1.00043

A weight for each component of each variable (Schema=2)

	Weights
Male.pop P	1.04032
Male.pop V	0.92821
Fem.pop P	0.96125
Fem.pop V	1.07734

A weight for each variable and each cluster (Schema 3)

	Cl.1	Cl.2	Cl.3	Cl.4
Male.pop	1.0486	0.9093	0.9838	1.0846
Fem.pop	0.9536	1.0998	1.0165	0.9220

A weight for each component of each variable and each cluster (Schema 4)

	Cl.1	Cl.2	Cl.3	Cl.4
Male.pop P	1.0799	0.9958	1.0797	1.0106
Male.pop V	1.0926	0.9445	0.9764	0.8157
Fem.pop P	0.9260	1.0042	0.9261	0.9895
Fem.pop V	0.9153	1.0587	1.0241	1.2259

Silhouette index an internal validity index

A internal validity index takes into account one or both the following information

- **Cluster Cohesion**: Measures how closely related are objects in a cluster. (Example: the within sum of squares)
- **Cluster Separation**: Measure how distinct or well-separated a cluster is from other clusters (Example: the between sum of squares)

An internal validity index that accounts for cohesion and separation is the Silhoutte index

It is an average of the silhouette value assigned to each observation $s(i) = \frac{b(i)-a(i)}{max(a(i),b(i))}$ where $a(i)$ is the average distance from the assigned cluster, $b(i)$ is the average from the second best cluster.

Method	Silhouette Index
Base	0.775862
Schema 1	0.775867
Schema 2	0.887746
Schema 3	0.899954
Schema 4	0.839478

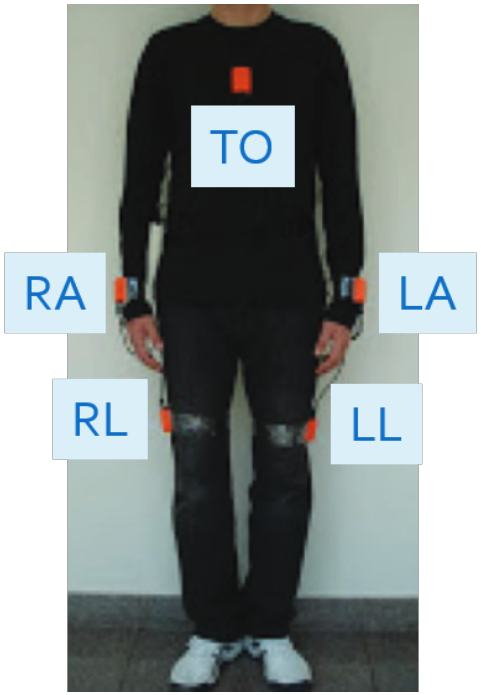
An application on an Activity Recognition dataset: 8 people walking

(Altun, Barshan, and Tunçel 2010) create an extensive dataset for activity recognition

Available at the UCI Machine Learning Repository

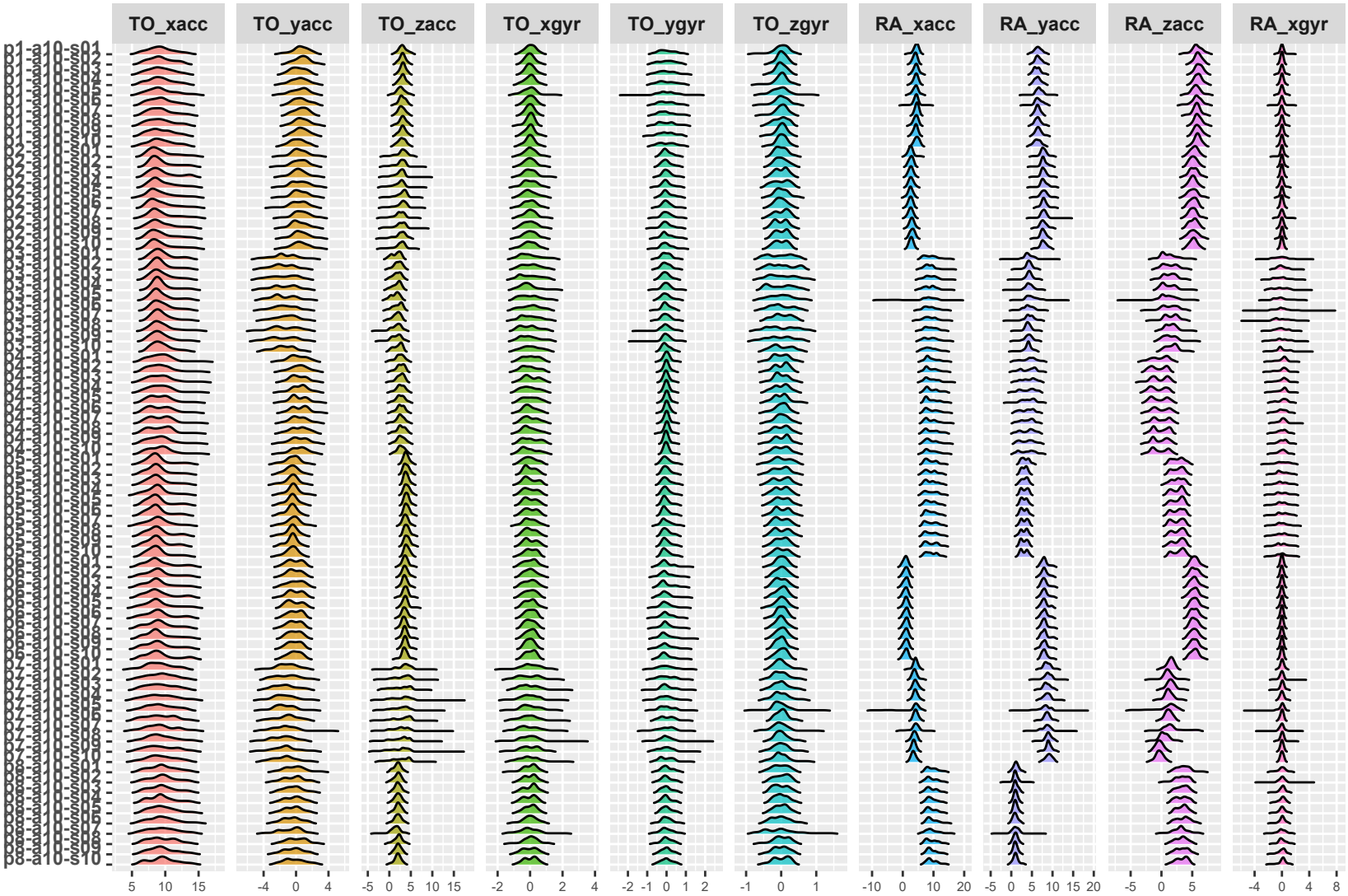
The data set consists of:

- 19 activities: sitting, standing, ascending/desdencing stairs, **walk 4kmH flat**, running, rowing, jumping, playing basketball, and more.
- 8 people between 20 and 30 years old (4 male, 4 female)
- Each person freely performed each activity during 5 minutes
- 45 measurements (5 times triaxial gyroscope, accelerometer, magnetometer) are recorder at a rate of 25Hz

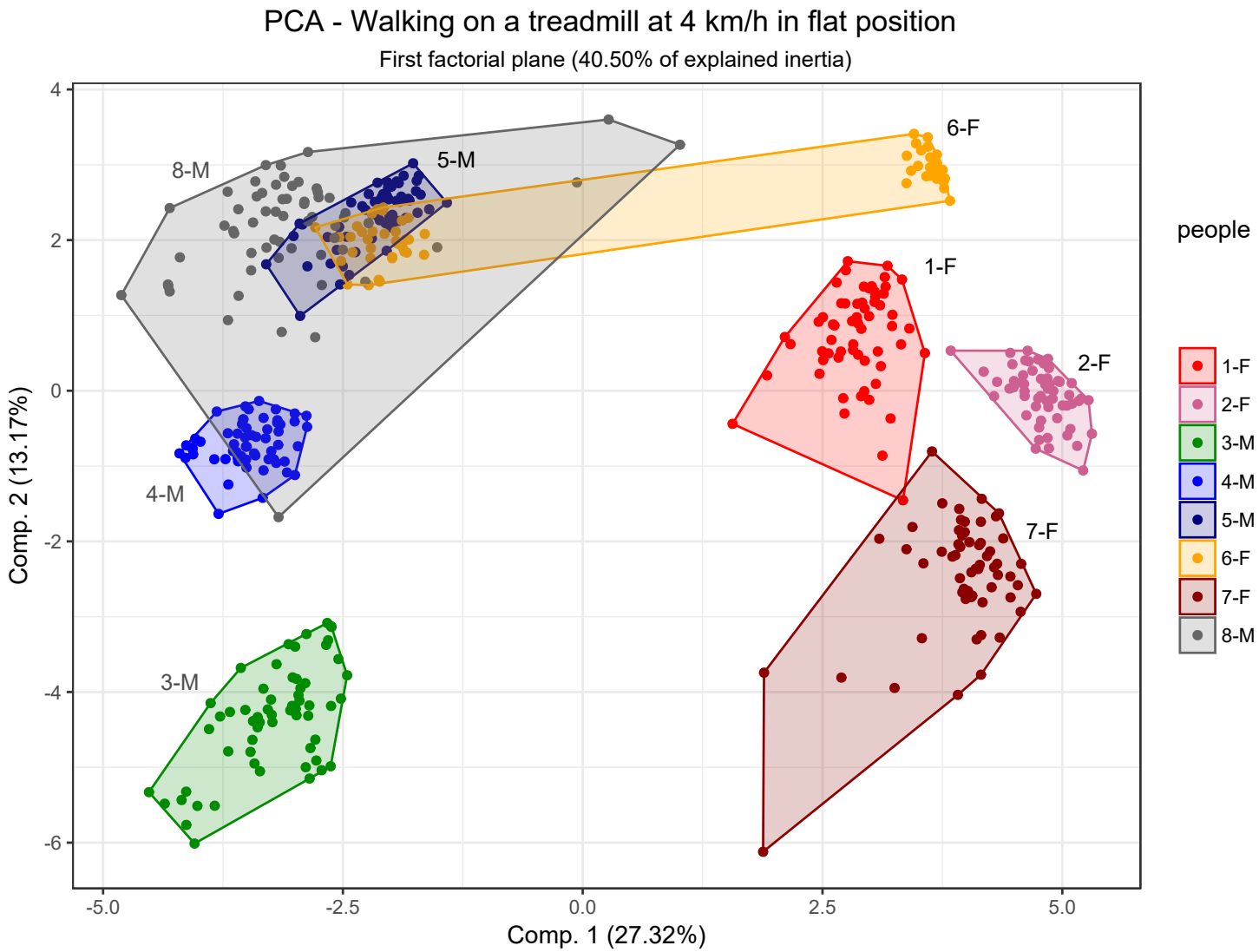


We don't use magnetometer in this application

An application on the AR dataset: 8 people walking 4kmH flat. (Each row is a 5 sec.-window of measurements)



A PCA on histograms: the individual plots



Dynamic clustering external validity indexes

Indexes:

ARI= adjusted Rand Index (accuracy), **PUR**=purity, **FM**=Folks-Mallows index, **NMI**=Normalized mutual information

Methods:

KM= Dynamic clustering (aka, K-means), **KMst**= Dynamic clustering whith stand. variables

ADA1 = 1 weight for each variable, **ADA2** = 2 weights for each variable (one for each component),

ADA3 = 1 weight for each variable and each cluster, **ADA4** = 2 weights for each variable (one for each component) and each cluster,

	k=4				k=5				k=6			
Mets	**ARI**	**PUR**	**FM**	**NMI**	**ARI**	**PUR**	**FM**	**NMI**	**ARI**	**PUR**	**FM**	**NMI**
KM	0.4652	0.5000	0.6051	0.6980	0.5555	0.6250	0.6608	0.7716	0.5627	0.6417	0.6626	0.7821
KMst	0.4489	0.5000	0.5995	0.7088	0.6330	0.6250	0.7141	0.8077	0.6811	0.7479	0.7468	0.8505
ADA_1	0.4448	0.5000	0.5962	0.7038	0.5043	0.6229	0.6330	0.7714	0.6766	0.7458	0.7434	0.8505
ADA_2	0.4126	0.5000	0.5744	0.6713	0.5059	0.6229	0.6331	0.7672	0.6884	0.7500	0.7530	0.8613
ADA_3	0.4225	0.5000	0.5838	0.6938	0.5077	0.6250	0.6334	0.7646	0.6884	0.7500	0.7530	0.8613
ADA_4	0.4090	0.5000	0.5679	0.6557	0.5024	0.6250	0.6258	0.7446	0.6674	0.7500	0.7454	0.8676

	k=7				k=8			
Mets	**ARI**	**PUR**	**FM**	**NMI**	**ARI**	**PUR**	**FM**	**NMI**
KM	0.5996	0.6417	0.6880	0.7987	0.6377	0.7479	0.7079	0.8338
KMst	0.7371	0.8208	0.7865	0.8931	0.8311	0.8750	0.8569	0.9171
ADA_1	0.7371	0.8208	0.7865	0.8931	0.8156	0.8667	0.8433	0.9015
ADA_2	0.7435	0.8229	0.7917	0.8990	**0.8336**	**0.8750**	**0.8592**	**0.9366**
ADA_3	0.7761	0.8208	0.8134	0.8914	**0.8839**	**0.9396**	**0.8988**	**0.9356**
ADA_4	0.8446	0.8646	0.8706	0.9241	0.7088	0.8042	0.7583	0.8499

Hierarchical clustering

Hierarchical clustering

```
results= WH_hclust (x, simplify=FALSE, qua=10, standardize=FALSE, distance=
                    method="complete")
```

Input

Input param.	Description
x	A MatH object (a matrix of distributionH)
simplify	A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
qua	An integer, if simplify=TRUE is the number of quantiles used for re-codify the histograms.
standardize	A logic value (default is FALSE). If TRUE histogram-valued data are standardized, variable by variable, using the Wasserstein-based standard deviation. Use if one wants to have variables with std equal to one.
distance	A string default WDIST the L_2 Wasserstein distance (other distances will be implemented)
method	A string, default="complete", is the the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of ward.D, ward.D2, single, complete, average (= UPGMA), mcquitty (= WPGMA), median (= WPGMC) or centroid (= UPGMC).

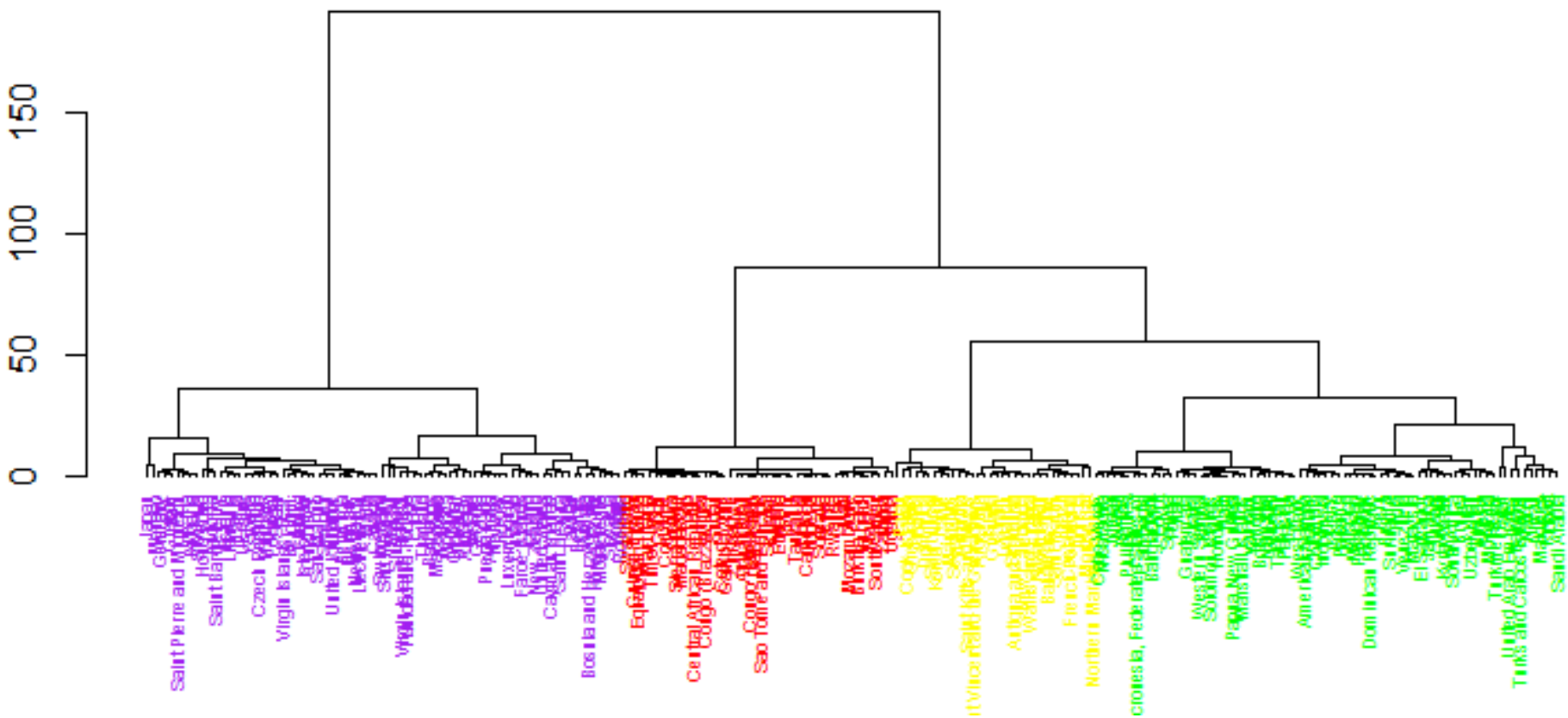
Output

An object of the class `hclust` which describes the tree produced by the method.

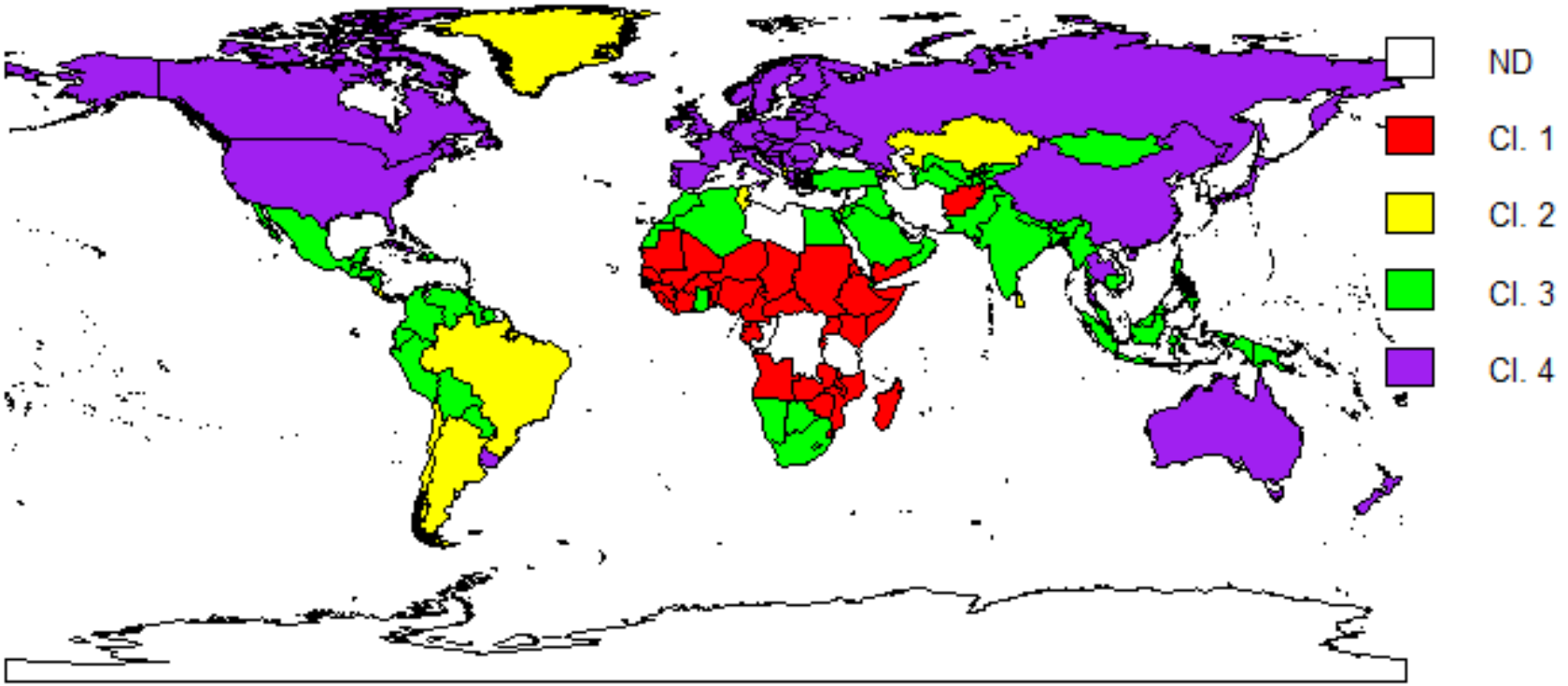
Application on the Age_pyramids dataset: the script

```
Work_data=Age_Pyramids_2014[2:229,2:3] #Take a part of the data
Hward=WH_hclust(Work_data,method = "ward.D2") #Do the dirty work
# cut dendrogram in 4 clusters
  hc=Hward
  hcd=as.dendrogram(hc)
  clusMember = cutree(hc, 4)
  labelColors = c("red",  "yellow", "green", "purple")
  # function to get color labels
  ...
  # using dendrapply
  clusDendro = dendrapply(hcd, colLab)
  # make plot
  clusDendro<-assign_values_to_leaves_nodePar(clusDendro, 0.5, "lab.cex")
  plot(clusDendro)
```

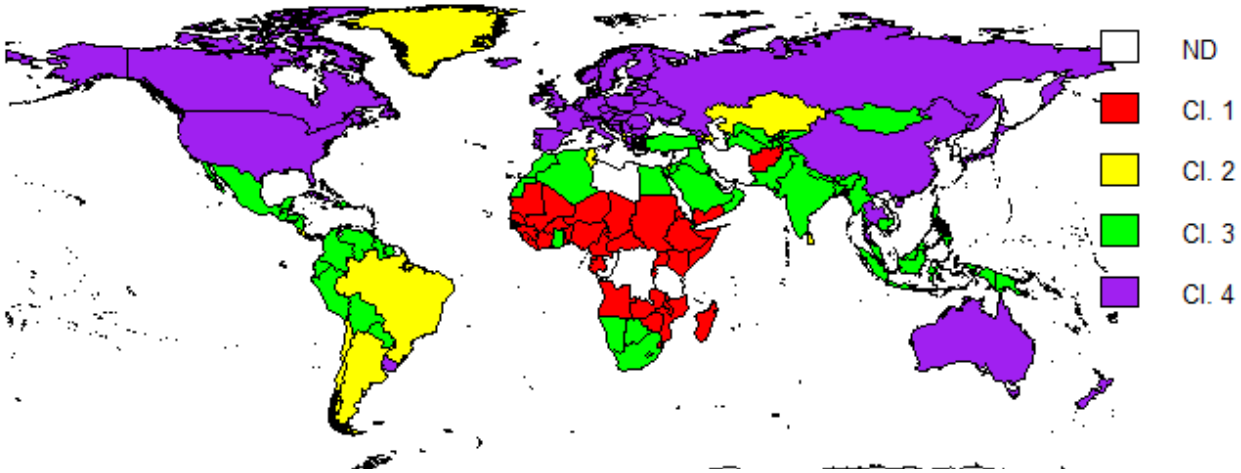
Show tree

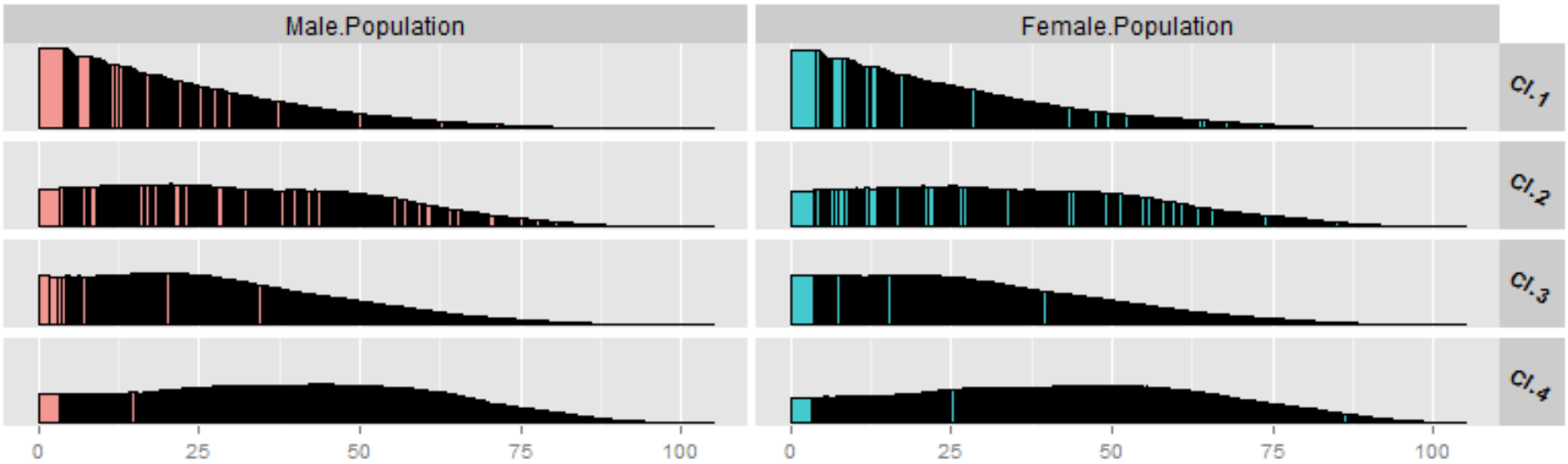


Show map



Show barycenters





Other implemented methods

Other methods

Kohonen Self Organizing Maps

Fuzzy c-means

Adaptive distances-based Fuzzy c-means

Open research issues and main references

Some open research issues

Considering the problem of finding "true clusters"

- How to combine qualitative and quantitative data (hetereogeneity)
- How to consider clustering as a predictive method (a great challenge!)
  We imagine that we have a set of variables defining clusters and a set of explicative variables (for validating clusters). Is it possible to define a general strategy for predictive clustering? This may be relevant in several applicative fields: marketing, time series forecasting, ...

References

Altun, Kerem, Billur Barshan, and Orkun Tunçel. 2010. "Comparative Study on Classifying Human Activities with Miniature Inertial and Magnetic Sensors." *Pattern Recognition* 43 (10): 3605–20. doi:https://doi.org/10.1016/j.patcog.2010.04.019.

Bock, H.H., and E. Diday. 2000. *Analysis of Symbolic Data: Exploratory Methods for Extracting Statistical Information from Complex Data*. Springer verlag.

Hennig, Christian. 2015. "What Are the True Clusters?" *Pattern Recognition Letters* 64: 53–62. doi:https://doi.org/10.1016/j.patrec.2015.04.009.

Irpino, A., R. Verde, and F.A.T. De Carvalho. 2014. "Dynamic Clustering of Histogram Data Based on Adaptive Squared Wasserstein Distances." *Expert Systems with Applications* 41 (7): 3351–66. doi:http://dx.doi.org/10.1016/j.eswa.2013.12.001.

Irpino, Antonio, and Rosanna Verde. 2015. "Basic Statistics for Distributional Symbolic Variables: A New Metric-Based Approach." *Advances in Data Analysis and Classification* 9 (2). Springer Berlin Heidelberg: 143–75. doi:10.1007/s11634-014-0176-4.

Mizuta, Masahiro. 2016. "Mini Data Approach to Big Data." *Medical Imaging and Information Sciences* 33 (1): 1–3. doi:10.11318/mii.33.1.

Verde, Rosanna, and Antonio Irpino. 2007. "Dynamic Clustering of Histogram Data: Using the Right Metric." In *Selected Contributions in Data Analysis and Classification*, edited by Paula Brito, Guy Cucumel, Patrice Bertrand, and Francisco Carvalho, 123–34. Studies in Classification, Data Analysis, and Knowledge Organization. Berlin, Heidelberg: Springer Berlin Heidelberg.