

# Package ‘kml’

November 13, 2009

**Type** Package

**Title** K-Means for Longitudinal Data (KmL)

**Version** 1.0

**Date** 2002-12-01

**Author** Christophe Genolini

**Maintainer** Christophe Genolini <genolini@u-paris10.fr>

**Description** KmL is an implementation of k-means specifically design to deal with longitudinal data. It provide facilities to deal with missing value and propose a graphical interphace for chosing the correct number of clusters.

**License** GPL (>= 2)

**Lazyload** yes

**Depends** methods,longitudinalData,clv

**URL** <http://www.r-project.org>,<http://christophe.genolini.free.fr/kml>

**Collate** global.r clusterization.r clusterizLongData.r kml.r

**Encoding** latin1

**Repository** CRAN

**Date/Publication** 2009-11-13 10:43:31

## R topics documented:

kml-package . . . . .	2
affectIndiv . . . . .	4
affectIndivGeneralized . . . . .	6
as.clusterizLongData . . . . .	7
calculCenterGeneralized . . . . .	9
calculMean . . . . .	11

choice . . . . .	12
clusterization . . . . .	15
Clusterization-class . . . . .	17
clusterizLongData . . . . .	19
ClusterizLongData-class . . . . .	21
exportClusterization . . . . .	24
kml . . . . .	26
partitionInitialise . . . . .	29
plot,ClusterizLongData . . . . .	31
plotAll,ClusterizLongData . . . . .	33
plotCriterion . . . . .	36
updateClusterization . . . . .	37
updateClusterizLongData . . . . .	39

<b>Index</b>	<b>41</b>
--------------	-----------

---

kml-package                      ~ Overview: K-means for Longitudinal data ~

---

## Description

KmL is a new implematation of k-means for longitudinal data (or trajectories). Here is an overview of the package. For the description of the algorithm, see [kml](#).

## Details

```

Package:    kml
Type:      Package
Version:   1.0
Date:     2009-12-01
License:   GPL (>= 2)
Lazyload: yes
Depends:  methods,clv,longitudinalData
URL:      http://www.r-project.org
URL:      http://christophe.genolini.free.fr/kml

```

## Overview

To clusterize data, KmL go through three steps, each of which is associated to some functions:

1. Data preparation
2. Building "optimal" clusterization.
3. Exporting results

## 1. Data preparation

kml works on object of class `ClusterizLongData`. Data preparation therefore simply consists in transforming data into an object `ClusterizLongData`. This can be done via function `clusterizLongData` (`cld` in short) or `as.clusterizLongData` (`as.cld` in short). The former lets the user build some data from scratch, the latter converts a `data.frame` in `ClusterizLongData`.

Instead of working on real data, one can also work on artificial data. Such data can be created with `generateArtificialLongData` (`gald` in short) from the package `longitudinalData`.

## 2. Building "optimal" clusterization

Once an object of class `ClusterizLongData` has been created, the algorithm `kml` can be run.

Starting with a `ClusterizLongData`, `kml` built a `Clusterization`. A object of class `Clusterization` is a partition of trajectories into subgroups that also contains some information like the percentage of trajectories contained in each group or some quality criterion (like the Calinski & Harabasz).

`kml` is a "hill-climbing" algorithm. The specificity of this kind of algorithm is that it always converges towards a maximum, but one cannot know whether it is a local or a global maximum. It offers no guarantee of optimality.

To maximize one's chances of getting a quality `Clusterization`, it is better to execute the hill climbing algorithm several times, then to choose the best solution. By default, `kml` executes the hill climbing algorithm 20 times and chooses the `Clusterization` maximising the determinant of the matrix between.

Likewise, it is not possible to know *beforehand* the optimum number of clusters. On the other hand, *afterwards*, it is possible to calculate clues that will enable us to choose.

In the end, `kml` tests by default 2, 3, 4, 5 et 6 clusters, 20 times each.

## 3. Exporting results

When `kml` has constructed some `Clusterization`, the user can examine them one by one and choose to export some. This can be done via function `choice`. `choice` opens a graphic windows showing various information including the trajectories cluterized by a specific `Clusterization`.

When some `Clusterization` has been selected (the user can select more than 1), it is possible to save them. The clusters are therefore exported towards the file `nom-cluster.csv`. Criteria are exported towards `nom-criteres.csv`. The graphs are exported according to their extension.

### Author(s)

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

### English translation

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

### References

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
Web site: <http://christophe.genolini.free.fr/kml>

### See Also

Overview: [kml-package](#)  
Classes : [ClusterizLongData](#), [Clusterization](#)  
Methods : [clusterizLongData](#), [kml](#), [choice](#), [as.clusterizLongData](#)  
Plot : [plot\(ClusterizLongData\)](#), [plotCriterion](#), [plotSubGroups\(ClusterizLongData\)](#),  
[plotAll\(ClusterizLongData\)](#)

### Examples

```
### 1. Data Preparation
myCld <- as.clusterizLongData(generateArtificialLongData())

### 2. Building "optimal" clusterization (with only 3 redrawings)
kml(myCld, nbRedrawing=3, print.cal=TRUE, print.traj=TRUE)

### 3. Exporting results
try(choice(myCld))
```

---

affectIndiv

~ Function: *affectIndiv* ~

---

### Description

Given some longitudinal data (trajectories) and k clusters centres, `affectIndiv` affect each individual to the cluster whose centre is the closest.

### Usage

```
affectIndiv(traj, clustersCenter, distance = "euclidean", power = 2)
```

## Arguments

<code>traj</code>	[matrix]: longitudinal data. Each line is an individual, each column is a time measurement.
<code>clustersCenter</code>	[matrix]: clusters centre. Each line is a cluster center, each column is a time measurement.
<code>distance</code>	[character]: use to estimate the distance between an individual and a clusters centre. Should be one off "manhattan", "euclidean", "minkowski", "maximum", "canberra" or "binary".
<code>power</code>	[numeric]: if the distance is "minkowski", <code>power</code> give the power to use.

## Details

EM algorithm (like k-means) alternates between two phases : Esperance and Maximisation. During Maximisation, each individual is affected to the closest cluster. This is what `affectIndiv` does.

Note that `affectIndiv` does not work with `ClusterizLongData` object but with a matrix.

`affectIndiv` used with `calculMean` simulates one k-means step.

This function is programmed in C, it is expected to be fast.

## Value

Object of class `Partition`.

## Author(s)

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

## English translation

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

## References

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####
### affectIndiv

### Some LongitudinalData
traj <- as.cld(gald())["traj"]

### 4 clusters centers
center <- traj[runif(4,1,nrow(traj)),]

### Affectation of each individual
affectIndiv(traj,center)
```

---

```
affectIndivGeneralized
~ Function: affectIndivGeneralized ~
```

---

**Description**

Given some longitudinal data (trajectories) and k clusters centers, `affectIndiv` affects each individual to the cluster whose center is the closest.

**Usage**

```
affectIndivGeneralized(traj, clustersCenter, distance = function(x, y) {
  return(dist(t(cbind(x, y))))
})
```

**Arguments**

<code>traj</code>	[matrix]: longitudinal data. Each line is an individual, each column is a time measurement.
<code>clustersCenter</code>	[matrix]: clusters center. Each line is a cluster center, each column is a time measurement.
<code>distance</code>	[function]: function used to estimate the distance between an individual and a clusters center. It can be used to deal with non classical distance.

**Details**

EM algorithm (like k-means) alternates between two phases : Esperance and Maximisation. During Maximisation, each individual is affected to the closest cluster. This is what `affectIndivGeneralized` does. In addition to `affectIndiv`, it also let the user to define a non-classical distance.

Note that `affectIndiv` does not work with `ClusterizLongData` object but with a matrix.

`affectIndivGeneralized` used with `calculCenterGeneralized` simulates one step of EM-algorithm.

This function is programmed in R, it is not expected to be as fast as `affectIndiv`.

**Value**

Object of class `Partition`.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
 Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####
### affectIndiv

### Some LongitudinalData
traj <- as.cld(gald())["traj"]

### 4 clusters centers
center <- traj[runif(4,1,nrow(traj)),]

### Distance unusual
distCor <- function(x,y){return(cor(x,y))}

### Affectation of each individual
affectIndivGeneralized(traj,center,distance=distCor)
```

---

```
as.clusterizLongData
```

*~ Function: as.clusterizLongData (or as.cld) ~*

---

**Description**

as.clusterizLongData (or as.cld) turns a data.frame into an object of class `ClusterizLongData`.

**Usage**

```

as.cld(data, ...)

#as.cld(data,id=data[,1],timeCol=2:length(data),timeReal=0:(length(timeCol)-1),traj_
#   varName=sub("[[:digit:]]*$", "", names(data)[timeCol[1]]), ...)

as.clusterizLongData(data, ...)

#as.clusterizLongData(data,id=data[,1],timeCol=2:length(data),timeReal=0:(length(ti
#   varName=sub("[[:digit:]]*$", "", names(data)[timeCol[1]]), ...)

```

**Arguments**

data	[data.frame]: contains the trajectories (longitudinal data). Each line refers to the trajectory of an individual. Each column refers to the time at which measures were made. Optionally, the first column may refer to identifiers.
...	<ul style="list-style-type: none"> <li>• <code>id[character]</code>: single identifier for each trajectory (i.e. each individual). By default, <code>id</code> is the first column of the <code>data.frame</code>.</li> <li>• <code>timeCol[numeric]</code>: column number in which longitudinal data can be found. By default, <code>timeCol</code> is all the columns except for the first.</li> <li>• <code>timeReal[numeric]</code>: time at which measures were made in "real" life. By default, <code>timeReal</code> is <code>0:(length(timeCol)-1)</code>.</li> <li>• <code>varName[character]</code>: Name of the variable being measured. By default, the name of the second column after suppressing the numbers is chosen (if the name of the second column is <code>SizeAt4</code>, then <code>varName</code> will be <code>SizeAt</code>).</li> <li>• <code>trajMinSize[numeric]</code>: The trajectories that include missing values can either be excluded or included. <code>trajMinSize</code> sets the minimum number of values that a trajectory must contain not to be excluded. For example, if the trajectories have 7 measurements (<code>time=7</code>) and <code>trajSizeMin</code> is set to 3, the trajectory <code>(5, 3, NA, 4, NA, NA, NA)</code> will be included in the calculation while <code>(2, NA, NA, NA, 4, NA, NA)</code> will be excluded. Please note that trajectories that are completely missing (0 present values) must always be excluded.</li> </ul>

**Details**

`as.cld` apply on a `data.frame` turn the `data.frame` into an object of class `ClusterizLongData`. Each line of the data frame refers to a trajectory (an individual), the columns specified in `timeCol` are the time.

**Value**

An object of class `ClusterizLongData`.

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
Classes : [ClusterizLongData](#)  
Methods : [clusterizLongData](#), [generateArtificialLongData](#)  
Plot : [plot\(ClusterizLongData\)](#), [plot\(Calinski\)](#), [plotSubGroups\(ClusterizLongData\)](#),  
[plotAll\(ClusterizLongData\)](#)

**Examples**

```
### Simple use
dn <- data.frame(i=11:13, size12=c(15, 13, 14), size14=c(16, 15, 17), size18=c(18, 16, 16))
as.clusterizLongData(dn)

### Changing parameters
as.clusterizLongData(dn, i=c("H101", "H108", "B103"), timeCol=c(2, 2, 3, 3, 3, 3, 4), timeReal=12:18)
```

---

calculCenterGeneralized

~ Function: calculCenterGeneralized ~

---

**Description**

Given some longitudinal data and a [Partition](#), calculCenterGeneralized computes the center trajectories of each cluster.

**Usage**

```
calculCenterGeneralized(traj, xPart, centerMethod = meanNA)
```

**Arguments**

`traj` [matrix]: longitudinal data. Each line is an individual, each column is a time measurement.

`xPart` [Partition]: affectation of each individual.

`centerMethod` [function]: function used to calculate the center of each cluster.

**Details**

EM algorithm (like k-means) alternates two phases: Esperance and Maximisation. During Esperance, the center of each cluster is evaluated. This is what `calculCenterGeneralized` does.

Note that `calculCenterGeneralized` does not work with `ClusterizLongData` object but with a matrix.

`affectIndivGeneralized` used with `calculCenterGeneralized` simulates one step of EM-algorithm.

This function is programmed in R, it is not expected to be as fast as `calculMean`.

**Value**

A matrix with `k` lines and `t` columns containing `k` clusters centers. Each line is a center, each column is a time measurement.

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author: <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####
### calculCenterGeneralized

### Some LongitudinalData
traj <- as.cld(gald())["traj"]

### A partition
part <- partition(floor(runif(200,1,5)),4)

### Clusters center
calculCenterGeneralized(traj,part,medianNA)
```

---

calculMean                      ~ *Function: calculMean* ~

---

**Description**

Given some longitudinal data and a [Partition](#), `calculMean` computes the mean trajectories of each cluster.

**Usage**

```
calculMean(traj, xPart)
```

**Arguments**

traj	[matrix]: longitudinal data. Each line is an individual, each column is a time measurement.
xPart	[Partition]: affectation of each individual.

**Details**

EM algorithm (like k-means) alternates between two phases : Esperance and Maximisation. During Esperance, the mean of each cluster is evaluated. This is what `calculMean` does.

Note that `calculMean` does not work with [ClusterizLongData](#) object but with a matrix.

`affectIndiv` used with `calculMean` simulates one k-means step.

This function is programmed in C, it is expected to be fast.

**Value**

A matrix with k line and t column containing k clusters centers. Each line is a center, each column is a time measurement.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
 Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####
### calculMean

### Some LongitudinalData
traj <- as.cld(gald())["traj"]

### A partition
part <- partition(floor(runif(200,1,5)),4)

### Clusters center
calculMean(traj,part)
```

---

 choice

 ~ Function: choice ~
 

---

**Description**

choice lets the user choose some Clusterization he wants to export.

**Usage**

```
choice(Object, typeGraph = "bmp", ...)
```

## Arguments

Object	[ClusterizLongData]: Object containing the trajectories and all the clusterizations found by <code>kml</code> from whom the user want to export some <code>Clusterization</code> .
typeGraph	[character] for every selected <code>clusterization</code> , choice export some graphs. <code>type</code> set the format that will be used. Possible formats are the one available for <code>savePlot</code>
...	

## Details

`choice` is a function that lets the user see the `Clusterization` found by `kml`. At first, `choice` opens a graphics window. On the left side are the Calinski criteria of all the `Clusterization` contained in `Object`. One `Clusterization` is 'active', it is the one marked by a black dot. On the right side, the trajectories of `Object` are drawn, according to the active `Clusterization`.

From there, `choice` offers numerous options :

**Arrow** Change the active `Clusterization`.

**Space** Select/unselect a `Clusterization` (the selected `Clusterization` are surrounded by a circle).

**Return** Export all the selected `Clusterization`, then quit the function `choice`.

'c' Switch the graphical representation of the Calinski criterions on/off.

'e' Switch the graphical representation of the trajectories on/off (see `plot` for details).

'd' Switch the graphical representation of the trajectories' subgroups on/off (see `plotSubGroups` for details).

'r' Change the trajectories color.

'f' Change the trajectories' subgroups color.

't' Change the mean trajectories color.

'g' Change the mean trajectories' subgroups color.

'y' Change the mean trajectories symbols.

'h' Change the mean trajectories' subgroups symbols.

'u' Increase the size of the mean trajectories symbols.

'j' Increase the size of the mean trajectories subgroups symbols.

'i' Decrease the size of the mean trajectories symbols.

'k' Decrease the size of the mean trajectories subgroups symbols.

Note that the letter on the upper line (e,r,t,y,u,i) concerns the main graph, the letter on the lower line (d,f,g,h,j,k) concerns the subgroups. ; the letter on the first column (e,d,c) switches the graphic representation on/off, the second and third columns (r,f,t,g) change the color and the other columns (y,h,u,j,i,k) deal with the symbols.

When 'return' is pressed, the selected `Clusterization` are exported. Exporting is done in a specific folder named `objectName-NumberOfCluster-OrderInTheSublist`. Four files are created :

**name-Clusters.csv** Table with two columns. The first is the identifier of each trajectory ; the second holds the cluster's affectation of the trajectory.

**name-Detail.csv** Table containing information about the clusterization (the Calinski criterion and the percentage of individual in each cluster)

**name-Traj.ext** Graph (of type 'ext') representing the trajectories. All the parameters set during the visualization (color of the trajectories, symbols used, mean color) are used for the export.

**name-SubGroups.ext** Graph (of type 'ext') representing the trajectories subgroups. All the parameters set during the visualization (color of the trajectories, symbols used, mean color) are used for the export.

### Value

For each selected `Clusterization`, a folder containing four files.

### Author(s)

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

### English translation

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

### Author(s)

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

### English translation

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

### References

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**References**

Article submitted

Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)

Classes : [ClusterizLongData](#), [Clusterization](#)

Methods : [kml](#)

Plot : [plot\(ClusterizLongData\)](#), [plotCriterion](#)

**Examples**

```
### Creation of artificial data
cld1 <- as.cld(gald())

### Clusterisation
#kml(cld1, nbRedrawing=3, printCal=TRUE, printTraj=TRUE)

### Selection of the clusterization we want
# (note that "try" is for compatibility with CRAN only,
# you probably can use "choice(cld1)")
try(choice(cld1))
```

---

clusterization      ~ *Function: clusterization* ~

---

**Description**

clusterization is the constructor of the class [Clusterization](#).

**Usage**

```
clusterization(xPartition, yLongData, convergenceTime = 0, criterionName = "calinsk
```

**Arguments**

xPartition      [Partition]: object that will be turn into a [Clusterization](#)

yLongData      [LongData]: longitudinal data on which the clusterization has been run.

convergenceTime

[numeric]: if the clusterization has been obtained through an algorithm, number of steps of this algorithm before convergence.

criterionName

[character]: criterion used to evaluate the quality of the partitioning.

criterionValue

[numeric]: value of the criterion used to evaluate the quality of the partitioning.

imputationMethod

[character]: name of the methods used to imput missing values. See [imputation](#).

startingCondition

[character]: name of the methods used to define starting condition. See [partitionInitialise](#).

algorithmUsed

[character]: algorithm used to obtain the partition. Only k-means is available at this time.

## Details

In KmL, strickly speaking, a [Partition](#) is just a sequence of letters (independent of any trajectories) ; a [Clusterization](#) is a [Partition](#) associated with a set of trajectories. So a [Clusterization](#) is a [Partition](#) with some additional information like some quality criterion, the name and convergence time of the algorithm use to clusterize the population...

## Value

Object of class [Clusterization](#).

## Author(s)

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

## English translation

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

## References

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

## Examples

```
### Creation of a partition
part <- partition(rep(c(1,2),4),2)

### Some trajectories
traj1 <- gald(nbEachClusters=2)

### Transformation of part into a Clusterization
clusterization(part,traj1)
```

```

# Calinski criterion is around 0.50...

### Some other trajectories
traj2 <- gald(nbEachClusters=4, functionClusters=list(function(t){5-t},function(t){5+t}))

### Transformation of part into a Clusterization
clusterization(part,traaj2)
# Calinski criterion is around 0.15...

# part is probably a good partition for traj1, but not for traj2...

```

---

```

Clusterization-class
~ Class: Clusterization ~

```

---

### Description

An object of class `Clusterization` is a [Partition](#) of trajectories in subgroups. The object also contains some information like the percentage of trajectories that each group contains or a quality criterion.

### Objects from the Class

Objects are not intended to be created by users. `Clusterization` are created by `kml` and directly added to a [ClusterizLongData](#) object.

### Slots

`nbClusters` [numeric]: number of groups, between 1 and 52

`clusters` [vector(factor)]: vector containing the affectation group of each individual. The groups are in capital or small letters, from A to `LETTERSletters[nbClusters]`. Please note that a group might be empty.

`percentEachCluster` [vector(numeric)]: percentage of trajectories contained in each group.

`criterionName` [character]: Name of the quality criterion used to evaluate the quality of the clusterization (Calinski criterion is the only criterion available at this time)

`criterionValue` [numeric]: Value of the quality criterion used to evaluate the quality of the clusterization.

`imputationMethod`: [numeric] the calculation of quality criterion does not hold missing value. This variable saves the name of the imputation method used. See [imputation](#).

`startingCondition` [character]: if the `Clusterization` has been obtained by running an algorithm, this variable saves the way of defining the starting condition.

`algorithmUsed` [character]: if the `Clusterization` has been obtained by running an algorithm, this variable saves the name of the algorithm used.

`convergenceTime` [numeric]: if the `Clusterization` has been obtained by running an algorithm, this variable saves the number of iteration necessary to converge.

**validation rules**

An object `Clusterization` must follow the same rules as a `Partition`.

**Construction**

Class `Clusterization` objects are constructed through the `kml` procedure and are directly added to a `ClusterizLongData` object. They can also be constructed by the users using `clusterization`.

**Getteur [**

- Object["clusters" ]** [vector(factor)]: Gets the cluster of each individual (the value of the slot `clusters`)
- Object["nbClusters" ]** [numeric]: Gets the number of clusters (the value of the slot `nbClusters`)
- Object["percentEachClusters" ]** [numeric]: Gets the percentage of individuals present in each cluster.
- Object["convergenceTime" ]** [numeric]: Gets the convergence time of the Clusterization.
- Object["criterionName" ]** [character]: Gets the name of criterion used to estimate the Clusterization quality.
- Object["criterionValue" ]** [numeric]: Gets the value of criterion used to estimate the Clusterization quality.
- Object["imputationMethod" ]** [character]: Gets the name of the imputation method.
- Object["startingCondition" ]** [character]: Gets the name of starting condition.
- Object["algorithmUsed" ]** [character]: Gets the name of the algorithm used to find the clusterization.
- Object["myCriterion" ]** [numeric]: If "myCriterion" is the name of a quality criterion, gets its value.

**Setteur**

`Clusterization` are not intended to be modified, no setter is defined.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
 Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
 Classes : [ClusterizLongData](#), [Clusterization](#)  
 Methods : [kml](#)

**Examples**

```
showClass("Clusterization")
```

---

`clusterizLongData` ~ *Function: clusterizLongData (or cld)* ~

---

**Description**

`clusterizLongData` (or `cld` in short) is the constructor for `ClusterizLongData` object.

**Usage**

```
clusterizLongData(traj, id, time, varName = "v", trajMinSize = 1)
cld(traj, id, time, varName = "v", trajMinSize = 1)
```

**Arguments**

<code>traj</code>	[array(numeric)]: contains longitudinal data. Each line is the trajectory of an individual. The columns refer to the time during which measures were made.
<code>id</code>	[character] : single identifier for each individual (i.e. each trajectories). Note that the identifiers are of type <code>character</code> (that allow to deal identifiers like XUK32-612 identifiers that our favorite epidemiologists are so good at providing) . If <code>numeric</code> are provided, they are converted into <code>characters</code> .
<code>time</code>	[numeric]: time during which measures were made.
<code>varName</code>	[character]: name of the variable measured.
<code>trajMinSize</code>	[numeric]: Trajectories whose values are partially missing can either be excluded by treatment, or included. <code>trajSizeMin</code> sets the minimum number of values that a trajectory must contain not to be excluded. For example, if the trajectories have 7 measurements ( <code>time=7</code> ) and <code>trajSizeMin</code> is set to 3, the trajectory <code>(5, 3, NA, 4, NA, NA, NA)</code> will be included in the calculation while <code>(2, NA, NA, NA, 4, NA, NA)</code> will be excluded. Please note that trajectories that are totally missing (i.e. 0 present values) are always excluded.

**Details**

`clusterizLongData` create a `ClusterizLongData` object and set its slot to the corresponding value. Note that the field `clusters` cannot be initialized through this function since this slot is not supposed to be manipulated by the user (only by `kml`).

**Value**

Object of class `ClusterizLongData`.

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
Classes : `ClusterizLongData`  
Methods : `choice`, `as.clusterizLongData`  
Plot : `plot(ClusterizLongData)`, `plotSubGroups(ClusterizLongData)`, `plotAll(ClusterizLongD`

**Examples**

```
#####
### Creation of some trajectories
mat <- matrix(c(1,2,3,1,NA,6,1,8,NA),3)

#####
### Creation of LongitudinalData
(ld1 <- cld(traj=mat,id=1:3,time=c(1,2,3),varName="V",trajMinSize=2))
(ld2 <- clusterizLongData(traj=mat,id=c("A-101","A-102","A-108"),time=c(2,4,8),varName="Age")
```

---

```
ClusterizLongData-class
~ Class: ClusterizLongData ~
```

---

## Description

ClusterizLongData is an object containing trajectories and associated clusterizations.

## Objects from the Class

`kml` is an algorithm that builds a set of `Clusterization` from longitudinal data. `ClusterizLongData` is the object containing the original longitudinal data and all the `Clusterization` that `kml` finds.

When created, an `ClusterizLongData` object simply contains initial data (the trajectories). After the execution of `kml`, it contains the original data and the `Clusterization` which has just been calculated by `kml`.

Please note that if `kml` is executed several times, every new `Clusterization` is added to the original ones, no pre-existing `Clusterization` is erased.

## Slots

`id` [`character`]: single identifier for each of the trajectories (each individual).

`time` [`numeric`]: time at which measures were made.

`traj` [`array(numeric)`]: contains longitudinal data. Each line corresponds to the trajectory of an individual. The columns refer to the time during which measures were made.

`varName` [`character`]: Name of the variable measured.

`trajMinSize` [`numeric`]: Trajectories whose values are partially missing can either be excluded or included in the computation. `trajMinSize` sets the minimum number of values that a trajectory must contain to not be excluded. For example, if the trajectories have 7 measurements (`time=7`) and `trajMinSize` is set to 3, the trajectory `(5, 3, NA, 4, NA, NA, NA)` will be included in the calculation while `(2, NA, NA, NA, 4, NA, NA)` will be excluded. Please note that trajectories that are totally missing (i.e. 0 present values) are always excluded.

`clusters` [`list(list(Clusterization))`]: `clusters` contains the list of `Clusterization` found by `kml`. More specifically, `clusterizList` contains 52 items: `c1`, `c2`, `c3`, `c4` up to `c52`. Each item contains a list of `Clusterization`: `c2` contains all the `Clusterization` having 2 clusters, `c3` contains all the `Clusterization` having 3 clusters and so on.

`other` [`list`]: list of additional information (see section `Value` in `generateArtificialLongData` for an example).

## Construction

Class `ClusterizLongData` objects can be constructed via function `cld` (build from scratch) and via `as.cld` (turning a `data.frame` or a `LongData` into a `ClusterizLongData`). Note that some artificial data can be generated using the combination of `as.cld` and `generateArtificialLongData`.

**Get [**

- Object["id" ]** [vecteur(character)]: Gets each individual Identifier the value of the slot `id`)
- Object["time" ]** [vecteur(numeric)]: Gets the times (the value of the slot `time`)
- Object["varName" ]** [character]: Gets the name of the variable (the value of the slot `varName`)
- Object["trajMinSize" ]** [numeric]: Gets the limit for not excluding a trajectory containing missing values (the value of the slot `trajMinSize`)
- Object["traj" ]** [matrix(numeric)]: Gets all the trajectories' values (the value of the slot `traj`)
- Object["clusters" ]** [list(list(Clusterization))]: Gets the list of all the sublists of `Clusterization` contained in the object.
- Object["clusters",3 ]** [list(Clusterization)]: Gets the sublists "`c3`", the sublist that contain all the `Clusterization` with 3 clusters.
- Object["clusters",c(3,2) ]** [Clusterization]: Get the second `Clusterization` of the sublist "`c3`".
- Object["criterionValue" ]** [matrix(numeric)]: Gets the matrix of the criterion values for all the `Clusterization` contained in the object.
- Object["criterionValue",3 ]** [vector(numeric)]: Gets the vector of the criterion values for all the `Clusterization` with 3 clusters (sublist "`c3`").
- Object["criterionValue",c(3,2) ]** [numeric]: Gets criterion values for the second `Clusterization` of the sublist "`c3`".
- Object["criterionName" ,Object["criterionName",3] or Object["criterionName",c(3,2)]]** [matrix(numeric)]: Gets the criterion names for the `Clusterization` contained in the object (all, the third line or the second of the third line, as for "`criterionValue`").
- Object["myCriterion" ]** [matrix(numeric)]: if "`myCriterion`" is a criterion name, gets all this criterion value for all the `Clusterization`.
- Object["calinski",c(4,5) ]** [numeric] Gets the fifth Calinski criterion of the "`c4`" list.
- Object["other" ]** [list] Gets the list contain in the field "`other`".

**Set [<-**

- Object["id" <-value]** [vecteur(character)]: Sets each individual Identifier to `value`
- Object["time" <-value]** [vecteur(numeric)]: Sets the times to `value`.
- Object["varName" <-value]** [character]: Sets the name of the variable to `value`)
- Object["trajMinSize" <-value]** [numeric]: Sets the limit for not excluding a trajectory with missing value to `value`.
- Object["traj" ]** [matrix(numeric)]: Sets all the trajectories' to `values`.
- Object["clusters","add" <-value]** [Clusterization]: Adds the `Clusterization` to the corresponding sublist (the sublist that contain `Clusterization` with that same number of clusters that the one currently added), then sorts the sublist by decreasing quality criterion value (highest criterion firsts).
- Object["clusters","clear" <-value]** [numeric]: Empties the sublist of `Clusterization` with `value` clusters.
- Object["clusters","clear" <-"all"]** [numeric]: Empties all the sublist of `clusters`.
- Object["clusters" <-value]** [Clusterization]: Short cut for `Object["clusters", "add"]<-value`.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)

Classes : [Clusterization](#), [LongData](#)

Methods : [clusterizLongData](#), [kml](#), [choice](#), [as.clusterizLongData](#)

Plot: [plot\(ClusterizLongData\)](#), [plotCriterion](#), [plotSubGroups\(ClusterizLongData\)](#), [plotAll\(ClusterizLongData\)](#)

**Examples**

```
#####
### Creation of some trajectories
mat <- matrix(c(1,2,3,1,NA,6,1,8,NA),3)

#####
### Creation of LongitudinalData
clustLd <- new("ClusterizLongData",id=c("1","2","3"),time=c(2,4,8),varName="Age",traj=mat,tra

#####
### get and set
clustLd["id"]
clustLd["time"]<- c(1,3,9)
clustLd["varName"]
clustLd["traj"]
clustLd["traj"][3,]<-c(2,7,9)

#####
### Creation of a clusterization
part <- partition(nbClusters=2,clusters=LETTERS[c(1,2,1)])
clus <- clusterization(xPartition=part,yLongData=clustLd)
```

```
#####
### Adding a clusterization to a clusterizLongData
(clustLd["clusters","add"] <- clus)

#####
### Removing all the clusterization from a clusterizLongData
clustLd["clusters","clear"] <- "all"
(clustLd)
```

---

```
exportClusterization
```

```
~ Function: exportClusterization ~
```

---

### Description

This function save all the information contained in a `Clusterization` object.

### Usage

```
exportClusterization(object, y, typeGraph = "bmp", col = "clusters", type = "l", co
```

### Arguments

<code>object</code>	[ <code>ClusterizLongData</code> ]: object containing the information that have to be saved
<code>y</code>	[ <code>couple(numeric)</code> ]: indicate which <code>Clusterization</code> shall be exported. The first numeric <code>y[1]</code> is the cluster number, the second <code>y[2]</code> is the rank of clusterization in the list <code>y[1]</code> .
<code>typeGraph</code>	[ <code>character</code> ]: indicates the type of the graph that will be exported. See <code>savePlot</code> for available options
<code>col</code>	[ <code>character</code> ], [ <code>numeric</code> ] or <code>vector[numeric]</code> : Specification of the plotting color of the individual trajectories. In addition to the standard possible values, <code>col="clusters"</code> can be used to color the individual trajectories according to their clusters.
<code>col.mean</code>	[ <code>character</code> ], [ <code>numeric</code> ] or <code>vector[numeric]</code> : Specification of the plotting color of the mean trajectories. In addition to the standard possible values, <code>col="clusters"</code> can be used to color each mean trajectories according to its clusters.
<code>main</code>	[ <code>character</code> ]: give the title of the graph.
<code>type</code>	[ <code>character</code> ]: what type of plot should be drawn for the individual trajectories ?
<code>type.mean</code>	[ <code>character</code> ]: what type of plot should be drawn for the mean trajectories ?
<code>cex</code>	[ <code>numeric</code> ]: fixes the size of the point on the mean trajectories.
<code>pch.mean</code>	[ <code>character</code> ]: specifies the symbol to be used as plotting point on the mean trajectories. Option <code>pch.mean="symbols"</code> or <code>pch.mean="letters"</code> can be used.
<code>pch.time</code>	[ <code>vector(numeric)</code> ]: specifies the time at which a point should be plot (useful if there is a important number of time, see <code>plot</code> for detail).
<code>...</code>	Graphical parameters to be passed to methods, see <code>plot(LongData)</code> and <code>par</code> for details.

## Details

`exportClusterization` is mainly used by the function `choice`. It exports the clusters' affectation (individual and its letters) in a file named "objectName-Clusters.csv", the information about the clusterization (algorithm used, clusters' number, convergence time, percentage in each cluster, criterion quality name and value, imputation method and starting condition) in another file named "objectName-Details.csv". It also saves two graphs, one with the trajectories, the other one with the subgroups.

## Value

Two files and two graphs.

## Author(s)

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

## English translation

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

## References

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

## Examples

```
#####  
### Creating a ClusterizLongData object, with 3 clusterizations (5 clusters each)  
dn <- as.cld(gald())  
kml(dn, 5, 3)  
  
### Exporting the second clusterization in pdf format  
try(exportClusterization(dn, c(5, 2), typeGraph="pdf"))
```

kml

~ Algorithm kml: K-means for Longitudinal data ~

## Description

kml is a new implementation of k-means for longitudinal data (or trajectories). This algorithm is able to deal with missing value and provides an easy way to re roll the algorithm several times, varying the starting conditions and/or the number of clusters looked for.

Here is the description of the algorithm. For an overview of the package, see [kml-package](#).

## Usage

```
kml(Object, nbClusters = 2:6, nbRedrawing = 20, saveFreq = 100,
     maxIt = 200, trajMinSize = 2, print.cal = FALSE,
     print.traj = FALSE, imputationMethod = "copyMean",
     distance, power = 2, centerMethod = meanNA, startingCond = "allMethods",
     distanceStartingCond = "euclidean", ...)
```

## Arguments

Object	[ClusterizLongData]: contains trajectories to clusterize as well as previous <a href="#">Clusterization</a> .
nbClusters	[vector(numeric)]: Vector containing the number of clusters with which kml must work. By default, nbClusters is 2:6 which indicates that kml must search partitions with respectively 2, then 3, ... up to 6 clusters. Maximum number of cluster is 52.
nbRedrawing	[numeric]: Sets the number of time that k-means must be re-run (with different starting conditions) for each number of clusters.
saveFreq	[numeric]: Long computations can take several days. So it is possible to save the object ClusterizLongData once in a while. saveFreq defines the frequency of the saving process. The ClusterizLongData is saved every saveFreq clusterization calculations. The object is saved in the file objectName.Rdata in the curent folder.
maxIt	[numeric]: Sets a limit to the number of iteration if convergence is not reached.
trajMinSize	[numeric]: The trajectories that include missing values can either be excluded or included. trajMinSize sets the minimum number of values that a trajectory must contain not to be excluded. For example, if the trajectories have 7 measurements (time=7) and trajMinSize is set to 3, the trajectory (5,3,NA,4,NA,NA,NA) will be included in the calculation while (2,NA,NA,NA,4,NA,NA) will be excluded. Please note that trajectories that are completely missing (0 present values) must always be excluded.
print.cal	[logical]: If TRUE, the quality criterion will be printed on screen during computation (if the number of redrawing is big, this can slow the overall calculation process).

<code>print.traj</code>	[logical]: If TRUE, each step of k-means is on screen during the calculation. This can slow the overall calculation process by a factor of 25, see "optimization" below.
<code>imputationMethod</code>	[character]: the calculation of quality criterion can not be done if some value are missing. <code>imputationMethod</code> define the method use to impute the missing value. It should be one of "LOCF", "LOCB", "linearInterpolation", "linearInterpolation2", "linearInterpolation3" or "copyMean". See <a href="#">imputation</a> for detail.
<code>distance</code>	[numeric <- function(trajjectory,trajectory)] function that computes the distance between two trajectories. If no function is specified, the Euclidian distance with Gower adjustment is used (Gower adjustment takes in accompanying missing value.) Using a classical distance speed up the overall calculation process by a factor 25, see "optimization" below.
<code>power</code>	[numeric]: power define the parameter of the Minkovski distance, if used.
<code>centerMethod</code>	[numeric <- function(vector(numeric))]: k-means algorithm computes the centers of each cluster. It is possible to personalize the definition of "center" by defining a function "centerMethod". This function should take a vector of numeric as argument and return a single numeric -the center of the vector-.
<code>startingCond</code>	[character]: specifies the starting condition. Should be one of "maxDist", "randomAll", "randomK" or "allMethods". See detail.
<code>distanceStartingCond</code>	[character]: some starting condition needs to compute the distance matrix of the trajectories. <code>distanceStartingCond</code> define the distance that will be use to calculate this matrix. It should be one of "euclidean", "maximum", "manhattan", "canberra", "binary" or "Minkowski".
<code>...</code>	For graphical parameters.

## Details

`kml` works on object of class `ClusterizLongData`. For each number included in `nbClusters`, `kml` computes a [Clusterization](#) then stores it in the field `clusters` of the object `ClusterizLongData` according to its number of clusters. The algorithm starts over as many times as it is told in `nbRedrawing`. By default, it is executed for 2, 3, 4, 5 and 6 clusters 20 times each, namely 100 times.

When a `Clusterization` has been found, it is added to the slot `clusters`. `clusters` is a list of 52 sublist called `c1`, `c2`, `c3` until `c52`. The sublist `cX` stores the all `Clusterization` with `X` clusters. Inside a sublist, the `Clusterization` are sorted from the biggest quality criterion to the smallest (the best are stored first).

Note that `Clusterization` are saved throughout the algorithm. If the user interrupts the execution of `kml`, the result is not lost. If the user run `kml` on an object then run `kml` again on the same object, the `Clusterization` that are computed the second time are added to the one already present in the object (unless you "clear" some list, see `Object["clusters", "clear"] <- value` in [ClusterizLongData](#)).

The possible starting conditions are "randomAll", "randomK" and "maxDist", as defined in [partitionInitialise](#). In addition, the method "allMethods" is a shortcut that run a "maxDist", a "randomAll" and "randomK" for all the other re rolling.

**Value**

A `ClusterizLongData` object, after having added some `Clusterization` to it.

**Optimisation**

Behind `kml`, there are two different procedures :

1. Fast: when the parameter `distance` is set to a classical distance (one of "euclidean", "maximum", "manhattan", "canberra", "binary" or "minkowski") and `print.traj` is set to `FALSE` (the default), `kml` call a C compiled (optimized) procedure.
2. Slow: when the user defines its own distance or if he wants to see the construction of the clusters by setting `print.traj=TRUE`, `kml` uses a R non compiled programmes.

The C procedure is 25 times faster than the R one.

So we advice to use the R procedure 1/ for trying some new method (like using a new distance) or 2/ to "see" the very first cluster construction, in order to check that every thing goes right, then to switch to the C procedure (like we do in `Example` section).

If for a specific use, you need a different distance, feel free to contact the author.

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in *Computational Statistics* (accepted on 11-11-2009)  
Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
Classes : [ClusterizLongData](#), [Clusterization](#)  
Methods : [clusterizLongData](#), [choice](#)

**Examples**

```
### Generation of some data
cld1 <- as.cld(generateArtificialLongData())

### We suspect 2, 3, 4 or 5 clusters, we want 3 redrawing.
# We want to "see" what happen (so printCal and printTraj are TRUE)
kml(cld1,2:6,3,printCal=TRUE,printTraj=TRUE)

### 4 seems to be the best. But to be sure, we try more redrawing 4 or 6 only.
# We don't want to see again, we want to get the result as fast as possible.
kml(cld1,c(4,6),10)
```

---

```
partitionInitialise
```

```
~ Function: partitionInitialise ~
```

---

**Description**

This function provide different way of setting the initial partition for an EM algorithm.

**Usage**

```
partitionInitialise(nbClusters, lengthPart, method = "randomK", matrixDist)
```

**Arguments**

nbClusters	[numeric]: number of clusters of that the initial partition should have.
lengthPart	[numeric]: number of individual in the partition.
method	[character]: one off "randomAll", "randomK" and "maxDist".
matrixDist	[matrix]: if the method "maxDist" is used, the function needs to know the matrix of the distance between each individual.

**Details**

Before alternating the phase Esperance and Maximisation, the EM algorithm needs to initialize a starting configuration. This initial partition has been proven to have an important impact on the final result and the convergence time.

This function provides different ways of setting the initial partition.

- randomAll: all the individual are randomly assigned to a cluster with at least one individual in each clusters.
- randomK: K individuals are randomly assigned to a cluster, all the other are not assigned (each cluster has only one individual).
- maxDist: K individuals are chosen. The two formers are the individual separated by the highest distance. The latter are added one by one, they are the "farthest" individual among those that are already been selected. "farthest" is the individual with the highest distance (min) to the selected individuals (if "t" are the individual already selected, the next selected individual is "i" such that  $\max_i(\min_t(\text{dist}(\text{IND}_i, \text{IND}_t)))$ )

**Value**

Object of class `Partition`.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
par(ask=TRUE)
#####
### Constrution of some longitudinal data
dn <- as.cld(gald())
plot(dn,type.mean="n",col=1)

#####
### partition using randomAll
pala <- partitionInitialise(3,lengthPart=200,method="randomAll")
plot(dn,pala)
palb <- partitionInitialise(3,lengthPart=200,method="randomAll")
plot(dn,palb)

#####
### partition using randomK
pa2a <- partitionInitialise(3,lengthPart=200,method="randomK")
plot(dn,pa2a)
pa2b <- partitionInitialise(3,lengthPart=200,method="randomK")
plot(dn,pa2b)

#####
### partition using maxDist
pa3 <- partitionInitialise(3,lengthPart=200,method="maxDist",
  matrixDist=as.matrix(dist(dn["traj"])))
plot(dn,pa3)
```

```

### maxDist is deterministic, so no need for a second example

#####
### Example to illustrate "maxDist" method on classical clusters
point <- matrix(c(0,0, 0,1, -1,0, 0,-1, 1,0),5,byrow=TRUE)
points <- rbind(point,t(t(point)+c(10,0)),t(t(point)+c(5,6)))
points <- rbind(points,t(t(points)+c(30,0)),t(t(points)+c(15,20)),t(-t(point)+c(20,10)))
plot(points,main="Some points")

paInit <- partitionInitialise(2,nrow(points),as.matrix(dist(points)),method="maxDist")
plot(points,main="Two farest points")
lines(points[!is.na(paInit["clusters"]),],col=2,type="p",lwd=3)

paInit <- partitionInitialise(3,nrow(points),as.matrix(dist(points)),method="maxDist")
plot(points,main="Three farest points")
lines(points[!is.na(paInit["clusters"]),],col=2,type="p",lwd=3)

paInit <- partitionInitialise(4,nrow(points),as.matrix(dist(points)),method="maxDist")
plot(points,main="Four farest points")
lines(points[!is.na(paInit["clusters"]),],col=2,type="p",lwd=3)

par(ask=FALSE)

```

---

```
plot,ClusterizLongData
```

*~ Function: plot for ClusterizLongData ~*

---

## Description

plot the trajectories of an object `ClusterizLongData` relatively to a `Clusterization`.

## Usage

```
## S4 method for signature 'ClusterizLongData,ANY':
plot(x,y,col="clusters",col.mean="clusters",main="",type="l",type.mean="",size=1,...
```

## Arguments

x	[ClusterizLongData]: Object containing the trajectories to plot.
y	[numeric] or [vector(numeric)]: Give the Clusterization to represent. If y is missing, the Clusterization with the highest quality criterion is selected. If y is a number, the first Clusterization of the sublist c-y is selected. If y is a couple of numeric, the y[2] th Clusterization of the sublist c-y[1] is selected.
col	[character], [numeric] or vector[numeric]: Specification of the plotting color of the individual trajectories. In addition to the standard possible values, col="clusters" can be use to color the individual trajectories according to their clusters.

<code>col.mean</code>	[character], [numeric] or vector[numeric]: Specification of the plotting color of the mean trajectories. In addition to the standard possible values, <code>col="clusters"</code> can be used to color each mean trajectories according to its clusters.
<code>main</code>	[character]: give the title of the graph.
<code>type</code>	[character]: what type of plot should be drawn for the individual trajectories.
<code>type.mean</code>	[character]: what type of plot should be drawn for the mean trajectories ?
<code>size</code>	[numeric]: If some point are added on the mean trajectories, <code>size</code> set the size of the point.
<code>...</code>	Graphical parameters to be passed to methods, see <code>plot(LongData)</code> and <code>par</code> for details.

**Details**

`plot` the trajectories of an object `ClusterizLongData` relatively to a `Clusterization`.

**Author(s)**

Christophe Genolini  
 PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
 INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
 Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
 University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
 Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
 Classes : [ClusterizLongData](#)  
 Plot: `plot`: [overview](#), [plotCriterion](#), [plotSubGroups\(ClusterizLongData\)](#), [plotAll\(ClusterizLongData\)](#)

**Examples**

```

clusLd <- as.cld(gald())
kml(clusLd,,1)
par(ask=TRUE)

### Default plotting
plot(clusLd)

### Only the trajectories in black
plot(clusLd,type="n",col.mean="1",type.mean="1")

### Only the mean trajectories, with letters (for publication ?)
plot(clusLd,type="n",col.mean="1",type.mean="b",cex=2)

### All at once.
plot(clusLd,col="clusters",col.mean="clusters",type.mean="1",legend=FALSE)

par(ask=FALSE)

```

---

```
plotAll, ClusterizLongData
```

*~ Function: plotAll for ClusterizLongData ~*

---

**Description**

plot the trajectories, the quality's criteria and the subgroups of an object [ClusterizLongData](#) on a single graph.

**Usage**

```

## S4 method for signature 'ClusterizLongData':
plotAll(x, y, print.cal = TRUE, print.traj = TRUE,
        print.sub = FALSE, allCrit = TRUE, nbCriterion = 100, col = 1,
        type = "l", col.mean = "clusters", type.mean = "b", main = "",
        size = 1, ylim = NA,
        col.sub = 1, type.sub = "l", col.mean.sub = "clusters",
        type.mean.sub = "b", main.sub = "", size.sub = 1, ylim.sub = NA,
        ...)

```

**Arguments**

x	[ClusterizLongData] Object containing the trajectories to plot.
y	[numeric] or [vector(numeric)] Give the Clusterization to represent. If y is missing, the Clusterization with the highest quality criterion is selected. If y is a number, the first Clusterization of the sublist c-y is selected. If y is a couple of numeric, the y[2] th Clusterization of the sublist c-y[1] is selected.

<code>print.cal</code>	[logical]: if TRUE, the quality criterion are printed.
<code>print.traj</code>	[logical]: if TRUE, a "main" graph, showing all the trajectories at once, is printed.
<code>print.sub</code>	[logical]: if TRUE, several sub-graphs, each containing a specific cluster of trajetories, are printed.
<code>allCrit</code>	[logical] shall all the quality criterion be display, or only the best one for each clusters' number ?
<code>nbCriterion</code>	[numeric]: gives an upper limit to the number of quality criterion to print.
<code>col</code>	[character], [numeric] or vector[numeric]: Specification of the plotting color of the individual trajectories. In addition to the standard possibles values, <code>col="clusters"</code> can be use to color the individual trajectories according to their clusters.
<code>col.mean</code>	[character], [numeric] or vector[numeric]: Specification of the plotting color of the mean trajectories. In addition to the standard possible values, <code>col="clusters"</code> can be used to color each mean trajectories according to its clusters.
<code>type</code>	[character]: what type of plot should be drawn for the individual trajectories.
<code>type.mean</code>	[character]: what type of plot should be drawn for the mean trajectories ?
<code>main</code>	[charater]: give the title of the graph.
<code>size</code>	[numeric]: If some point are added on the mean trajectories, <code>size</code> fixe the size of the point.
<code>ylim</code>	[couple(numeric)]: fixe the ylim parameter.
<code>col.sub</code>	[character], [numeric] or vector[numeric]: Specification of the plotting color of the individual trajectories on the sub-groups graph. In addition to the standard possible values, <code>col="clusters"</code> can be used to color the individual trajectories according to their clusters.
<code>type.sub</code>	[character]: what type of plot should be drawn for the individual trajectories on the sub-groups graph.
<code>col.mean.sub</code>	[character], [numeric] or vector[numeric]: Specification of the plotting color of the mean trajectories on the sub-groups graph. In addition to the standard possible values, <code>col="clusters"</code> can be used to color each mean trajectories according to its clusters.
<code>type.mean.sub</code>	[character]: what type of plot should be drawn for the mean trajectories on the sub-groups graph ?
<code>main.sub</code>	[charater]: gives the title of the sub-groups graph.
<code>size.sub</code>	[numeric]: If some point are added on the mean trajectories, <code>size</code> sets the size of the point on the sub-groups graph.
<code>ylim.sub</code>	[couple(numeric)]: sets the ylim parameter for the sub-groups graph.
<code>...</code>	Graphical parameters to be passed to methods, see <code>plot(LongData)</code> and <code>par</code> for details.

**Details**

For every `clusterizLongData` object, there is three possible graphical representation: quality criterion, all the trajectories at once and trajectories clusters by clusters. This function combines these three graphic outputs (or only two, or one, according to the three arguments `print.cal`, `print.traj` and `print.sub`). Thus every option (like `col.mean`,...) are duplicated : one is for the main graph (`col.mean`), the second is for the sub-graphs (`col.mean.sub`).

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)  
Web site: <http://christophe.genolini.free.fr/kml>

**See Also**

Overview: [kml-package](#)  
Classes : [ClusterizLongData](#)  
Plot: [plot\(ClusterizLongData\)](#), [plotCriterion](#), [plotSubGroups\(ClusterizLongData\)](#)

**Examples**

```
#####
### Data construction
dn <- as.cld(gald())
kml(dn, 2:5, 5)
kml(dn, 16, 5)
par(ask=TRUE)

### Default plotting
plotAll(dn)
plotAll(dn, legend=FALSE)

### Only the calinski criterion (same effect than plotCalinski(ld))
plotAll(dn, print.cal=TRUE, print.traj=FALSE, print.sub=FALSE)
```

```

### Groups and sub-groups
plotAll(dn, print.cal=FALSE, print.traj=TRUE, print.sub=TRUE, legend=FALSE)
plotAll(dn, print.cal=FALSE, print.traj=TRUE, print.sub=TRUE, col="black", type.mean="n", legend=F

### All at once
plotAll(dn, print.cal=TRUE, print.traj=TRUE, print.sub=TRUE)

### To see the clusterization with only 2 clusters
plotAll(dn, 2, print.cal=FALSE, print.traj=TRUE, print.sub=TRUE)

### To see the third clusterization with 16 clusters
plotAll(dn, c(16, 3), print.cal=FALSE, type.mean="b", type="n")

par(ask=FALSE)

```

---

plotCriterion      ~ Function: plotCriterion ~

---

## Description

This function graphically displays the quality criterion of all the [Clusterization](#) of a [ClusterizLongData](#) object.

## Usage

```
plotCriterion(x, nbCriterion = 100, allCrit = FALSE)
```

## Arguments

x	[ClusterizLongData]: object whose quality criterion should be displayed.
nbCriterion	[numeric]: if there is a big number of Clusterization, the graphical display of all of them can be slow. nbCriterion lets the user limit the number of criteria that will be taken in account.
allCrit	[logical]: if TRUE, up to nbCriterion for each possible clusters number are considered. If FALSE, only the best criterion for each possible clusters number are displayed.

## Details

This function display graphically the quality criterion (probably to decide the best clusters' number). It can either display all the criterion ; this is useful to see the consistency of the result : is the best clusterization obtain several time or only one ? It can also display only the best result for each clusters number : this helps to find the local maximum, which is classically used to chose the "correct" clusters' number.

## Value

No value are return. A graph is printed.

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####  
### Data generation  
dn <- as.cld(gald())  
  
### Trying several clusters number and several starting condition  
kml(dn)  
  
### Display the quality criterion, both way :  
par(mfrow=c(1,2))  
plotCriterion(dn)  
plotCriterion(dn,allCrit=TRUE)
```

---

updateClusterization

*~ Function: updateClusterization ~*

---

**Description**

This fonction converts an object Clusterization V0.9 to an object Clusterization V1.0.

**Usage**

```
updateClusterization(object)
```

**Arguments**

object            [Clusterization]: object to convert

**Details**

Objects used by `KmL` change with version V1.0. The new object structure is not compatible with the old one. This function lets the user convert its old object `Clusterization` to the new structure.

For that, just open R ; load the old object ; load the new `KmL` library ; use `updateClusterization`.

**Value**

`Clusterization` version V1.0

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####  
### Not executable  
### if oldObject is your old data :  
###  
#  
# newObject <- updateClusterization(oldObject)  
#
```

---

`updateClusterizLongData`*~ Function: updateClusterizLongData ~*

---

**Description**

This fonction converts an object `ClusterizLongData V0.9` to an object `ClusterizLongData V1.0`.

**Usage**

```
updateClusterizLongData(object)
```

**Arguments**

`object`            `[ClusterizLongData]`: object to convert

**Details**

Objects used by `KmL` change with version `V1.0`. The new object structure is not compatible with the old one. This function lets the user convert its old object `ClusterizLongData` to the new structure.

For that, just open R ; load the old object ; load the new `KmL` library ; use `updateClusterizLongData`.

**Value**

`ClusterizLongData` version `V1.0`

**Author(s)**

Christophe Genolini  
PSIGIAM: Paris Sud Innovation Group in Adolescent Mental Health  
INSERM U669 / Maison de Solenn / Paris

Contact author : <genolini@u-paris10.fr>

**English translation**

Raphaël Ricaud  
Laboratoire "Sport & Culture" / "Sports & Culture" Laboratory  
University of Paris 10 / Nanterre

**References**

Article "KmL: K-means for Longitudinal Data", in Computational Statistics (accepted on 11-11-2009)

Web site: <http://christophe.genolini.free.fr/kml>

**Examples**

```
#####  
### Not executable  
### if oldObject is your old data :  
###  
#  
# newObject <- updateClusterizLongData(oldObject)  
#
```

# Index

- \*Topic **chron**
  - as.clusterizLongData, 7
  - choice, 12
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
- \*Topic **classes**
  - Clusterization-class, 17
  - clusterizLongData, 19
  - ClusterizLongData-class, 21
- \*Topic **classif**
  - choice, 12
  - Clusterization-class, 17
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
- \*Topic **cluster**
  - choice, 12
  - Clusterization-class, 17
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
- \*Topic **datagen**
  - as.clusterizLongData, 7
- \*Topic **dplot**
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
- \*Topic **iplot**
  - choice, 12
  - kml-package, 2
  - plot, ClusterizLongData, 31
- plotAll, ClusterizLongData, 33
- \*Topic **methods**
  - affectIndivGeneralized, 6
  - updateClusterization, 37
  - updateClusterizLongData, 39
- \*Topic **models**
  - kml-package, 2
- \*Topic **nonparametric**
  - choice, 12
  - Clusterization-class, 17
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
- \*Topic **package**
  - kml-package, 2
- \*Topic **robust**
  - kml, 26
  - kml-package, 2
- \*Topic **spatial**
  - as.clusterizLongData, 7
  - choice, 12
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
- \*Topic **ts**
  - as.clusterizLongData, 7
  - choice, 12
  - Clusterization-class, 17
  - ClusterizLongData-class, 21
  - kml, 26
  - kml-package, 2
  - plot, ClusterizLongData, 31
  - plotAll, ClusterizLongData, 33
  - [, ClusterizLongData-method  
(ClusterizLongData-class),  
21
  - [, Clusterization

- (*Clusterization-class*), 17
- [, *Clusterization-method*
  - (*Clusterization-class*), 17
- [<-, *ClusterizLongData-method*
  - (*ClusterizLongData-class*), 21
- affectIndiv, 4, 6
- affectIndivGeneralized, 6
- as.cld, 2, 21
- as.cld(*as.clusterizLongData*), 7
- as.clusterizLongData, 2, 4, 7, 20, 23
- as.clusterizLongData, *data.frame-method*
  - (*as.clusterizLongData*), 7
- calculCenterGeneralized, 6, 9, 10
- calculMean, 5, 11, 11
- choice, 3, 4, 12, 20, 23, 25, 28
- choice, *ClusterizLongData-method*
  - (*choice*), 12
- choice-methods (*choice*), 12
- cld, 2, 21
- cld(*clusterizLongData*), 19
- Clusterization, 3, 4, 12, 14–16, 19, 21, 23, 24, 26–28, 31, 32, 36, 38
- Clusterization
  - (*Clusterization-class*), 17
- clusterization, 15, 17
- Clusterization-class, 17
- ClusterizLongData, 2, 4–11, 14, 17, 19, 20, 27, 28, 31–33, 35, 36, 39
- ClusterizLongData
  - (*ClusterizLongData-class*), 21
- clusterizLongData, 2, 4, 9, 19, 23, 28
- clusterizLongData, ANY, ANY, ANY, ANY, ANY-method
  - (*clusterizLongData*), 19
- clusterizLongData, missing, missing, missing, missing-method
  - (*clusterizLongData*), 19
- ClusterizLongData-class, 21
- exportClusterization, 24
- exportClusterization, *ClusterizLongData*
  - (*exportClusterization*), 24
- exportClusterization, *ClusterizLongData-method*
  - (*exportClusterization*), 24
- gald, 2
- generateArtificialLongData, 2, 9, 21
- imputation, 15, 17, 27
- kml, 2–4, 12, 14, 17, 19, 21, 23, 26
- kml, *ClusterizLongData-method*
  - (*kml*), 26
- kml-package, 4, 9, 14, 19, 20, 23, 28, 32, 35
- kml-method (*kml*), 26
- kml-package, 2, 26
- LongData, 21, 23
- par, 24, 32, 34
- Partition, 5, 6, 9, 11, 15, 17, 30
- partitionInitialise, 15, 27, 29
- partitionInitialise, numeric, numeric
  - (*partitionInitialise*), 29
- partitionInitialise, numeric, numeric-method
  - (*partitionInitialise*), 29
- plot, 13, 24
- plot (*plot, ClusterizLongData*), 31
- plot (Calinski), 9
- plot (*ClusterizLongData*), 4, 9, 14, 20, 23, 35
- plot (*LongData*), 24, 32, 34
- plot, *ClusterizLongData*, 31
- plot, *ClusterizLongData, ANY-method*
  - (*plot, ClusterizLongData*), 31
- plot, *ClusterizLongData, missing-method*
  - (*plot, ClusterizLongData*), 31
- plot, *ClusterizLongData, numeric-method*
  - (*plot, ClusterizLongData*), 31
- plot: overview, 32
- plotAll
  - (*plotAll, ClusterizLongData*), 33
- plotAll (*ClusterizLongData*), 4, 9, 20, 23, 32
- plotAll, *ClusterizLongData*, 33
- plotAll, *ClusterizLongData-method*
  - (*plotAll, ClusterizLongData*), 33
- plotCriterion, 14, 23, 35, 36

`plotCriterion`), 4, 32  
`plotCriterion, ClusterizLongData`  
    (*plotCriterion*), 36  
`plotCriterion, ClusterizLongData-method`  
    (*plotCriterion*), 36  
`plotCriterion-method`  
    (*plotCriterion*), 36  
`plotSubGroups`, 13  
`plotSubGroups (ClusterizLongData)`,  
    4, 9, 20, 23, 32, 35  
  
`savePlot`, 12, 24  
  
`updateClusterization`, 37  
`updateClusterizLongData`, 39