# Package 'funcy'

May 11, 2016

**Type** Package

**Title** Functional Clustering Algorithms

**Version** 0.8.5

**Date** 2016-05-03

**Description** Unified framework to cluster functional data according to one of seven models. All models are based on the projection of the curves onto a basis. The main function funcit() calls wrapper functions for the existing algorithms, so that input parameters are the same. A list is returned with each entry representing the same or extended output for the corresponding method. Method specific as well as general visualization tools are available.

**License** GPL-2

**Depends** flexclust, splines

**Imports** MASS, Matrix, fda, methods, wavethresh, kernlab, parallel, car, fields, calibrate, cluster, sm, caTools, plyr

**Suggests** scatterplot3d, Funclustering, funHDDC

**LazyLoad** yes

**Encoding** UTF-8

**NeedsCompilation** yes

**RoxygenNote** 5.0.1

**Author** Christina Yassouridis [aut, cre],
Dominik Ernst [ctb],
Madison Giacofci [ctb],
Sophie Lambert-Lacroix [ctb],
Guillemette Marot [ctb],
Franck Picard [ctb],
Nicoleta Serban [ctb],
Huijing Jiang [ctb],
Gareth James [ctb],
Catherine Sugar [ctb],
Hans-Georg Mueller [ctb],
Jie Peng [ctb],
Chiou Jeng-Min [ctb],
Pai-Ling Li [ctb]

**Maintainer** Christina Yassouridis <c_ya02@yahoo.de>

**Repository** CRAN

**Date/Publication** 2016-05-11 12:06:31

# R topics documented:



---

`accordance` *Accordance for cluster outcomes from different methods.*

---

## Description

Votes for the cluster with maximum agreement and shows relative number of methods that voted for this cluster.

## Usage

```
accordance(cls, ctrs=NULL, relabel=FALSE)
```

### Arguments

| | |
|---|---|
| cls | Numeric or character matrix of cluster outcomes for different methods. Cluster outcomes for each method are in columns. |
| ctrs | If relabel=TRUE, a list of centers for the different methods, if NULL, clusters must have been labeled in an appropriate way before. |
| relabel | If TRUE, clusters are relabeled according to maximum agreement between their center curves. ctrs must not be NULL in this case. |

### Details

If relabel=FALSE, clusters must have been relabeled in an appropriate way before, so that cluster labels representing the same or the most similar clusters are identical. If relabel=TRUE, centers must be given as list to ctrs. Each list entry consists of a matrix with cluster centers stored in columns.

### Value

| | |
|---|---|
| votedCluster | Cluster with maximum agreement between the methods. |
| accordance | Percentage of methods that voted for the cluster. |

### Author(s)

Christina Yassouridis

### Examples

```
##Generate dataset
k <- 3
set.seed(2511)
ds <- sampleFuncy(obsNr=30, timeNr=10, reg=TRUE, k=k, sd=.4)

##Cluster dataset
res1 <- funcit(methods=1:3, data=Data(ds), k=4, clusters=Cluster(ds))

##Retrieve clusters and centers
cl <- Cluster(res1)
ctr <- Center(res1)

accordance(cl, ctr, relabel=TRUE)
```

---

## bones *Bone density*

---

### Description

Relative spinal bone mineral density measurements on 261 North American adolescents. Each value is the difference in mineral density taken on two consecutive visits, divided by the average. The age is the average age over the two visits.

## Usage

```
data("bones")
```

## Format

General format "Format1" (see formatFuncy): A matrix of dimension 261x3.

## Source

http://statweb.stanford.edu/~tibs/ElemStatLearn/

## References

Hastie, Tibshirani and Friedman. The Elements of Statistical Learning (2nd edition) (2009). Springer-Verlag. 763 pages.

Gareth M. James and Trevor J. Hastie. Functional Linear Discriminant Analysis for Irregularly Sampled Curves. Journal of the Royal Statistical Society Series B. 63. 533–550. 2001

Gareth M. James and Catherine A. Sugar. Clustering for Sparsely Sampled Functional Data . Journal of the American Statistical Assocation. 98. 397–408. 2003

---

# CClust-methods *Methods for CClust object.*

---

## Description

Mehods for CClust objects.

## Arguments

Not documented, for internal use only.

## Value

Not documented, for internal use only.

## Author(s)

Christina Yassouridis

`dist2centers` *Calculates curve distances to the centers.*

## Description

Distance to all cluster centers is calculated for each curve. Dataset can be in general format "Format1" or matrix format "Format2" (see `formatFuncy`).

## Usage

```
dist2centers(data, centers)
```

## Arguments

| | |
|---|---|
| `data` | Dataset in `"Format1"` or `"Format2"`. |
| `centers` | Matrix with cluster center curves in columns. |

## Details

Calculates the sum of the squared distances between the curves and the cluster centers on the available time points and divides it by the total number of time points for the corresponding curve. If the dataset is in irregular format `"Format1"`, center curves must correspond to evaluations on the unique union of all time points.

## Value

Matrix of dimension nr_curves x nr_clusters.

## Author(s)

Christina Yassouridis

## Examples

```
##Generate dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=100, k=4, timeNrMin=5, timeNrMax=10, reg=FALSE)
data <- Data(ds)
##Get unique union of all time points
time <- sort(unique(data[,3]))
##Generate center curves
c1 <- sin(time)+rnorm(30)
c2 <- cos(time)+rnorm(30)
c3 <- time^2+rnorm(3)
centers <- cbind(c1,c2,c3)
##Calculate distance to the centers
dist2centers(data, centers)
```

## electricity *Electric power*

### Description

The electric power home consumers at Buenos Aires, Argentina in 2001. For 104 households, measurements were taken at each of the 96 subintervals of 15 min in every week day, Monday to Friday, during January 2001.

### Usage

```
data("electricity")
```

### Format

Regular format `"Format2"` (see `formatFuncy`): A matrix of dimension 261x104.

### References

Juan Antonio Cuesta-Albertos and Ricard Fraiman. Impartial trimmed k-means for functional data. Computational Statistics & Data Analysis. 51. 4864–4877. 2007

## formatFuncy *Switch between different data formats.*

### Description

Datasets can be stored in different formats.

**"Format1":** General format for regular and irregular data. One long matrix with curve-ID in first column, curve evaluations in second and time points in third column.

**"Format2":** Format for regular data only. Matrix of dimension nr_time x nr_curves.

**"Format3":** Format for regular and irregular datasets. List of three matrices, all with same dimension. One evaluation matrix `Yin` where curves are stored in rows. One matrix of time points `Tin`. One incidence matrix `isobs` with entry 1, if curve has evaluation at this time point.

### Usage

```
## S4 method for signature 'list,character'
formatFuncy(data, format="Format1", regTime=NULL)
## S4 method for signature 'matrix,character'
formatFuncy(data, format="Format1", regTime=NULL)
```

## Arguments

| | |
|---|---|
| data | Data in format "Format1", "Format2" or "Format3". |
| format | Format to transform data to. One of "Format1" or "Format3". "Format2" is possible only if dataset in data is regular. |
| regTime | Optional vector of time points if original data is in "Format2". |

## Details

Data, especially for irregular time points is often stored in different ways. To switch back and forth between data formats, formatFuncy can be used. For regular datasets in format "Format2", an optional vector of evaluation time points can be given as regTime.

## Value

Numeric matrix if format="Format1" or format="Format2". If format="Format3", a list of the three matrices Yin, Tin, isobs and two vectors N and t_all. N stores the number of time points for each curve and t_all is a vector of unique time points (see Arguments).

## Author(s)

Christina Yassouridis

## Examples

```
##Generate regular dataset
set.seed(2005)
ds <- sampleFuncy(obsNr=100, k=6, timeNr=20, reg=TRUE)
Data(ds)

##Format dataset to Format1
newdat <-formatFuncy(data=Data(ds), format="Format1")
newdat

##Back to matrix out of Format1
formatFuncy(newdat, format="Format2")

##To Format3
formatFuncy(newdat, format="Format3")

##Generate irregular dataset
set.seed(2005)
ds <- sampleFuncy(obsNr=100, k=5, timeNrMin=5, timeNrMax=10, reg=FALSE)

res <- formatFuncy(Data(ds), format="Format3", reg=FALSE)
res

##Back to Format1
formatFuncy(data=res, format="Format1", reg=FALSE)
```

fpca *Functional principal component analysis*

## Description

Performs a functional principal component analysis.

## Usage

```
fpca(data=data, dimBase=4, fpcCtrl=NULL)
```

## Arguments

| | |
|---|---|
| data | Dataset in "Format1" for irregular or "Format2" for regular time measurements (see formatFuncy). |
| dimBase | Dimension of the basis. |
| fpcCtrl | Control object, see fpcCtrl. |

## Details

A functional principal component analysis is executed. Therefore, a smoothed mean curve and a smoothed covariance matrix are built and upon these, eigenvalues and eigenvectors are calculated. If average=TRUE, data is stored in a sparse matrix of dimension (nr_curves x nr_union_time_points) and matrix operations speed up the calculation. However, if the curves do not have many common time points, average should be set to FALSE. If the smoothing parameter (either for the mean or for the covariance) specified in fpcCtrl led to NA-values in the smoothing process, it is automatically increased by 10% of its actual value and a warning is printed. This process is repeated until smoothing is possible.

## Value

| | |
|---|---|
| yreg | Smoothed dataset evaluated at union time points. |
| time | Vector of union time points. |
| meanfcn | Smoothed mean function. |
| covfcn | Smoothed covariance matrix. |
| base | Basis functions. |
| eigval | Eigenvalues. |
| coeffs | Coefficients for the basis functions. |
| varprop | Proportion of the variance. |

## Author(s)

Christina Yassouridis

## See Also

fpcCtrl

## Examples

```
##Generate a regular dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=40, timeNr=15, reg=TRUE)

##Execute functional principal component analysis
res <- fpca(Data(ds))
matplot(res$base,type='l')

##Generate an irregular dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=30, k=5, timeNrMin=5, timeNrMax=7, reg=FALSE)

##Execute functional principal component analysis
res <- fpca(Data(ds))
matplot(res$base, type='l', main="First 4 basis functions.")
```

---

# fpcCtrl-class *Class "fpcCtrl"*

---

## Description

Hyperparameters for functional principal component analysis. See fpca.

## Objects from the class

Objects can be created by calls of the form `new("fpcCtrl")`. In addition, named lists can be coerced to `fpcCtrl` objects, names are completed if unique.

## Slots

Objects of class `fpcCtrl` have the following slots:

`select`: Character string, one of `"automatic"` or `"manual"` specifying whether the bandwidth for smoothing is given or should be calculated based on the data.

`h1Dim`: If `select="manual"`, bandwidth for smoothing the mean.

`h2Dim`: If `select="manual"`, bandwidth for smoothing the covariance.

`sm1Dim`: Character string, name of the mean smoothing function. One of `"sm.regression"` or `"sm1"`.

`sm2Dim`: Character string, name of the covariance smoothing function. One of `"sm.regression"` or `"sm2"`.

`coeffsCalc`: Character string, specifying how to calculate the coefficients. One of `"estimate"` or `"integrate"`.

`nrMaxTime`: Maximum number of evaluation time points for the covariance matrix.

`average`: If TRUE, matrix calculation is used and speeds up the calculation if curves have many evaluation time points in common.

## Details

`"sm.regression"` is a nonparametric regression estimate from the R-package **sm**. `"sm1"` and `"sm2"` are kernel smoothers defined by Chiou2007. The coefficients for the basis functions can be computed by numerical integration (`coeffCalc="integrate"`) or by a sparse estimation defined by Mueller2005 (`coeffCalc="estimate"`).

## Author(s)

Christina Yassouridis

## References

Chiou J-M and Li P-L. "Functional clustering and identifying substructures of longitudinal data". Journal of the Royal Statistical Society: Series B. 69 (4). pp. 679–699. 2007

F. Yao and H-G. Mueller and J-L. Wang. Functional Data Analysis for Sparse Longitudinal Data. Journal of the American Statistical Association. 100 (470). 577–590. 2005

## Examples

```
##Have a look at the defaults
new("fpcCtrl")

##Coerce a list
mycont = list(select="automatic", sm1Dim="sm1", sm2Dim="sm2", nrMaxTime=20)
as(mycont, "fpcCtrl")

##Default values for coefficients calculation procedure
new("fpcCtrl")@coeffsCalc
```

---

# funcit *Functional Cluster Analysis*

---

## Description

Main function for clustering functional data according to one or several of seven algorithms.

## Usage

```
funcit(data, k, methods=c("fitfclust","distclust", "iterSubspace",
       "funclust", "funHDDC", "fscm", "waveclust"), seed=NULL, regTime=NULL,
       clusters=NULL, funcyCtrl=NULL, fpcCtrl=NULL, parallel=FALSE,
       save.data=TRUE, ...)
```

## Arguments

| | |
|---|---|
| data | Data in format "Format1" or format "Format2" (see formatFuncy). |
| k | Number of clusters. |
| methods | **"fitfclust":** Model based cluster algorithm - based on a functional mixed mixture model. Allows irregular measurements, eigenbasis possible.<br>**"distFPCA":** Cluster algorithm - based on a distance measure. Allows irregular measurements, eigenbasis possible.<br>**"iterSubspace":** Model based cluster algorithm - based on a subspace projection. Allows irregular measurements, eigenbasis possible, dimension between clusters can vary.<br>**"funclust":** Model based cluster algorithm - based on a functional mixed mixture model.<br>**"funHDDC":** Model based cluster algorithm - based on a functional mixed mixture model. Dimension between clusters can vary.<br>**"fscm":** Model based cluster algorithm - based on a functional mixed mixture model. Curves can dependent on location. A matrix location is then an optional input parameter (see Details).<br>**"waveclust":** Model based cluster algorithm - based on a functional mixed mixture model. Wavelet basis is the only possible.<br>For a detailed description of the methods please see the references. |
| seed | Seed for initial clustering. See funcyCtrl. |
| regTime | If data is in "Format2", optional vector representing the time points (see formatFuncy). If regTime=NULL and format="Format2", equidistant time points from 1 to number of curves are used. |
| clusters | Optional vector of true cluster labels. |
| funcyCtrl | A control object of class funcyCtrl. If a model based clustering algorithm is used, further parameters can be specified by using the extended class fpcCtrlMbc. |
| fpcCtrl | A control object of class fpcCtrl. Only used for eigenbasis calculation (baseType="eigenbasis" in funcyCtrl). |
| parallel | If TRUE, package **parallel** is used for parallel computing. |
| save.data | Save a copy of the data in the return object? Must be set to TRUE in order to use plot function plot. |
| ... | Additional optional model specific parameters. Works only if exactly one method is called in methods. The parameters are the following:<br>**"fitfclust"**<br>**p:** Rank of the covariance matrix $\Gamma$, must be at least dimBase.<br>**pert:** Adds a ridge term to the least squares fit, helps if only few obeservations per curve were registered.<br>**"distclust"**<br>**method:** One of "hclust" or "pam" specifying how distance matrix is processed.<br>**"iterSubspace"** |

**simplify:** FALSE, if curve affiliation is tested again by projecting the curve onto the current subspace created without the current curve (leave-one-out-curve-estimation).

**"funclust"**

**nbInit:** The number of small-EM used to determine the intialization of the main EM-like algorithm.

**nbIterInit:** The maximum number of iterations for each small-EM.

**"funHDDCWrapper"**

**model:** The chosen model among "AkjBkQkDk", "AkjBQkDk", "AkBkQkDk","AkBQkDk","ABkQkDk","ABQkDk". See (Bouveyron & Jacques, 2011) for details.

**"fscm"**

**location:** A two-dimensional matrix of the curve locations (coordinates).

**knn:** Number of neighbours each curve depends on.

**useCode:** "R" or "C". If C is installed, a lot faster than R.

**verbose:** TRUE, if number of iterations and sigma, theta and f are to be printed.

**"waveclust"**

**gamma:** One of "group", "scale.location", "group.scale.location" or "constant".

**init:** One of "rEM" or "SEM" for random or stochastic EM.

**plotLoglik:** TRUE, if loglikelihood is to be plotted.

## Details

funcit is the core function to execute one or more methods to cluster functional data. Functional data can be measured on a regular or on an irregular grid. While for regular datasets, all curves are measured on the same time points, for irregular datasets, number or/and location of time points can differ (see formatFuncy for different formats). Only algorithms "fitfclust","distclust" and "iterSubspace" are applicable to irregular datsets. All methods are based on the projection of the curves onto a basis defined in funcyCtrl and building mixed effects models of the basis coefficients.

## Value

Returns an object of class funcyOutList.

## Author(s)

Christina Yassouridis

## References

**"fitfclust":** Gareth James and Catherine A. Sugar. Clustering for Sparsely Sampled Functional Data. Journal of the American Statistical Association. 98 (462). 297–408. 2003

**"distFPCA":** Jie Peng and Hans-Georg Mueller. Distance-based clustering of sparsely observed stochastic processes, with applications to online auctions. The Annals of Applied Statistics. 2 (3). 1056–1077, 2008

**"iterSubspace":** Chiou Jeng-Min and Pai-Ling Li. Functional clustering and identifying substructures of longitudinal data. Journal of the Royal Statistical Scociety: Series B. 69 (4). 679–699. 2007

**"waveclust":** Madison Giacofci and Sophie Lambert-Lacroix and Guillemette Marot and Franck Picard. Wavelet-based clustering for mixed-effects functional models in high dimension. Biometrics. 69. 31–40. 2011

**"fscm":** Nicoleta Serban and Huijing Jiang.Clustering Random Curves Under Spatial Interdependence With Application to Service Accessibility. Technometrics. 54 (2). 108–119. 2012

**"funclust":** Julien Jacques and Cristian Preda. Funclust: a curves clustering method using functional random variables density approximation. Neurocomputing. 112. 164–171. 2013

**"funHDDC":** Charles Bouveyron and Julien and Jacques. Model-based clustering of time series in group-specific functional subspaces. Advances in Data Analysis and Classification. 5 (4). 281–300. 2011

## Examples

```
##Cluster the data with methods for regular sets
##Sample a regular dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=50, k=4, timeNr=8, reg=TRUE)

##Cluster the functions with all available methods.
res <- funcit(data=Data(ds), clusters=Cluster(ds),
              methods=c(1,2,3,4), seed=2404,
              k=4)
summary(res)
Cluster(res)

##Additional method specific parameters for method fitfclust
res <- funcit(data=Data(ds), clusters=Cluster(ds), methods="fitfclust", seed=2405,
              k=4, p=5, pert=0)


##Cluster the data with methods for irregular sets
##Sample an irregular dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=50, k=4, timeNrMin=4, timeNrMax=8,
                  reg=FALSE)
data <- Data(ds)
clusters <- Cluster(ds)

res <- funcit(data=data, clusters=clusters,
              methods=c("fitfclust","distclust", "iterSubspace"), seed=2406,
              k=4, parallel=TRUE)

summary(res)
Cluster(res)
plot(res)

##Two reallife examples
```

```
## Not run:
data("genes")
data <- genes$data
clusters <- genes$clusters

##Cluster the functions with all available methods.
res <- funcit(data=data, clusters=clusters,
              methods=1:7, seed=2404,
              k=4)
summary(res)
Cluster(res)

## End(Not run)

## Not run:
data("electricity")
res <- funcit(data=electricity, methods=c("fitfclust","distclust",
"waveclust"), seed=2406, k=5, parallel=TRUE)
plot(res, legendPlace="topleft")

## End(Not run)
```

---

`funcyCtrl` *Class* `"funcyCtrl"`

---

## Description

Hyperparameters for functional cluster algorithms.

## Objects from the Class

Objects can be created by calls of the form `new("funcyCtrl")`. In addition, named lists can be coerced to `funcyCtrl` objects, names are completed if unique (see examples).

## Slots

Objects of class `funcyCtrl` have the following slots:

`baseType`**:** Type of basis functions, one of `"eigenbasis"`, `"splines"`, `"exponential"`, `"fourier"` or `"power"`.

`dimBase`**:** Dimension of the basis functions.

`flexDim`**:** If `TRUE`, dimension can vary between clusters (if supported by the algorithm). `dimBase` is therefore the maximum dimension.

`init`**:** Algorithm for initial clustering, one of `"kmeans"`, `"random"` or `"hclust"`.

`nrep`**:** Number of replications for initial clustering.

`seed`**:** Seed number.

`thd`**:** Threshold if fpca was integrated into method.

redDim: Reduced dimension if coefficients are additionally projected onto lower subspace.

Objects of class funcyCtrlMbc inherit from funcyCtrl and have the following additional slots:

eps: Convergence threshold for EM-algorithm.

maxit: Maximum number of iterations.

hard: Hard classification?

## Author(s)

Christina Yassouridis

## See Also

funcit

## Examples

```
##Show slots
showClass("funcyCtrl")

##Define new parameters
mycont = new("funcyCtrl", baseType="fourier", dimBase=4, flexDim=TRUE,
init="hclust")
mycont
```

---

# funcyOut-class *Class* "funcyOut"

---

## Description

List entry of funcyOutList for an object created by calls to the function funcit.

## Slots

control: Object of class funcyCtrl.

methodName: Name of the method.

kOut: Number of output clusters (can be smaller than k, if method did not find k clusters).

dimBaseOut: Output dimensions of the basis functions (only relevant if flexDim in funcyCtrl is TRUE).

time: Vector of measurement time points.

cluster: Vector of cluster outcomes.

centers: Matrix of center functions with centers stored in columns.

props: Vector of cluster proportions.

dist2centers: Distance to the centers. Matrix of dimension nr_curves x nr_clusters.

cldist: Matrix of dimension nr_curves x 2. Distances to closest and second closest cluster center.

`calcTime`: Calculation time, object of class "`proc_time`".

`plotParams`: Plot parameters, only relevant for call to `plot`.

`correctCl`: Numeric, Rand index if correct clusters were given as input.

Objects of class `funcyOut-iterSubspace` inherit from `funcyOut` and have the following additional slots:

`groupDimBase`: Cluster specific dimensions.

`prms`: List of model specific parameters.

`nrIter`: Number of iterations.

Object of class `funcyOut-Mbc` inherit from `funcyOut-iterSubspace` and can have the following additional slots:

`probs`: Class probabilities for each curve, matrix of dimension nr_curves x nr_clusters.

`AIC`: AIC.

`BIC`: BIC.

`loglik`: Log-likelihood.

Objects of class `funcyOutMbc-fclust` inherit from `funcyOut-Mbc` and can have the following additional slots:

`fit`: Output needed for the method specific plots.

Objects of class `funcyOutMbc-fscm` inherit from `funcyOut-Mbc` and can have the following additional slots:

`trends`: Cluster trends.

`location`: Location matrix of the curves.

## Author(s)

Christina Yassouridis

## See Also

`funcyOutList`

funcyOutList-class *Class* "funcyOutList"

## Description

Return object, created by calls of the function funcit.

## Slots

**call:** Method call of funcit.

**models:** List of all funcyOut-objects.

**data:** Input data.

**timeNr:** Number of time points.

**reg:** Regular or irregular data (see formatFuncy).

**k:** Number of clusters.

**methodName:** Method names.

**allClusters:** Matrix of all cluster results. Result for each method in column.

**randIndex:** Matrix of Rand indices showing the similarity between the methods. If true cluster membership was given, correct classification on diagonal.

**votedCluster:** Cluster, which majority of methods voted for.

**accordance:** Percentage of methods voting for the voted Cluster.

## Author(s)

Christina Yassouridis.

## See Also

funcyOut

## Examples

```
set.seed(2808)
ds <- sampleFuncy(obsNr=30, k=4, timeNr=7)
data <- Data(ds)
clusters <- Cluster(ds)

res <- funcit(data=data, clusters=clusters, seed=2808,
              methods=1:5, k=4, parallel=TRUE)
class(res)
summary(res)
```

funcyOutList-methods *Methods for the output of* funcit*, an object of class* funcyOutList.

## Description

Apply a function to funcyOutList which is the result from calling the function funcit.

## Usage

```
## S4 method for signature 'funcyOutList'
calcTime(object)
## S4 method for signature 'funcyOutList'
Center(object)
## S4 method for signature 'funcyOutList'
Cluster(object)
## S4 method for signature 'funcyOutList'
props(object)
## S4 method for signature 'funcyOutList,ANY'
randIndex(x)
## S4 method for signature 'funcyOutList'
summary(object)
```

## Arguments

| | |
|---|---|
| x, object | object funcyOutList as a result of function funcit. |

## Value

**calcTime** Numeric matrix of dimension nr_methods x 5. Calculation times for the different methods.

**Cluster** Numeric matrix of dimension nr_curves x methods. Cluster results for all methods.

**Center** List of matrices of cluster centers for the different methods. Centers are stored in columns.

**props** data.frame, proportion of the clusters for all methods. Can include NAs if a method reduced cluster number.

**randIndex** signature(x="funcyOutList", y="missing"): Quadratic matrix of Rand indices showing the similarity between the methods. If true cluster membership was given, correct classification on diagonal.

**summary** Summary showing method call, custer proportions, Rand indices and calculation time.

## Author(s)

Christina Yassouridis

## genes *Gene expression profiles*

### Description

A subet of the Drosophila life cycle gene expression data of Arbeitman et al. (2002). The original data set contains 77 gene expression profiles during 58 sequential time points from the embryonic, larval, and pupal periods of the lifecycle.

### Usage

```
data("genes")
```

### Format

Data in format "Format2" (see `formatFuncy`). A matrix of dimension 44x77, a cluster vector of length 44.

### Source

GDS191 in Gene Expression Ominibus http://www.ncbi.nlm.nih.gov/geo/

### References

J.-M. Chiou and P.-L. Li J. R. Functional clustering and identifying substructures of longitudinal data. Statist. Soc. B. Volume 69. 679–699. 2007

Arbeitman, M.N and al. Gene expression during the life cycle of Drosophila melanogaster. Science. 297(5590). 2002

## getUniCl *Unique cluster labels*

### Description

For repeated measurements on the same ID, build unique cluster labels according to the vector of IDs.

### Usage

```
getUniCl(id, clusters, reduce=TRUE)
```

### Arguments

| | |
|---|---|
| `id` | Vector of IDs. |
| `clusters` | Vector of cluster labels. |
| `reduce` | `TRUE` if cluster labels shall be reduced according to unique IDs. `FALSE` if cluster labels shall be duplicated according to IDs. |

## Details

Data might have been stored in 4 columns: curveID, curve evaluations, time points and cluster labels. Cluster labels were therefore repeated for each curve evaluation point. Method `funcit` accepts dataset only in formats "Format1" and "Format2" and an optional vector of cluster true labels `clusters` of length nr_curves. `getUniCl` can be applied to clumns curveID and repeated labels to reduce them to the number of curves.

## Value

A vector of either reduced (`reduce=TRUE`) or duplicated (`reduce=FALSE`) cluster labels.

## Author(s)

Christina Yassouridis

## Examples

```
##Generate dataset
nr_time <- sample(1:5, 100,  replace=TRUE)
clusters <- sample(1:4, 100, replace=TRUE)
IDs <- rep(1:100, nr_time)
cls <- rep(clusters,nr_time)
IDs
cls

##Get reduced cluster vector according to IDs
unicl <- getUniCl(IDs,cls)
unicl

##Get original cluster vector from reduced one
dupcl <- getUniCl(IDs,unicl, reduce=FALSE)
dupcl
```

---

`label2lowerk` *Relabeling of clusters.*

---

## Description

Relabels arbitrary cluster labels for k classes to 1 to k.

## Usage

```
label2lowerk(cluster)
```

## Arguments

| | |
|---|---|
| `cluster` | Original cluster labels. |

## Value

New cluster labels.

## Author(s)

Christina Yassouridis

## Examples

```
##Generating cluster labels for 4 clusters
cl <- rep(sample(2:10, 4), sample(3:5,4, replace=TRUE))
cl

##Labels them to 1:4
label2lowerk(cl)
```

---

`lowflow` *Gauges in Austria*

---

## Description

The dataset consists of 82 gauges in Upper Austria where streamflow minima were identified each winter during the years (1976-2008). `lowflow` is a list of data in format "Format2" (see `formatFuncy`) and a location matrix of measurements.

## Usage

```
data("lowflow")
```

## Format

Regular format `"Format2"` (see `formatFuncy`): A matrix of dimension 33x82.

## References

G. Laaha, D. Koffler, K. Haslinger, W. Schöner, J. Parajka, A. Viglione, J. Zehetgruber, C. Yassouridis, and G. Blöschl. Assessing climate impacts on low flows and droughts combining upward and downward approaches. 10th International Conference on Geostatistics for Environmental Applications. JUL 9-11. Paris, 2014

makeCommonTime *Create appropriate evaluation time points.*

## Description

Create an appropriate grid of common time points based on the available time points from the curves.

## Usage

```
makeCommonTime(data, timeNr, plot = TRUE)
```

## Arguments

| | |
|---|---|
| data | Dataset in general format Format1 (see funcit). |
| timeNr | Number of time points. |
| plot | If TRUE, original and new time points are plotted. |

## Details

The evaluation time points of all curves together are clustered and new time points are represented by the cluster centers.

## Value

A vector of new time points.

## Author(s)

Christina Yassouridis

## Examples

```
##sample an irregular dataset
set.seed(2804)
ds <- sampleFuncy(obsNr=100, k=4, timeNrMin=5, timeNrMax=10, reg=FALSE)
makeCommonTime(Data(ds), 10)
```

plot-methods *Methods for Function* plot *in Package* **funcy**

## Description

Plots clustered curves and/or cluster centers and other results.

## Usage

```
## S4 method for signature 'funcyOut,missing'
plot(x, y, data, type="all",
     showLegend=TRUE, legendPlace="bottomleft", main, ...)

## S4 method for signature 'funcyOutList,missing'
plot(x, y, data=NULL,
     select=NULL, type="all", showLegend=TRUE,
     legendPlace="bottomleft", main, ...)
```

## Arguments

| | |
|---|---|
| x | An object of class "funcyOut" or "funcyOutList". |
| y | Not used. |
| data | Data to include in plot. If the cluster object x is of type "funcyOutList" that was created with save.data=TRUE, then these are used by default. |
| select | Select the methods, you want to generate the plot for. |
| type | Plot type, see details. |
| showLegend | If TRUE, cluster legend is shown. |
| legendPlace | Legend placement. |
| main | Plot title, can be missing. |
| ... | Further plotting parameters |

## Details

If data was clustered by funcit with save.data=TRUE, different plots can be used. Some plots are available for all methods, others depend on method which was used. The plot types are listed below. If method specific plots are used, method must be extracted by select=method name, see examples.

**"all methods":**

**all:** Plots data and cluster centers.

**centers:** Plots only cluster centers.

**shadow:** Creates a shadow plot (see function shadow in package **flexclust** - Leisch 2010).

**dist2centers:** Multiple plots for each cluster. Thickness of lines corresponds to the proximity to the cluster centers. Thicker lines means curve is closer to its center.

**fpc:** Only if `baseType="eigenbasis"` in `funcyCtrl`. Plots the smoothed mean function, covariance matrix and eigenbasis.

**"fitfclust":**

**discrim** Plots discriminant functions to show the time points of maximum discrimination between clusters (see James2003).

**conf** Plots confidence intervals for the curves.

**"fscm":**

**overview:** Plots curve locations, temporal trends and overall trends (see Serban2012). For the spatial coefficients, dots are colored according to spatial dependency from yellow to blue. Darker dots mean stronger depencency.

## References

Friedrich Leisch. Neighborhood graphs, stripes and shadow plots for cluster visualization. Statistics and Computing. 20(4). 457–469. 2010

Gareth James and Catherine A. Sugar. Clustering for Sparsely Sampled Functional Data. Journal of the American Statistical Association. 98 (462). 297–408. 2003

Nicoleta Serban and Huijing Jiang.Clustering Random Curves Under Spatial Interdependence With Application to Service Accessibility. Technometrics. 54 (2). 108–119. 2012

## Examples

```
set.seed(2804)
ds <- sampleFuncy(obsNr=60, k=4, timeNrMin=5, timeNrMax=10, reg=FALSE)
data <- Data(ds)
clusters <- Cluster(ds)
res <- funcit(data=data, clusters=clusters,
              methods=c("fitfclust","distclust", "iterSubspace") ,
              k=4, parallel=TRUE)
plot(res)
plot(res, select="fitfclust", type="conf")
plot(res, select="fitfclust", type="discrim")
plot(res, select="distclust", type="shadow")
```

---

`plotFuncy` *Plot functional data.*

---

## Description

Plot functional data, color curves according to cluster labels and add center curves.

## Usage

```
## S4 method for signature 'matrix'
plotFuncy(data, regTime=NULL, col=NULL,
          ctr=NULL, ctrOnly=FALSE, ctrCols=NULL, showLegend=TRUE,
          legendPlace="bottomleft", lty=3, lwd=NULL, xlim=NULL, ylim=NULL, ...)

## S4 method for signature 'sampleFuncy'
plotFuncy(data, regTime=NULL, col=NULL,
          ctr=NULL, ctrOnly=FALSE, ctrCols=NULL, showLegend=TRUE,
          legendPlace="bottomleft", lty=3, lwd=NULL, xlim=NULL, ylim=NULL, ...)
```

## Arguments

| | |
|---|---|
| data | For irregular time measurements, a matrix consisting of curveID in first column, curve evaluations in second and time evaluation points in third column (Format1). For regular time measurements, a matrix consisting of curves in rows and time evaluations in columns (Format2). Alternatively an object created by sampleFuncy. |
| regTime | If Format2, optional vector representing the evaluation time points. If regTime=NULL equidistant time points from 1 to number of curves are used. |
| col | Numeric vector of cluster labels. |
| ctr | Numeric matrix of cluster centers |
| ctrOnly | Plot only centers, no curves? |
| ctrCols | Numeric vector specifying the color of the center curves. |
| showLegend | Whether to show cluster legend. |
| legendPlace | Where to place cluster legend. |
| lty | Line type of the functional data. |
| lwd | Line width of the functional data. |
| xlim | Range of x-axis. |
| ylim | Range of y-axis. |
| ... | Further plotting parameters. |

## Author(s)

Christina Yassouridis

## Examples

```
##cluster the data with methods for regular sets
##sample a regular dataset
set.seed(2001)
ds <- sampleFuncy(obsNr=40, k=4, timeNr=20, reg=TRUE)
data <- Data(ds)
clusters <- Cluster(ds)
```

```
##plot sampled functions.
plotFuncy(data, col=clusters, lty=1, showLegend=TRUE, legendPlace="topleft")

##sample an irregular dataset
set.seed(2001)
ds <- sampleFuncy(obsNr=40, k=4, timeNrMin=2, timeNrMax=10, reg=FALSE)
data <- Data(ds)
clusters <- Cluster(ds)

##plot sampled functions.
plotFuncy(data, col=clusters, lty=1, showLegend=TRUE, legendPlace="topleft")


##cluster functions with method fitfclust
res <- funcit(data=data, clusters=clusters,
              methods="fitfclust",
              k=4)

##plot including the centers
plotFuncy(data, col=clusters, ctr=Center(res), showLegend=TRUE, legendPlace="topleft")
```

---

`regFuncy` *Converts irregular data into regular.*

---

## Description

Converts irregular data into regular by projecting it to a basis or interpolating it.

## Usage

```
regFuncy(data, timeNr = 10, method = "project", baseType = NULL, nbasis = 4, plot = TRUE)
```

## Arguments

| | |
|---|---|
| `data` | Dataset in "Format1" (see `funcit`). |
| `timeNr` | Number of time points the regular dataset shall be evaluated at. |
| `method` | Method to tranform regularization with one of "project", "interpolate", "pace". See Details. |
| `baseType` | Base type. Only used if method="project". |
| `nbasis` | Number of basis functions. |
| `plot` | Plot the result? |

## Details

Data is either interpolated or projected to a basis. If method="pace", regularization is done via functional principal component analysis by conditional estimation (Müller2005). This is especially useful if the dataset is sparse. "pace" is implemented in the supplementary package "funcyOctave" which is loaded automatically but works only on Unix-platforms. For all methods, curve evaluation takes place on time points calculated by `makeCommonTime`.

## Value

| | |
|---|---|
| `data` | Numeric matrix of "Format1" (see `formatFuncy`). |
| `time` | Vector of evualuation time points. |

## Author(s)

Christina Yassouridis

## References

F. Yao and H.G. Müller and J.L. Wang. Functional data analysis for sparse longitudinal data. J. American Statistical Association. 100. 577–590. 2005 URL: http://www.stat.ucdavis.edu/PACE/

## Examples

```
##Generate irregular dataset
set.seed(2705)
ds <- sampleFuncy(reg=FALSE, obsNr=100, timeNrMin=5, timeNrMax=10)


reg <- regFuncy(Data(ds), timeNr=10, baseType="splines",nbasis=5, method="project")

reg <- regFuncy(Data(ds),timeNr=10, method="interpolate")

## Not run:
reg <- regFuncy(Data(ds), timeNr=10, baseType="eigenbasis", nbasis=5,
method="project")

## End(Not run)
```

---

# relabel *Relabel cluster IDs.*

---

## Description

Relabels cluster IDs of two or more cluster configurations according to the minimal distance between their centers.

## Usage

```
relabel(cl1, cl2, ctr1, ctr2)
relabelMethods(methodNames=NULL, cls, ctrs)
```

## Arguments

| | |
|---|---|
| cl1 | Cluster IDs of the first configuration. |
| cl2 | Cluster IDs of the second configuration. |
| ctr1 | Numeric matrix of cluster centers of the first configuration. |
| ctr2 | Numeric matrix of cluster centers of the second configuration. |
| methodNames | Character vector of names for the different cluster methods. |
| cls | Numeric or character matrix of cluster outcomes for different methods. |
| ctrs | List of centers for the different methods. |

## Details

The two configurations for the method relabel do not have to have the same number of observations neither the same number of clusters. The configuration with less observations has to be put on place cl1.
relabelMethods can be used for more than two configurations. Cluster outputs must therefore be saved in a matrix cls and thus have the same number of observations. The relabeling works only correctly if the number of classes is the same. The relabeling of the methods follows the following scheme:

1. Methods are sorted in that way that one of the two most similars is on first place.
2. Methods are successively added in the order of the highest similarity to one of the already added methods.
3. Once the order is fixed, the methods are relabeled after the ones they are most similar to.

## Value

For method relabel:

| | |
|---|---|
| recode | Recoding scheme of the second cluster labels cl2. |
| cluster | New cluster labels for cl2. |
| centers | Cluster centers ctrs2 in the new order. |

For method relabelMethods:

| | |
|---|---|
| allClusters | Matrix of new cluster labels. |
| allCenters | List of cluster centers in the new order. |
| fromTo | Recoding scheme of the methods. |

## Author(s)

Christina Yassouridis

## Examples

```
##Generate dataset
k <- 6
set.seed(2004)
ds <- sampleFuncy(obsNr=50, timeNrMin=3, timeNrMax=10, reg=FALSE, k=k, sd=.5)

##Cluster with different methods
res1 <- funcit(methods="fitfclust", data=Data(ds), k=k, reg=FALSE)
res2 <- funcit(methods="iterSubspace", data=Data(ds), k=k, reg=FALSE)
res3 <- funcit(methods="distclust", data=Data(ds), k=k, reg=FALSE)

##Relabel two configurations
relabel(Cluster(res3),Cluster(res1),Center(res3),Center(res1))

##Make matrix of clutser configurations
cls <- cbind(Cluster(res1),Cluster(res2),Cluster(res3))
##Make list of Centers
ctrs <- list(Center(res1), Center(res2), Center(res3))

##Relabel cluster configurations
rel <- relabelMethods(cls=cls, ctrs=ctrs)

##Compare
cls
rel$allClusters
```

---

`rIMethods` *Rand index for cluster configurations of different methods.*

---

## Description

Calculates a matrix of Rand indices for the configurations of all cluster method combinations.

## Usage

```
rIMethods(methodNames = NULL, cls, trueCluster = NULL)
```

## Arguments

| | |
|---|---|
| `methodNames` | Character Vector of names for the methods. |
| `cls` | Numeric or character matrix of cluster outcomes for different methods. |
| `trueCluster` | Vector of true clusters if knows. |

## Details

If `trueCluster` is given, the Rand index between the true cluster outcome and the clusters calculated by the methods is on diagonal.

## Value

A matrix of Rand indices showing the similarity of the methods.

## Author(s)

Christina Yassouridis

## Examples

```
set.seed(1234)
ds <- sampleFuncy(obsNr=80, timeNr=20, reg=TRUE, k=4, sd=.4)
res1 <- funcit(methods=1:4, data=Data(ds), k=4, clusters=Cluster(ds))

cls <- Cluster(res1)
rIMethods(methodNames=c("method1","method2","method3","method4"),cls=cls, trueCluster=Cluster(ds))
```

---

# sampleFuncy *Simulate functional data.*

---

## Description

Generates a functional dataset for 2-6 clusters.

## Usage

```
sampleFuncy(obsNr=100, k=4, timeNr=20, timeNrMax=NULL,
            timeNrMin=NULL, timeInterval=c(0, 1),
            nrGridPts=30, sd=0.3, reg=TRUE)
## S4 method for signature 'sampleFuncy'
Cluster(object)
## S4 method for signature 'sampleFuncy'
Data(object,...)
```

## Arguments

| | |
|---|---|
| obsNr | Number of curves. |
| k | Number of classes. |
| timeNr | Number of time points for regular datasets in Format2 (see funcit). |
| timeNrMax | Maximal number of time points for irregular datasets in Format1 (see funcit). |
| timeNrMin | Minimal number of time points for irregular dataset in Format1 (see funcit). |
| timeInterval | Time interval where time points are drawn from. |
| nrGridPts | Time interval is divided into nrGridPts grid points where time points are randomly drawn from. |
| sd | Standard deviation from the center curves. |
| reg | Regular dataset in Format2 or irregular dataset in Format1 (see funcit). |
| object | An object of class sampleFuncy |
| ... | Not used. |

## Details

Curves are generated by adding a normally distributed error term with mean 0 and standard deviation sd to the center functions. The center functions are sampled from

- $x^2$
- $\sqrt{x}$
- $sin(2 * \pi * x)$
- $x^3$
- $-x^2$
- $x - 1$

If `reg=TRUE` all curves are evaluated on the same time points. If `reg=FALSE` evaluation place and number can differ for each curve.

## Value

sampleFuncy A list with entries `data` and `clusters`.

Cluster Retrieve vector of cluster assignments

Data Retrieve matrix of dataset

## Author(s)

Christina Yassouridis

## See Also

`funcit`

## Examples

```
##sample a regular dataset
set.seed(2705)
ds <- sampleFuncy(obsNr=100, k=4, timeNr=20, reg=TRUE)
plotFuncy(ds)

##sample an irregular dataset
set.seed(2705)
ds <- sampleFuncy(obsNr=100, k=4, timeNrMin=3, timeNrMax=10, reg=FALSE)
plotFuncy(ds, lty=1)
```

# Index