Package 'dad'

July 22, 2025

Type Package

Title Three-Way / Multigroup Data Analysis Through Densities

Version 4.1.6

Author Rachid Boumaza [aut],
Pierre Santagostini [aut, cre],
Smail Yousfi [aut],
Gilles Hunault [ctb],
Julie Bourbeillon [ctb],
Besnik Pumo [ctb],
Sabine Demotes-Mainard [aut]

Maintainer Pierre Santagostini <pierre.santagostini@agrocampus-ouest.fr>

Description The data consist of a set of variables measured on several groups of individuals. To each group is associated an estimated probability density function. The package provides tools to create or manage such data and functional methods (principal component analysis, multidimensional scaling, cluster analysis, discriminant analysis...) for such probability densities.

URL https://forge.inrae.fr/dad/dad

BugReports https://forge.inrae.fr/dad/dad/-/issues

Depends R (>= 3.6.0)

Imports methods, stats, graphics, grDevices, utils, ggplot2, e1071,
DescTools

Suggests MASS, knitr, markdown, rmarkdown

Encoding UTF-8

License GPL (>= 2)

VignetteBuilder knitr

RoxygenNote 7.3.2

NeedsCompilation no

Repository CRAN

Date/Publication 2025-06-10 14:30:02 UTC

Contents

---

dad–package *Three-Way Data Analysis Through Densities*

Description

The three-way data consists of a set of variables measured on several groups of individuals. To each group is associated an estimated probability density function. The package provides functional methods (principal component analysis, multidimensional scaling, cluster analysis, discriminant analysis...) for such probability densities.

Details

Package:	dad
Type:	Package
Version:	4.1.2
Date:	2023-08-28
License:	GPL-2

URL: https://forgemia.inra.fr/dad/dad BugReports: https://forgemia.inra.fr/dad/dad/issues

To cite dad, use `citation("dad")`.

The main functions applying to the probability densities are:

- `fpcad`: functional principal component analysis,

- `fpcat`: functional principal component analysis applied to data indexed according to time,

- `fmdsd`: multidimensional scaling,

- `fhclustd`: hierarchical clustering,

- `fdiscd.misclass`: functional discriminant analysis in order to compute the misclassification ratio with the one-leave-out method,

- `fdiscd.predict`: discriminant analysis in order to predict the class (synonymous with cluster, not to be confused with the class attribute of an R object) of each probability density whose class is unknown,

- `mdsdd`: multidimensional scaling of discrete probability distributions,
- `discdd.misclass`: functional discriminant analysis of discrete probability distributions, in order to compute the misclassification ratio with the one-leave-out method,
- `discdd.predict`: discriminant analysis of discrete probability distributions, in order to predict the class of each probability distribution whose class is unknown,

The above functions are completed by:

- A `print()` method for objects of class fpcad, fmdsd, fdiscd.misclass, fdiscd.predict or mdsdd, in order to display the results of the corresponding function,
- A `plot()` method for objects of class fpcad, fmdsd, fhclustd or mdsdd, in order to display some useful graphics attached to the corresponding function,
- A generic function `interpret` that applies to objects of class fpcad fmdsd or mdsdd, helps the user to interpret the scores returned by the corresponding function, in terms of moments (`fpcad` or `fmdsd`) or in terms of marginal probability distributions (`mdsdd`).

We also introduce classes of objects and tools in order to handle collections of data frames:

- `folder` creates an object of class folder, that is a list of data frames which have in common the same columns.

  The following functions apply to a folder and compute some statistics on the columns of its elements: `mean.folder`, `var.folder`, `cor.folder`, `skewness.folder` or `kurtosis.folder`.
- `folderh` creates an object of class folderh, that is a list of data frames with a hierarchic relation between each pair of consecutive data frames.
- `foldert` creates an object of class foldert, that is a list of data frames indexed according to time, concerning the same individuals and variables or not.
- `read.mtg` creates an object of class foldermtg from an MTG (Multiscale Tree Graph) file containing plant architecture data.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard with the contributions from Gilles Hunault, Julie Bourbeillon and Besnik Pumo

References

Boumaza, R. (1998). Analyse en composantes principales de distributions gaussiennes multidimensionnelles. Revue de Statistique Appliqu?e, XLVI (2), 5-20.

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Boumaza, R. (2004). Discriminant analysis with independently repeated multivariate measurements: an L^2 approach. Computational Statistics & Data Analysis, 47, 823-843.

Delicado, P. (2011). Dimensionality reduction when data are density functions. Computational Statistics & Data Analysis, 55, 401-420.

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens. Centre de Recherches Archéologiques Médiévales de Saverne, 5, 5-38.

Rachev, S.T., Klebanov, L.B., Stoyanov, S.V. and Fabozzi, F.J. (2013). The methods of distances in the theory of probability and statistics. Springer.

Yousfi, S., Boumaza, R., Aissani, D., Adjabi, S. (2014). Optimal bandwith matrices in functional principal component analysis of density functions. Journal of Statistical Computation and Simulation, 85 (11), 2315-2330.

---

appendtofolderh	*Adds a data frame to a* folderh.

Description

Creates an object of class folderh by appending a data frame to an object of class folderh. The appended data frame will be the first or last element of the returned folderh.

Usage

```
appendtofolderh(fh, df, key, after = FALSE)
```

Arguments

fh	object of class [folderh](#).
df	data frame to be appended to fh.
key	character string. The key defining the relation $1toN$ between df and the first (if after = FALSE, the default value) or last (if after = TRUE) data frame of fh.
after	logical. If FALSE (default), the data frame df is related to the first data frame of fh, and is appended as the first element of the returned folderh. If TRUE, df is related to the last data frame of fh and becomes the last element of the returned folderh.

Value

Returns an object of class [folderh](#), that is a list of $n + 1$ data frames where n is the number of data frames of fh. The value of the attribute attr(, "keys") is c(key, attr(fh, "keys")) if after = FALSE), c(attr(fh, "keys"), key) otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folderh](#).

---

as.data.frame.folder   *Folder to data frame*

Description

Builds a data frame from an object of class folder.

Usage

```
## S3 method for class 'folder'
as.data.frame(x, row.names = NULL, optional = FALSE, ..., group.name = "group")
```

Arguments

x   object of class folder that is a list of data frames with the same column names.

row.names, optional

                         for consistency with as.data.frame. as.data.frame.folder does not take
                         them into account.

...   further arguments passed to or from other methods.

group.name   the name of the grouping variable. It is the name of the last column of the
                         returned data frame.

Details

The data frame is simply obtained by row binding the data frames of the folder and adding a factor
(as last column). The name of this column is given by group.name argument. The levels of this
factor are the names of the elements of the folder.

Value

as.data.frame.folder returns a data frame.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder: object of class folder. as.folder.data.frame: build an object of class folder from a
data frame.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
print(iris.fold)

iris.df <- as.data.frame(iris.fold)
print(iris.df)
```

---

as.data.frame.folderh   *Hierarchic folder to data frame*

Description

Builds a data frame from a folderh.

Usage

```
## S3 method for class 'folderh'
as.data.frame(x, row.names = NULL, optional = FALSE, ...,
        elt = names(x)[2], key = attr(x, "keys")[1])
```

Arguments

x	object of class [folderh](#) containing N (N>1) data frames: x[[1]],..., x[[N]], related by (N-1) keys: keys[1],..., keys[N-1].
row.names, optional	
	for consistency with [as.data.frame](#). Not taken into account.
...	further arguments passed to or from other methods.
elt	string. The name of one element of x, that is the data frame, say the j-th, whose rows are the rows of the returned data frame. See details.
key	string. The name of an element of attr(x, "keys"), that is the key, say the k-th with k<j, which is the factor designating the last column of the returned data frame. See details.

Value

as.data.frame.folderh returns a data frame whose row names are those of x[[elt]] (that is x[[j]]). The data frame contains the values of x[[elt]] and the corresponding values of the data frames x[[k]], these correspondances being defined by the keys of the hierarchic folder.

The column names of the returned data frame are organized in three parts.

1. The first part consists in the key names keys[k],..., keys[j-1].
2. The second part consists in the values of x[[j]].
3. The third part consists in the values of x[[k]] except the key keys[k].

See the examples to view these details.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder, folderh, as.folder.folderh.

Examples

```
# First example: rose flowers
data(roseflowers)
flg <- roseflowers$variety
flx <- roseflowers$flower

flfh <- folderh(flg, "rose", flx)
print(flfh)

fldf <- as.data.frame(flfh)
print(fldf)

# Second example: castles
data(castles.dated)
cag <- castles.dated$periods
cax <- castles.dated$stones

cafh <- folderh(cag, "castle", cax)
print(cafh)

cadf <- as.data.frame(cafh)
print(summary(cadf))

# Third example: leaves (example of a folderh with more than two data frames)
data(roseleaves)
lvr <- roseleaves$rose
lvs <- roseleaves$stem
lvl <- roseleaves$leaf
lvll <- roseleaves$leaflet

lfh <- folderh(lvr, "rose", lvs, "stem", lvl, "leaf", lvll)

lf1 <- as.data.frame(lfh, elt = "lvs", key = "rose")
print(lf1)

lf2 <- as.data.frame(lfh, elt = "lvl", key = "rose")
print(lf2)

lf3 <- as.data.frame(lfh, elt = "lvll", key = "rose")
print(lf3)

lf4 <- as.data.frame(lfh, elt = "lvll", key = "stem")
print(lf4)
```

---

as.data.frame.foldert *foldert to data frame*

Description

Builds a data frame from an object of class foldert.

Usage

```
## S3 method for class 'foldert'
as.data.frame(x, row.names = NULL, optional = FALSE, ..., group.name = "time")
```

Arguments

x   object of class foldert with the same row names. An object of class foldert is
                   a list of data frames with the same column names, each of them corresponding
                   to a time of observation.

row.names, optional

                   for consistency with as.data.frame. as.data.frame.foldert does not take
                   them into account.

...   further arguments passed to or from other methods.

group.name   the name of the grouping variable. It is the name of the last column of the
                   returned data frame.

                   As the observations are indexed by time, the default value is group.name =
                   "time".

Details

as.data.frame.foldert uses as.data.frame.folder.

Value

as.data.frame.foldert returns a data frame.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

foldert: object of class foldert. as.foldert.data.frame: build an object of class foldert
from a data frame. as.foldert.array: build an object of class foldert from a $3d$-array.

Examples

```
data(floribundity)
ftflor <- foldert(floribundity, cols.select = "union", rows.select = "union")
print(ftflor)
dfflor <- as.data.frame(ftflor)
summary(dfflor)
```

---

as.folder   *Coerce to a folder*

Description

Coerces a data frame or an object of class "folderh" to an object of class "folder".

Usage

```
as.folder(x, ...)
```

Arguments

x   an object of class data.frame or folderh.

                       • data.frame: see as.folder.data.frame
                       • folderh: see as.folder.folderh

...   further arguments passed to or from other methods.

Value

an object of class folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder: objects of class folder. as.data.frame.folder: build a data frame from an object
of class folder. as.folder.data.frame: build an object of class folder from a data frame.
as.folder.folderh: build an object of class folder from an object of class folderh.

as.folder.data.frame   *Data frame to folder*

Description

Builds an object of class folder from a data frame.

Usage

```
## S3 method for class 'data.frame'
as.folder(x, groups = tail(colnames(x), 1), ...)
```

Arguments

x	data frame.
groups	string. The name of the column of x containing the grouping variable. x[, groups] must be a factor, otherwise, there is an error.
	If omitted, the last column of x is used as grouping variable.
...	further arguments passed to or from other methods.

Value

as.folder.data.frame returns an object of class [folder](#) that is a list of data frames with the same column names.

Each element of the folder contains the data corresponding to one level of x[, groups].

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#): objects of class folder. [as.data.frame.folder](#): build a data frame from an object of class folder. [as.folder.folderh](#): build an object of class folder from an object of class folderh.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
print(iris.fold)

# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
print(roses.fold)
```

---

as.folder.folderh   *Hierarchic folder to folder*

Description

Creates an object of class folder, that is a list of data frames with the same column names, from a folderh.

Usage

```
## S3 method for class 'folderh'
as.folder(x, elt = names(x)[2], key = attr(x, "keys")[1], ...)
```

Arguments

x	object of class [folderh](#) containing N (N>1) data frames: x[[1]],..., x[[N]], related by (N-1) keys: keys[1],..., keys[N-1].
elt	string. The name of one element of x, that is data frame, say the j-th, whose rows are distributed among the data frames of the returned folder. See details.
key	string. The name of an element of attr(x, "keys"), that is the key, say the k-th with k<j, which is the factor whose levels are the names of the data frames of the returned folder. See details.
...	further arguments passed to or from other methods.

Value

as.folder.folderh returns an object of class [folder](#), a list of data frames with the same columns. These data frames contain the values of x[[elt]] (or x[[j]]) and the corresponding values of the data frames x[[j-1]], ... x[[k]], these correspondances being defined by the keys of the hierarchic folder. The names of these data frames are given by the levels of the key attr(x, "keys")[k].

The rows of the data frame x[[elt]] (or x[[j]]) are distributed among the data frames of the returned folder accordingly to the levels of the key attr(x, "keys")[k]. So the row names of the l-th data frame of the returned folder consist in the rows of x[[j]] corresponding to the l-th level of the key attr(x, "keys")[k].

The column names of the data frames of the returned folder are the union of the column names of the data frames x[[k]],..., x[[j]] and are organized in two parts.

1. The first part consists in the columns of x[[k]] except the column corresponding to the key attr(x, "keys")[k].

2. For each i=k+1,...,j the column names of the data frame x[[i]] are reorganized so that the key attr(x, "keys")[i] is its first column. The columns of the reorganized data frames x[[k+1]],..., x[[j]] are concatenated. The result forms the second part.

Notice that if:

- the folderh has two data frames df1 and df2, where the factor corresponding to the key has T levels, and one column of df2, say df2[, "Fa"], is a factor with levels "a1", ..., "ap"

- and the folder returned by as.folder includes T data frames dat1, ..., datT,

then each of dat1, ..., datT has a column named "Fa" which is a factor **with the same levels** "a1", ..., "ap" **as** df2[, "Fa"].

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder, folderh. as.folder.folderh to build an object of class folder from an object of class folderh. as.data.frame.folder to build a data frame from an object of class folder. as.data.frame.folderh to build a data frame from an object of class folderh.

Examples

```
# First example: flowers
data(roseflowers)
flg <- roseflowers$variety
flx <- roseflowers$flower

flfh <- folderh(flg, "rose", flx)
print(flfh)

flf <- as.folder(flfh)
print(flf)

# Second example: castles
data(castles.dated)
cag <- castles.dated$periods
cax <- castles.dated$stones

cafh <- folderh(cag, "castle", cax)
print(cafh)

caf <- as.folder(cafh)
print(caf)

# Third example: leaves (example of a folderh of more than two data frames)
data(roseleaves)
lvr <- roseleaves$rose
lvs <- roseleaves$stem
lvl <- roseleaves$leaf
lvll <- roseleaves$leaflet

lfh <- folderh(lvr, "rose", lvs, "stem", lvl, "leaf", lvll)

lf1 <- as.folder(lfh, elt = "lvs", key = "rose")
print(lf1)

lf2 <- as.folder(lfh, elt = "lvl", key = "rose")
```

```
print(lf2)

lf3 <- as.folder(lfh, elt = "lvll", key = "rose")
print(lf3)

lf4 <- as.folder(lfh, elt = "lvll", key = "stem")
print(lf4)
```

---

as.folderh   *Coerce to a folderh*

Description

Coerces an object to an object of class [folderh](#).

Usage

```
as.folderh(x, classes)
```

Arguments

x   an object to be coerced to an object of class folderh. In the current version, it
               is an object of class "foldermtg" (see [as.folderh.foldermtg](#)).

classes   argument useful for [as.folderh.foldermtg](#).

Value

an object of class [folderh](#).

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[as.folderh.foldermtg](#): build an object of class folderh from an object of class foldermtg.

---

as.folderh.foldermtg   *Build a hierarchic folder from an object of class foldermtg*

Description

Creates an object of class `folderh` from an object of class `foldermtg`.

Usage

```
## S3 method for class 'foldermtg'
as.folderh(x, classes)
```

Arguments

x	object of class `foldermtg`.
classes	character vector. Codes of the vertex classes in the returned folderh. These codes are the names of the elements (data frames) of x containing the features on the vertices corresponding to the codes.
	These codes must be distinct, and the corresponding classes must have distinct scales (see `foldermtg`). Otherwise, there is an error.
	These codes, except the one with the highest scale, are the keys of the returned folderh.

Details

This function uses `folderh`.

Value

An object of class `folderh`. Its elements are the data frames of x containing the features on vertices. Hence, each data frame matches with a class of vertex, and a scale. These data frames are in increasing order of the scale.

A column (factor) is added to the first data frame, containing the identifier of the vertex. Two columns are added to the second data frame:

1. the first one is a factor which gives, for each vertex, the name of the vertex of the first data frame which is its "parent",

2. and the second one is also a factor and contains the vertex's identifier.

And so on for the third and following data frames, if relevant.

The column containing the vertex identifiers is redundant with the row names; anyway, it is necessary for `folderh`.

The key of the relationship between the two first data frame is given by the first column of each of these data frames. If there are more than two data frames, the key of the relationship between the n-th and $(n+1)$-th data frames ($n > 1$) is given by the second column of the n\$th data frame and the first column of the $(n+1)$-th data frame.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

read.mtg: reads a MTG file and creates an object of class "foldermtg". folderh : object of class
folderh.

Examples

```
mtgfile <- system.file("extdata/plant1.mtg", package = "dad")
x <- read.mtg(mtgfile)

# folderh containing the plant ("P") and the stems ("A")
as.folderh(x, classes = c("P", "A"))

# folderh containing the plant ("P"), axes ("A") and phytomers ("M")
as.folderh(x, classes = c("P", "A", "M"))

# folderh containing the plant ("P") and the phytomers ("M")
as.folderh(x, classes = c("P", "M"))

# folderh containing the axes and phytomers
fhPM <- as.folderh(x, classes = c("A", "M"))
# coerce this folderh into a folder, and compute statistics on this folder
fPM <- as.folder(fhPM)
mean(fPM)
```

---

as.foldert   *Coerce to a foldert*

Description

Coerces a data frame or array to an object of class foldert.

Usage

```
as.foldert(x, ...)
```

Arguments

x   an object of class data.frame or array.

- data.frame: see as.foldert.data.frame
- array: see as.foldert.array

...   arguments passed to as.foldert.data.frame or as.foldert.array, further
arguments passed to or from other methods.

Value

an object of class [foldert](#).

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

---

as.foldert.array   *Data frame to foldert*

Description

Builds an object of class [foldert](#) from a $3d$-array.

Usage

```
## S3 method for class 'array'
as.foldert(x, ind = 1, var = 2, time = 3, ...)
```

Arguments

x	a $3d$-array.
ind, var, time	three distinct integers among 1, 2 and 3.
	ind gives the dimension of the observations, var gives the dimension of the variables and ind gives the dimension of the times.
...	further arguments passed to or from other methods.

Value

an object ft of class [foldert](#) that is a list of data frames, each of them corresponding to a time of observation; these data frames have the same column names.

They necessarily have the same row names (attr(ft, "same.rows")=TRUE). The "times" attribute of ft: attr(ft, "times") is a numeric vector, an ordered factor or an object of class Date, and contains the values nf the dimension of x given by time argument.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](#): objects of class foldert.

[as.foldert.data.frame](#): build an object of class foldert from a data frame.

Examples

```
x <- array(c(rep(0, 5), rep(0, 5), rep(0, 5),
              rnorm(5, 2, 1), rnorm(5, 3, 2), rnorm(5, -2, 0.5),
              rnorm(5, 4, 1), rnorm(5, 5, 3), rnorm(5, -3, 1)),
            dim = c(5, 3, 3),
            dimnames = list(1:5, c("z1", "z2", "z3"), c("t1", "t2", "t3")))
# The individuals which were observed are on the 1st dimension,
# the variables are on the 2nd dimension and the times are on the 3rd dimension.
ft <- as.foldert(x, ind = 1, var = 2, time = 3)
```

---

as.foldert.data.frame   *Data frame to foldert*

Description

Builds an object of class [foldert](#) from a data frame.

Usage

```
## S3 method for class 'data.frame'
as.foldert(x, method = 1, ind = 1, timecol = 2, nvar = NULL, same.rows = TRUE, ...)
```

Arguments

x	data frame.
method	1 or 2. Indicates the layout of the data frame x and, therefore, the method used to extract the data and build the foldert.

- If method = 1, there is a column containing the identifiers of the measured objects and a column containing the times. The other columns contain the observations.
- If method = 2, there is a column containing the identifiers of the measured objects, and the observations are organized as follows:
  - the observations corresponding to the 1st time are on columns timecol : (timecol + nvar - 1)
  - the observations corresponding to the 2nd time are on columns (timecol + nvar) : (timecol + 2 * nvar - 1)
  - and so on.

ind	string or numeric. The name of the column of x containing the indentifiers of the measured objects, or the number of this column.
timecol	string or numeric.

- If method = 1, timecol is the name or the number of the column of x containing the times of observation, or the number of this column. x[, timecol] must be of class "numeric", "ordered", "Date", "POSIXlt" or "POSIXct", otherwise, there is an error.

- If `method=2`, `timecol` is the name or the number of the first column corresponding to the first observation. If there are duplicated column names and several columns are named by `timecol`, the first one is considered.

nvar   integer. If `method=2`, indicates the number of variables observed at each time.
                Omitted if `method=1`.

same.rows   logical. If `TRUE` (default), the elements of the returned foldert are data frames with the same row names.
                Necessarily `TRUE` if `method = 2`.

...   further arguments passed to or from other methods.

Value

an object `ft` of class [foldert](), that is a list of data frames organised according to time; these data frames have the same column names.

If `method = 1`, they can have the same row names (`attr(ft, "same.rows") = TRUE`) or not (`attr(ft, "same.rows") = FALSE`). The time attribute `attr(ft, "times")` has the same class as `x[, timecol]` (numeric vector, ordered factor or object of class `"Date"`, `"POSIXlt"` or `"POSIXct"`) and contains the values of `x[, timecol]`.

If `method = 2`, they necessarily have the same row names: `attr(ft, "same.rows") = TRUE` and `attr(ft, "times")` is `1:length(ft)`.

The rownames of each data frame are the identifiers of the individuals, as given by `x[, ind]`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](): objects of class foldert.

[as.data.frame.foldert](): build a data frame from an object of class foldert.

[as.foldert.array](): build an object of class foldert from a $3d$-array.

Examples

```
# First example: method = 1

times <- as.Date(c("2017-03-01", "2017-04-01", "2017-05-01"))
x1 <- data.frame(t=times[1], ind=1:6,
                 f=c("a","a","a","b","b","b"), z1=rep(0,6), z2=rep(0,6),
                 stringsAsFactors = TRUE)
x2 <- data.frame(t=times[2], ind=c(1,4,6),
                 f=c("a","b","b"), z1=rnorm(3,1,1), z2=rnorm(3,3,2),
                 stringsAsFactors = TRUE)
x3 <- data.frame(t=times[3], ind=c(1,3:6),
                 f=c("a","a","a","b","b"), z1=rnorm(5,3,2), z2=rnorm(5,6,3),
                 stringsAsFactors = TRUE)
x <- rbind(x1, x2, x3)
```

```
ft1 <- as.foldert(x, method = 1, ind = "ind", timecol = "t", same.rows = TRUE)
print(ft1)

ft2 <- as.foldert(x, method = 1, ind = "ind", timecol = "t", same.rows = FALSE)
print(ft2)

data(castles.dated)
periods <- castles.dated$periods
stones <- castles.dated$stones
stones$stone <- rownames(stones)

castledf <- merge(periods, stones, by = "castle")
castledf$period <- as.numeric(castledf$period)
castledf$stone <- as.factor(paste(as.character(castledf$castle),
                            as.character(castledf$stone), sep = "_"))

castfoldt1 <- as.foldert(castledf, method = 1, ind = "stone", timecol = "period",
                         same.rows = FALSE)
summary(castfoldt1)


# Second example: method = 2

times <- as.Date(c("2017-03-01", "2017-04-01", "2017-05-01"))
y1 <- data.frame(z1=rep(0,6), z2=rep(0,6))
y2 <- data.frame(z1=rnorm(6,1,1), z2=rnorm(6,3,2))
y3 <- data.frame(z1=rnorm(6,3,2), z2=rnorm(6,6,3))
y <- cbind(ind = 1:6, y1, y2, y3)

ft3 <- as.foldert(y, method = 2, ind = "ind", timecol = 2, nvar = 2)
print(ft3)
```

---

association measures   *Association measures between several categorical variables of a data frame*

Description

Computes pairwise association measures (Cramer's V, Pearson's contingency coefficient, phi, Tschuprow's T) between the categorical variables of a data frame, using functions of the package DescTools (see [Assocs](#)).

Usage

```
cramer.data.frame(x, check = TRUE)
pearson.data.frame(x, check = TRUE)
phi.data.frame(x, check = TRUE)
tschuprow.data.frame(x, check = TRUE)
```

Arguments

x	a data frame (can also be a tibble). Its columns should be factors.
check	logical. If TRUE (default) the function checks if each column of x is a factor, and there is a warning if it is not.

Value

A square matrix whose elements are the pairwise association measures.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Examples

```
data(roses)
xr = roses[,c("Sha", "Den", "Sym", "rose")]
xr$Sha = cut(xr$Sha, breaks = c(0, 5, 7, 10))
xr$Den = cut(xr$Den, breaks = c(0, 4, 6, 10))
xr$Sym = cut(xr$Sym, breaks = c(0, 6, 8, 10))
cramer.data.frame(xr)
pearson.data.frame(xr)
phi.data.frame(xr)
tschuprow.data.frame(xr)
```

---

association measures for folder

Association measures between categorical variables of the data frames of a folder

---

Description

Computes the pairwise association measures (Cramer's V, Pearson's contingency coefficient, phi, Tschuprow's T) between the categorical variables of an object of class folder. The computation is carried out using the functions cramer.data.frame, tschuprow.data.frame, pearson.data.frame or phi.data.frame. These functions are built from corresponding functions of the package DescTools (see Assocs)

Usage

```
cramer.folder(xf)
tschuprow.folder(xf)
pearson.folder(xf)
phi.folder(xf)
```

Arguments

xf	an object of class folder that is a list of data frames with the same column names. Its columns should be factors, otherwise there is a warning.

Value

A list the length of which is equal to the number of data frames of the folder. Each element of the list is a square matrice giving the pairwise association measures of the variables of the corresponding data frame.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Examples

```
data(roses)
xr = roses[,c("Sha", "Den", "Sym", "rose")]
xr$Sha = cut(xr$Sha, breaks = c(0, 5, 7, 10))
xr$Den = cut(xr$Den, breaks = c(0, 4, 6, 10))
xr$Sym = cut(xr$Sym, breaks = c(0, 6, 8, 10))
xfolder = as.folder(xr, groups = "rose")
cramer.folder(xfolder)
pearson.folder(xfolder)
phi.folder(xfolder)
tschuprow.folder(xfolder)
```

---

bandwidth.parameter   *Parameter of the normal reference rule*

Description

Computation of the parameter of the normal reference rule in order to estimate the (matrix) bandwidth.

Usage

```
bandwidth.parameter(p, n)
```

Arguments

p   sample dimension.

n   sample size.

Details

The parameter is equal to:

$$h = (\frac{4}{n(p+2)})^{\frac{1}{p+4}}$$

It is based on the minimisation of the asymptotic mean integrated square error in density estimation when using the Gaussian kernel method (Wand and Jones, 1995).

Value

Returns the value required by the functions [fpcad](), [fmdsd](), [fdiscd.misclass]() and [fdiscd.predict]() when their argument windowh is set to NULL.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Wand, M. P., Jones, M. C. (1995). Kernel Smoothing. Boca Raton, FL: Chapman and Hall.

Examples

```
# Sample size :
n <- 20
# Number of variables :
p <- 3
bandwidth.parameter(p, n)
```

---

castles   *Alsacian castles by year of building*

Description

The data were collected by J.M. Rudrauf on Alsacian castles whose building year is known (even approximatively). On each castle, he measured 4 structural parameters on a sample of building stones.

These data are about the same castles as in [castles.dated]() data set.

Usage

```
data(castles)
```

Format

castles is a list of 46 data frames. Each of these data frames matches with one year (between 1136 and 1510) and contains measures on one or several castles which have been built since that year.

Each data frame has 5 to 101 rows (stones) and 5 columns: height, width, edging, boss (numeric) and castle (factor).

Source

Rudrauf, J.M., Boumaza, R. (2001). Contribution a l'etude de l'architecture medievale: les caracteristiques des pierres a bossage des chateaux forts alsaciens. Centre de Recherches Archeologiques Medievales de Saverne, 5, 5-38.

Examples

```
data(castles)
foldert(castles)
```

---

castles.dated   *Dated Alsacian castles*

Description

The data were collected by J.M. Rudrauf on Alsacian castles whose building period is known (even approximately). On each castle, he measured 4 structural parameters on a sample of building stones.

Usage

```
data(castles.dated)
```

Format

`castles.dated` is a list of two data frames:

castles.dated$stones: this first data frame has 1262 cases (rows) and 5 variables (columns) that are named `height`, `width`, `edging`, `boss` (numeric) and `castle` (factor).

castles.dated$periods: this second data frame has 68 cases and 2 variables named `castle` and `period`; the column `castle` corresponds to the levels of the factor `castle` of the first data frame; the column `period` is a factor with 6 levels indicating the approximative building period. Thus this factor defines 6 classes of castles.

Source

Rudrauf, J.M., Boumaza, R. (2001). Contribution a l'etude de l'architecture medievale: les caracteristiques des pierres a bossage des chateaux forts alsaciens. Centre de Recherches Archeologiques Medievales de Saverne, 5, 5-38.

Examples

```
data(castles.dated)
summary(castles.dated$stones)
summary(castles.dated$periods)
```

---

castles.nondated *Non dated Alsacian castles*

Description

The data were collected by J.M. Rudrauf on Alsacian castles whose building period is unknown. On each castle, he measured 4 structural parameters on a sample of building stones.

Usage

```
data(castles.nondated)
```

Format

castles.nondated is a list of two data frames:

castles.nondated$stones: this first data frame has 1280 cases (rows) and 5 variables (columns) that are named height, width, edging, boss (numeric) and castle (factor).

castles.nondated$periods: this second data frame has 67 cases and 2 variables named castle and period; the column castle corresponds to the levels of the factor castle of the first data frame; the column period is a factor indicating NA as the building period is unknown.

Notice that the data frames corresponding to the castles whose building period is known are those in castles.dated.

Source

Rudrauf, J.M., Boumaza, R. (2001). Contribution a l'etude de l'architecture medievale: les caracteristiques des pierres a bossage des chateaux forts alsaciens. Centre de Recherches Archeologiques Medievales de Saverne, 5, 5-38.

Examples

```
data(castles.nondated)
summary(castles.nondated$stones)
summary(castles.nondated$periods)
```

---

cor.folder *Correlation matrices of a folder of data sets*

Description

Computes the correlation matrices of the elements of an object of class folder.

Usage

```
cor.folder(x, use = "everything", method = "pearson")
```

Arguments

x   an object of class [folder](#) that is a list of data frames with the same column names.

use   an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs" (see [var](#)).

method   a character string indicating which correlation coefficient (or covariance) is to be computed. One of "pearson" (default), "kendall", or "spearman": can be abbreviated.

Details

It uses [cor](#) to compute the variance matrix of the numeric columns of each element of the folder. If some columns of the data frames are not numeric, there is a warning, and the variances are computed on the numeric columns only.

Value

A list whose elements are the correlation matrices of the elements of the folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#) to create an object is of class folder. [mean.folder](#), [var.folder](#), [skewness.folder](#), [kurtosis.folder](#) for other statistics for folder objects.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
iris.cor <- cor.folder(iris.fold)
print(iris.cor)

# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
roses.cor <- cor.folder(roses.fold)
print(roses.cor)
```

---

cut.data.frame   *Change numeric variables into factors*

Description

This function changes numerical columns of a data frame x into factors. For each of these columns,
its range is divided into intervals and the values of this column is recoded according to which
interval they fall.

For that, [cut](#) is applied to each column of x.

Usage

```
## S3 method for class 'data.frame'
cut(x, breaks, labels = NULL, include.lowest = FALSE, right = TRUE, dig.lab = 3L,
    ordered_result = FALSE, cutcol = NULL, ...)
```

Arguments

x   data frame (can also be a tibble).

breaks   list or numeric.

- If breaks is a list, its length is equal to the number of columns in the data
  frame. It can be:
  - a list of numeric vectors. The jth element corresponds to the column
    x[, j], and is a vector of two or more unique cut points
  - or a list of single numbers (each greater or equal to 2). breaks[[j]]
    element gives the number of intervals into which th j^{th} variable of the
    folder is to be cut. The elements breaks[[j]] corresponding to non-
    numeric columns must be NULL; if not, there is a warning.
- If breaks is a numeric vector, it gives the number of intervals into which
  every column x[, j] is to be cut (see [cut](#)).

labels	list of character vectors. If given, its length is equal to the number of columns of x. labels[[j]] gives the labels for the intervals of the jth columns of the data frame. By default, the labels are constructed using "(a,b]" interval notation. If labels = FALSE, simple integer codes are returned instead of a factor.
	See [cut]().
include.lowest	logical, indicating if, for each column x[, j], an x[i, j] equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included (see [cut]()).
right	logical, indicating if the intervals should be closed on the right (and open on the left) or vice versa (see [cut]()).
dig.lab	integer or integer vector, which is used when labels are not given. It determines the number of digits used in formatting the break numbers.
	• If it is a single value, it gives the number of digits for all variables of the folder (see [cut]()).
	• If it is a list of integers, its length is equal to the number of variables, and the jth element gives the number of digits for the jth variable of the folder.
ordered_result	logical: should the results be ordered factors? (see [cut]())
cutcol	numeric vector: indices of the columns to be converted into factors. These columns must all be numeric. Otherwise, there is a warning.
...	further arguments passed to or from other methods.

Value

A data frame with the same column and row names as x.

If cutcol is given, each numeric column x[, j] whose number is contained in cutcol is replaced by a factor. The other columns are unmodified.

If any column x[, j] whose number is in cutcol is not numeric, it is unmodified.

If cutcol is omitted, every numerical columns are replaced by factors.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Examples

```
data("roses")
x <- roses[roses$rose %in% c("A", "B"), c("Sha", "Sym", "Den", "rose")]

cut(x, breaks = 3)
cut(x, breaks = 5)
cut(x, breaks = c(0, 4, 6, 10))
cut(x, breaks = list(c(0, 6, 8, 10), c(0, 5, 7, 10), c(0, 6, 7, 10)))
cut(x, breaks = list(c(0, 6, 8, 10), c(0, 5, 7, 10)), cutcol = 1:2)
```

cut.folder   *In a folder: change numeric variables into factors*

Description

This function applies to a [folder](). For each elements (data frames) of this folder, it changes its numerical columns into factors, using `cut.data.frame`.

Usage

```
## S3 method for class 'folder'
cut(x, breaks, labels = NULL, include.lowest = FALSE, right = TRUE, dig.lab = 3L,
    ordered_result = FALSE, cutcol = NULL, ...)
```

Arguments

x	an object of class [folder]().
breaks	list or numeric, defining the intervals into which the variables of each element of the folder is to be cut. See `cut.folder`.
labels	list of character vectors. If not omitted, it gives the labels for the intervals of each column of the elements of x. See `cut.folder`.
include.lowest	logical, indicating if a value equal to the lowest (or highest, for right = FALSE) 'breaks' value should be included (see `cut.folder`).
right	logical, indicating if the intervals should be closed on the right (and open on the left) or vice versa (see `cut.folder`).
dig.lab	integer or integer vector, which is used when labels are not given. It determines the number of digits used in formatting the break numbers. See `cut.folder`.
ordered_result	logical: should the results be ordered factors? (see `cut.folder`)
cutcol	numeric vector: indices of the columns of the elements of x to be converted into factors. These columns must all be numeric. Otherwise, there is a warning. See `cut.folder`.
...	further arguments passed to or from other methods.

Value

An object of class folder with the same length and names as x. Its elements (data frames) have the same column and row names as the elements of x.

For more details, see `cut.data.frame`

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Examples

```
data("roses")

x <- as.folder(roses[, c("Sha", "Den", "Sym", "rose")], groups = "rose")
summary(x)

x3 <- cut(x, breaks = 3)
summary(x3)

x7 <- cut(x, breaks = 7)
summary(x7)
```

---

ddchisqsym   *Distance between probability distributions of discrete variables given samples*

Description

Symmetrized chi-squared distance between two multivariate ($q > 1$) or univariate ($q = 1$) discrete probability distributions, estimated from samples.

Usage

```
ddchisqsym(x1, x2)
```

Arguments

x1, x2   vectors or data frames of q columns (can also be a tibble).

                      If they are data frames and have not the same column names, there is a warning.

Details

Let p_1 and p_2 denote the estimated probability distributions of the discrete samples x_1 and x_2. The symmetrized chi-squared distance between the discrete probability distributions of the samples are computed using the [ddchisqsympar](#) function.

Value

The distance between the two probability distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddchisqsympar: chi-squared distance between two discrete distributions, given the probabilities on their common support.

Other distances: ddhellinger, ddjeffreys, ddjensen, ddlp.

Examples

```
# Example 1
x1 <- c("A", "A", "B", "B")
x2 <- c("A", "A", "A", "B", "B")
ddchisqsym(x1, x2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
ddchisqsym(x1, x2)
```

---

ddchisqsympar	*Distance between discrete probability distributions given the probabilities on their common support*

Description

Symmetrized chi-squared distance between two discrete probability distributions on the same support (which can be a Cartesian product of q sets) , given the probabilities of the states (which are q-tuples) of the support.

Usage

```
ddchisqsympar(p1, p2)
```

Arguments

p1	array (or table) the dimension of which is q. The first probability distribution on the support.
p2	array (or table) the dimension of which is q. The second probability distribution on the support.

Details

The chi-squared distance between two discrete distributions p_1 and p_2 is given by:

$$\sum_x (p_1(x) - p_2(x))^2 / p_2(x)$$

Then the symmetrized chi-squared distance is given by the formula:

$$||p_1 - p_2|| = \sum_x (p_1(x) - p_2(x))^2 / (p_1(x) + p_2(x))$$

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddchisqsym: chi-squared distance between two estimated discrete distributions, given samples.

Other distances: ddhellingerpar, ddjeffreyspar, ddjensenpar, ddlppar.

Examples

```
# Example 1
p1 <- array(c(1/2, 1/2), dimnames = list(c("a", "b")))
p2 <- array(c(1/4, 3/4), dimnames = list(c("a", "b")))
ddchisqsympar(p1, p2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
p1 <- table(x1)/nrow(x1)
p2 <- table(x2)/nrow(x2)
ddchisqsympar(p1, p2)
```

---

ddhellinger	*Distance between probability distributions of discrete variables given samples*

Description

Hellinger (or Matusita) distance between two multivariate ($q > 1$) or univariate ($q = 1$) discrete probability distributions, estimated from samples.

Usage

```
ddhellinger(x1, x2)
```

Arguments

x1, x2   data frames of q columns or vectors (can also be tibbles).

                If they are data frames and have not the same column names, there is a warning.

Details

Let p_1 and p_2 denote the estimated probability distributions of the discrete samples x_1 and x_2. The Matusita distance between the discrete probability distributions of the samples are computed using the [ddhellingerpar](#) function.

Value

The distance between the two probability distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddhellingerpar](#): Hellinger metric (Matusita distance) between two discrete distributions, given the on their common support probabilities.

Other distances: [ddchisqsym](#), [ddjeffreys](#), [ddjensen](#), [ddlp](#).

Examples

```
# Example 1
x1 <- c("A", "A", "B", "B")
x2 <- c("A", "A", "A", "B", "B")
ddhellinger(x1, x2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
ddhellinger(x1, x2)
```

---

ddhellingerpar	*Distance between discrete probability distributions given the probabilities on their common support*

Description

Hellinger (or Matusita) distance between two discrete probability distributions on the same support (which can be a Cartesian product of q sets) , given the probabilities of the states (which are q-tuples) of the support.

Usage

```
ddhellingerpar(p1, p2)
```

Arguments

p1   array (or table) the dimension of which is q. The first probability distribution on the support.

p2   array (or table) the dimension of which is q. The second probability distribution on the support.

Details

The Hellinger distance between two discrete distributions p_1 and p_2 is given by: $\sqrt{\sum_x \left(\sqrt{p_1(x)} - \sqrt{p_2(x)} \right)^2}$

Notice that some authors divide this expression by $\sqrt{2}$.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddhellinger: Hellinger distance between two estimated discrete distributions, given samples.

Other distances: ddchisqsympar, ddjeffreyspar, ddjensenpar, ddlppar.

Examples

```
# Example 1
p1 <- array(c(1/2, 1/2), dimnames = list(c("a", "b")))
p2 <- array(c(1/4, 3/4), dimnames = list(c("a", "b")))
ddhellingerpar(p1, p2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
p1 <- table(x1)/nrow(x1)
p2 <- table(x2)/nrow(x2)
ddhellingerpar(p1, p2)
```

---

ddjeffreys	*Divergence between probability distributions of discrete variables given samples*

Description

jeffreys's divergence (symmetrized Kullback-Leibler divergence) between two multivariate ($q > 1$) or univariate ($q = 1$) discrete probability distributions, estimated from samples.

Usage

```
ddjeffreys(x1, x2)
```

Arguments

x1, x2   vectors or data frames of q columns (can also be a tibble).

If they are data frames and have not the same column names, there is a warning.

Details

Let p_1 and p_2 denote the estimated probability distributions of the discrete samples x_1 and x_2. The jeffreys's divergence between the discrete probability distributions of the samples are computed using the [ddjeffreyspar](#) function.

Value

The divergence between the two probability distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddjeffreyspar](#): Jeffrey's distances between two discrete distributions, given the probabilities on their common support.

Other distances: [ddchisqsym](#), [ddhellinger](#), [ddjensen](#), [ddlp](#).

Examples

```
# Example 1
x1 <- c("A", "A", "B", "B")
x2 <- c("A", "A", "A", "B", "B")
ddjeffreys(x1, x2)

# Example 2 (Its value can be infinity -Inf-)
x1 <- c("A", "A", "B", "C")
x2 <- c("A", "A", "A", "B", "B")
ddjeffreys(x1, x2)

# Example 3
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
ddjeffreys(x1, x2)
```

---

ddjeffreyspar | *Distance between discrete probability distributions given the probabilities on their common support*

Description

Jeffreys divergence (symmetrized Kullback-Leibler divergence) between two discrete probability distributions on the same support (which can be a Cartesian product of q sets) , given the probabilities of the states (which are q-tuples) of the support.

Usage

```
ddjeffreyspar(p1, p2)
```

Arguments

p1   array (or table) the dimension of which is q. The first probability distribution on the support.

p2   array (or table) the dimension of which is q. The second probability distribution on the support.

Details

Jeffreys divergence $||p_1 - p_2||$ between two discrete distributions p_1 and p_2 is given by the formula:

$$||p_1 - p_2|| = \sum_x (p_1(x) - p_2(x)) log(p_1(x)/p_2(x))$$

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddjeffreys](): Jeffreys distance between two estimated discrete distributions, given samples.

Other distances: [ddchisqsympar](), [ddhellingerpar](), [ddjensenpar](), [ddlppar]().

Examples

```
# Example 1
p1 <- array(c(1/2, 1/2), dimnames = list(c("a", "b")))
p2 <- array(c(1/4, 3/4), dimnames = list(c("a", "b")))
ddjeffreyspar(p1, p2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
p1 <- table(x1)/nrow(x1)
p2 <- table(x2)/nrow(x2)
ddjeffreyspar(p1, p2)
```

---

ddjensen	*Divergence between probability distributions of discrete variables given samples*

Description

Jensen-Shannon divergence between two multivariate ($q > 1$) or univariate ($q = 1$) discrete probability distributions, estimated from samples.

Usage

```
ddjensen(x1, x2)
```

Arguments

x1, x2   vectors or data frames of q columns (can also be tibbles).

If they are data frames and have not the same column names, there is a warning.

Details

Let p_1 and p_2 denote the estimated probability distributions of the discrete samples x_1 and x_2. The Jensen-Shannon divergence between the discrete probability distributions of the samples are computed using the [ddjensenpar]() function.

Value

The distance between the two probability distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

`ddjensenpar`: Jensen-Shannon distance between two discrete distributions, given the probabilities on their common support.

Other distances: `ddchisqsym`, `ddhellinger`, `ddjeffreys`, `ddlp`.

Examples

```
# Example 1
x1 <- c("A", "A", "B", "B")
x2 <- c("A", "A", "A", "B", "B")
ddjensen(x1, x2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
ddjensen(x1, x2)
```

---

ddjensenpar   *Divergence between discrete probability distributions given the prob-
                        abilities on their common support*

Description

Jensen-Shannon divergence between two discrete probability distributions on the same support (which can be a Cartesian product of q sets), given the probabilities of the states (which are q-tuples) of the support.

Usage

```
ddjensenpar(p1, p2)
```

Arguments

p1	array (or table) the dimension of which is q. The first probability distribution on the support.
p2	array (or table) the dimension of which is q. The second probability distribution on the support.

Details

The Jensen-Shannon divergence $||p_1 - p_2||$ between two discrete distributions p_1 and p_2 is given by the formula:

$$||p_1 - p_2|| = \sum_x \left(p_1(x) log(2p_1(x)/(p_1(x) + p_2(x)))) \right) + (p_2(x) log(2p_2(x)/(p_1(x) + p_2(x))))$$

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddjensen](): Jensen-Shannon distance between two estimated discrete distributions, given samples.

Other distances: [ddchisqsympar](), [ddhellingerpar](), [ddjeffreyspar](), [ddlppar]().

Examples

```
# Example 1
p1 <- array(c(1/2, 1/2), dimnames = list(c("a", "b")))
p2 <- array(c(1/4, 3/4), dimnames = list(c("a", "b")))
ddjensenpar(p1, p2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
p1 <- table(x1)/nrow(x1)
p2 <- table(x2)/nrow(x2)
ddjensenpar(p1, p2)
```

---

ddlp   *Distance between probability distributions of discrete variables given samples*

Description

L^p distance between two multivariate ($q > 1$) or univariate ($q = 1$) discrete probability distributions, estimated from samples.

Usage

```
ddlp(x1, x2, p = 1)
```

Arguments

x1, x2	vectors or data frames of q columns (can also be tibbles).
	If they are data frames and have not the same column names, there is a warning.
p	integer. Parameter of the distance.

Details

Let p_1 and p_2 denote the estimated probability distributions of the discrete samples x_1 and x_2. The L^p distance between the discrete probability distributions of the samples are computed using the [ddlppar](#) function.

Value

The distance between the two discrete probability distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddlppar](#): L^p distance between two discrete distributions, given the probabilities on their common support.

Other distances: [ddchisqsym](#), [ddhellinger](#), [ddjeffreys](#), [ddjensen](#).

Examples

```
# Example 1
x1 <- c("A", "A", "B", "B")
x2 <- c("A", "A", "A", "B", "B")
ddlp(x1, x2)
ddlp(x1, x2, p = 2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
ddlp(x1, x2)
```

---

ddlppar	*Distance between discrete probability distributions given the probabilities on their common support*

Description

L^p distance between two discrete probability distributions on the same support (which can be a Cartesian product of q sets) , given the probabilities of the states (which are q-tuples) of the support.

Usage

```
ddlppar(p1, p2, p = 1)
```

Arguments

p1   array (or table) the dimension of which is q. The first probability distribution on the support.

p2   array (or table) the dimension of which is q. The second probability distribution on the support.

p   integer. Parameter of the distance.

Details

The L^p distance $||p_1 - p_2||$ between two discrete distributions p_1 and p_2 is given by the formula:

$$||p_1 - p_2||^p = \sum_x |p_1(x) - p_2(x)|^p$$

If $p = 1$, it is the variational distance.

If $p = 2$, it is the Patrick-Fisher distance.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddlp](#): L^p distance between two estimated discrete distributions, given samples.

Other distances: [ddchisqsympar](#), [ddhellingerpar](#), [ddjeffreyspar](#), [ddjensenpar](#).

Examples

```
# Example 1
p1 <- array(c(1/2, 1/2), dimnames = list(c("a", "b")))
p2 <- array(c(1/4, 3/4), dimnames = list(c("a", "b")))
ddlppar(p1, p2)
ddlppar(p1, p2, p=2)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
p1 <- table(x1)/nrow(x1)
p2 <- table(x2)/nrow(x2)
ddlppar(p1, p2)
```

---

departments   *French departments and regions*

Description

Departments and regions of metropolitan France.

Usage

```
data(departments)
```

Format

departments is a data frame with 96 rows and 4 columns (factors):

coded: departments: numbers
named: departments: names
coder: regions: ISO code
namer: region: names

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Source

INSEE. Code officiel g\'eographique au 1er janvier 2018.

Examples

```
data(departments)
print(departments)
```

---

discdd.misclass	*Misclassification ratio in functional discriminant analysis of discrete probability distributions.*

Description

Computes the one-leave-out misclassification ratio of the rule assigning T groups of individuals, one group after another, to the class of groups (among K classes of groups) which achieves the minimum of the distances or divergences between the probability distribution associated to the group to assign and the K probability distributions associated to the K classes.

Usage

```
discdd.misclass(xf, class.var, distance = c("l1", "l2", "chisqsym", "hellinger",
        "jeffreys", "jensen", "lp"), crit = 1, p)
```

Arguments

xf   object of class folderh with two data frames or list of arrays (or tables).

- If it is a folderh:
  - The first data.frame has at least two columns. One column contains the names of the T groups (all the names must be different). An other column is a factor with K levels partitionning the T groups into K classes.
  - The second one has $(q + 1)$ columns. The first q columns are factors (otherwise, they are coerced into factors). The last column is a factor with T levels defining T groups. Each group, say t, consists of n_t individuals.
- If it is a list of arrays or tables, the tth element $(t = 1, \ldots, T)$ is the table of the joint distribution (absolute or relative frequencies) of the tth group. These arrays have the same shape:
  Each array (or table) xf[[i]] has:
  - the same dimension(s). If $q = 1$ (univariate), dim(xf[[i]]) is an integer. If $q > 1$ (multivariate), dim(xf[[i]]) is an integer vector of length q.
  - the same dimension names dimnames(xf[[i]]) (is non NULL). These dimnames are the names of the variables.

class.var   string (if xf is an object of class "folderh") or data.frame with two columns (if
                   xf is a list of arrays).

- If xf is of class "folder", class.var is the name of the class variable.
- If xf is a list of arrays or a list of tables, class.var is a data.frame with
  at least two columns named "group" and "class". The "group" column
  contains the names of the T groups (all the names must be different). The
  "class" column is a factor with K levels partitioning the T groups into K
  classes.

distance   The distance or dissimilarity used to compute the distance matrix between the
                   densities. It can be:

- "l1" (default) the L^p distance with $p = 1$
- "l2" the L^p distance with $p = 2$
- "chisqsym" the symmetric Chi-squared distance
- "hellinger" the Hellinger metric (Matusita distance)
- "jeffreys" Jeffreys distance (symmetrised Kullback-Leibler divergence)
- "jensen" the Jensen-Shannon distance
- "lp" the L^p distance with p given by the argument p of the function.

crit   1 or 2. In order to select the densities associated to the classes. See Details.

p   integer. Optional. When distance = "lp" (L^p distance with $p > 2$), p is the
                   parameter of the distance.

Details

- If xf is an object of class "folderh" containing the data:

  The T probability distributions f_t corresponding to the T groups of individuals are estimated
  by frequency distributions within each group.

  To the class k consisting of T_k groups is associated the probability distribution g_k, knowing
  that when using the one-leave-out method, we do not include the group to assign in its class
  k. The crit argument selects the estimation method of the g_k's.

  - crit=1 The probability distribution g_k is estimated using the whole data of this class, that
    is the rows of x corresponding to the T_k groups of the class k.
    The estimation of the g_k's uses the same method as the estimation of the f_t's.
  - crit=2 The T_k probability distributions f_t are estimated using the corresponding data
    from xf. Then they are averaged to obtain an estimation of the density g_k, that is $g_k = \frac{1}{T_k} \sum f_t$.

- If xf is a list of arrays (or list of tables):

  The tth array is the joint frequency distribution of the tth group. The frequencies can be
  absolute or relative.

  To the class k consisting of T_k groups is associated the probability distribution g_k, knowing
  that when using the one-leave-out method, we do not include the group to assign in its class
  k. The crit argument selects the estimation method of the g_k's.

  - crit=1 $g_k = \frac{1}{\sum n_t} \sum n_t f_t$, where n_t is the total of xf[[t]].
    Notice that when xf[[t]] contains relative frequencies, its total is 1. That is equivalent
    to crit=2.
  - crit=2 $g_k = \frac{1}{T_k} \sum f_t$.

Value

Returns an object of class `discdd.misclass`, that is a list including:

`classification` data frame with 4 columns:

- factor giving the group name. The column name is the same as that of the column $(q + 1)$ of x,
- the prior class of the group if it is available, or NA if not,
- `alloc`: the class allocation computed by the discriminant analysis method,
- `misclassed`: boolean. `TRUE` if the group is misclassed, `FALSE` if it is well-classed, `NA` if the prior class of the group is unknown.

`confusion.mat` confusion matrix,

`misalloc.per.class`

the misclassification ratio per class,

`misclassed` the misclassification ratio,

`distances` matrix with T rows and K columns, of the distances (d_{tk}): d_{tk} is the distance between the group t and the class k,

`proximities` matrix of the proximity indices (in percents) between the groups and the classes. The proximity between the group t and the class k is: $(1/d_{tk})/\sum_{l=1}^{l=K}(1/d_{tl})$.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens, Centre de Recherches Archéologiques médiévales de Saverne, 5, 5-38.

Examples

```
# Example 1 with a folderh obtained by converting numeric variables
data("castles.dated")
stones <- castles.dated$stones
periods <- castles.dated$periods
stones$height <- cut(stones$height, breaks = c(19, 27, 40, 71), include.lowest = TRUE)
stones$width <- cut(stones$width, breaks = c(24, 45, 62, 144), include.lowest = TRUE)
stones$edging <- cut(stones$edging, breaks = c(0, 3, 4, 8), include.lowest = TRUE)
stones$boss <- cut(stones$boss, breaks = c(0, 6, 9, 20), include.lowest = TRUE )

castlefh <- folderh(periods, "castle", stones)

# Default: dist="l1", crit=1
discdd.misclass(castlefh, "period")

# Hellinger distance, crit=2
discdd.misclass(castlefh, "period", distance = "hellinger", crit = 2)
```

```
# Example 2 with a list of 96 arrays
data("dspgd2015")
data("departments")
classes <- departments[, c("coded", "namer")]
names(classes) <- c("group", "class")

# Default: dist="l1", crit=1
discdd.misclass(dspgd2015, classes)

# Hellinger distance, crit=2
discdd.misclass(dspgd2015, classes, distance = "hellinger", crit = 2)
```

---

discdd.predict   *Predicting the class of a group of individuals with discriminant anal-*
                       ysis of probability distributions.

Description

Assigns several groups of individuals, one group after another, to the class of groups (among K classes of groups) which achieves the minimum of the distances or divergences between the probability distribution associated to the group to assign and the K probability distributions associated to the K classes.

Usage

```
discdd.predict(xf, class.var, distance =  c("l1", "l2", "chisqsym", "hellinger",
          "jeffreys", "jensen", "lp"), crit = 1, misclass.ratio = FALSE, p)
```

Arguments

xf   object of class folderh with two data frames or list of arrays (or tables).

                  • If it is a folderh:
                       – The first data.frame has at least two columns. One column contains the
                         names of the T groups (all the names must be different). An other
                         column is a factor with K levels partitionning the T groups into K
                         classes.
                       – The second one has $(q + 1)$ columns. The first q columns are factors
                         (otherwise, they are coerced into factors). The last column is a factor
                         with T levels defining T groups. Each group, say t, consists of n_t
                         individuals.
                  • If it is a list of arrays or tables, the tth element ($t = 1, \ldots, T$) is the table
                    of the joint distribution (absolute or relative frequencies) of the tth group.
                    These arrays have the same shape:
                    Each array (or table) xf[[i]] has:
                       – the same dimension(s). If $q = 1$ (univariate), dim(xf[[i]]) is an
                         integer. If $q > 1$ (multivariate), dim(xf[[i]]) is an integer vector of
                         length q.
```

          – the same dimension names dimnames(xf[[i]]) (is non NULL). These dimnames are the names of the variables.

class.var       string (if xf is an object of class "folderh") or data.frame with two columns (if xf is a list of arrays).

- If xf is of class "folder", class.var is the name of the class variable.
- If xf is a list of arrays or a list of tables, class.var is a data.frame with at least two columns named "group" and "class". The "group" column contains the names of the $T$ groups (all the names must be different). The "class" column is a factor with $K$ levels partitioning the $T$ groups into $K$ classes.

distance       The distance or dissimilarity used to compute the distance matrix between the densities. It can be:

- "l1" (default) the $L^p$ distance with $p = 1$
- "l2" the $L^p$ distance with $p = 2$
- "chisqsym" the symmetric Chi-squared distance
- "hellinger" the Hellinger metric (Matusita distance)
- "jeffreys" Jeffreys distance (symmetrised Kullback-Leibler divergence)
- "jensen" the Jensen-Shannon distance
- "lp" the $L^p$ distance with $p$ given by the argument p of the function.

crit             1 or 2. In order to select the densities associated to the classes. See Details.

misclass.ratio  logical (default FALSE). If TRUE, the confusion matrix and misclassification ratio are computed on the groups whose prior class is known. In order to compute the misclassification ratio by the one-leave-out method, use the [discdd.misclass](discdd.misclass) function.

p              integer. Optional. When distance = "lp" ($L^p$ distance with $p > 2$), p is the parameter of the distance.

## Details

- If xf is an object of class "folderh" containing the data:

  The $T$ probability distributions $f_t$ corresponding to the $T$ groups of individuals are estimated by frequency distributions within each group.

  To the class $k$ consisting of $T_k$ groups is associated the probability distribution $g_k$. The crit argument selects the estimation method of the $g_k$'s.

  - crit=1 The probability distribution $g_k$ is estimated using the whole data of this class, that is the rows of x corresponding to the $T_k$ groups of the class $k$.
    The estimation of the $g_k$'s uses the same method as the estimation of the $f_t$'s.
  - crit=2 The $T_k$ probability distributions $f_t$ are estimated using the corresponding data from xf. Then they are averaged to obtain an estimation of the density $g_k$, that is $g_k = \frac{1}{T_k} \sum f_t$.

- If xf is a list of arrays (or list of tables):

  The $t^{th}$ array is the joint frequency distribution of the $t^{th}$ group. The frequencies can be absolute or relative.

  To the class $k$ consisting of $T_k$ groups is associated the probability distribution $g_k$. The crit argument selects the estimation method of the $g_k$'s.

– crit=1 $g_k = \frac{1}{\sum n_t} \sum n_t f_t$, where $n_t$ is the total of xf[[t]].
  Notice that when xf[[t]] contains relative frequencies, its total is 1. That is equivalent to crit=2.
– crit=2 $g_k = \frac{1}{T_k} \sum f_t$.

## Value

Returns an object of class discdd.predict, that is a list including:

prediction      data frame with 3 columns:

- factor giving the group name. The column name is the same as that of the column $(q + 1)$ of x,
- class.known: the prior class of the group if it is available, or NA if not,
- class.predict: the class allocation predicted by the discriminant analysis method. If misclass.ratio = TRUE, the class allocations are computed for all groups. Otherwise (default), they are computed only for the groups whose class is unknown.

distances       matrix with $T$ rows and $K$ columns, of the distances $(d_{tk})$: $d_{tk}$ is the distance between the group $t$ and the class $k$, computed with the measure given by argument,

proximities     matrix of the proximities (in percents). The proximity of a group $t$ to the class $k$ is computed as so: $(1/d_{tk})/ \sum_{l=1}^{l=K} (1/d_{tl})$.

confusion.mat   the confusion matrix (if misclass.ratio = TRUE)

misclassed      the misclassification ratio (if misclass.ratio = TRUE)

## Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

## References

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens, Centre de Recherches Archéologiques médiévales de Saverne, 5, 5-38.

## Examples

```
data(castles.dated)
data(castles.nondated)
stones <- rbind(castles.dated$stones, castles.nondated$stones)
periods <- rbind(castles.dated$periods, castles.nondated$periods)
stones$height <- cut(stones$height, breaks = c(19, 27, 40, 71), include.lowest = TRUE)
stones$width <- cut(stones$width, breaks = c(24, 45, 62, 144), include.lowest = TRUE)
stones$edging <- cut(stones$edging, breaks = c(0, 3, 4, 8), include.lowest = TRUE)
stones$boss <- cut(stones$boss, breaks = c(0, 6, 9, 20), include.lowest = TRUE )

castlesfh <- folderh(periods, "castle", stones)
```

```
# Default: dist="l1", crit=1
discdd.predict(castlesfh, "period")

# With the calculation of the confusion matrix and misclassification ratio
discdd.predict(castlesfh, "period", misclass.ratio = TRUE)

# Hellinger distance
discdd.predict(castlesfh, "period", distance = "hellinger")

# crit=2
discdd.predict(castlesfh, "period", crit = 2)
```

---

distl2d   *L^2 distance between probability densities*

Description

L^2 distance between two multivariate ($p > 1$) or univariate (dimension: $p = 1$) probability densities, estimated from samples.

Usage

```
distl2d(x1, x2, method = "gaussiand", check = FALSE, varw1 = NULL, varw2 = NULL)
```

Arguments

x1, x2	the samples from the probability densities (see l2d).
method	string. It can be:
	• "gaussiand" if the densities are considered to be Gaussian.
	• "kern" if they are estimated using the Gaussian kernel method.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices (if method = "gaussiand") or smoothing bandwidth matrices (if method = "kern") are not degenerate, before computing the inner product.
	Notice that if $p = 1$, it checks if the variances or smoothing parameters are not zero.
varw1, varw2	the bandwidths when the densities are estimated by the kernel method (see l2d).

Details

The function distl2d computes the distance between f_1 and f_2 from the formula

$$||f_1 - f_2||^2 = <f_1, f_1> + <f_2, f_2> -2 <f_1, f_2>$$

For some information about the method used to compute the L^2 inner product or about the arguments, see l2d.

Value

The L^2 distance between the two densities.

Be careful! If check = FALSE and one smoothing bandwidth matrix is degenerate, the result returned can not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

matdistl2d in order to compute pairwise distances between several densities.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
distl2d(x1, x2, method = "gaussiand")
distl2d(x1, x2, method = "kern")
distl2d(x1, x2, method = "kern", varw1 = v1, varw2 = v2)
```

---

distl2dnorm   *$L\hat{\ }2$ distance between $L\hat{\ }2$-normed probability densities*

Description

L^2 distance between two multivariate ($p > 1$) or univariate (dimension: $p = 1$) L^2-normed probability densities, estimated from samples, where a L^2-normed probability density is the original probability density function divided by its L^2-norm.

Usage

```
distl2dnorm(x1, x2, method = "gaussiand", check = FALSE, varw1 = NULL, varw2 = NULL)
```

Arguments

x1, x2	the samples from the probability densities (see [l2d](l2d).
method	string. It can be:

- "gaussiand" if the densities are considered to be Gaussian.
- "kern" if they are estimated using the Gaussian kernel method.

check	logical. When `TRUE` (the default is `FALSE`) the function checks if the covariance matrices (if `method = "gaussiand"`) or smoothing bandwidth matrices (if `method = "kern"`) are not degenerate, before computing the inner product.
	Notice that if $p = 1$, it checks if the variances or smoothing parameters are not zero.
varw1, varw2	the bandwidths when the densities are estimated by the kernel method (see `l2d`.

Details

Given densities f_1 and f_2, the function `distl2dnormpar` computes the distance between the L^2-normed densities $f_1/||f_1||$ and $f_2/||f_2||$:

$$2 - 2 < f_1, f_2 > /(||f_1||||f_2||)$$

For some information about the method used to compute the L^2 inner product or about the arguments, see `l2d`.

Value

The L^2 distance between the two L^2-normed densities.

Be careful! If `check = FALSE` and one smoothing bandwidth matrix is degenerate, the result returned can not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

`distl2d` for the distance between two probability densities.

`matdistl2dnorm` in order to compute pairwise distances between several L^2-normed densities.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
distl2dnorm(x1, x2, method = "gaussiand")
distl2dnorm(x1, x2, method = "kern")
distl2dnorm(x1, x2, method = "kern", varw1 = v1, varw2 = v2)
```

---

distl2dnormpar   *L^2 distance between L^2-normed Gaussian densities given their parameters*

Description

L^2 distance between two multivariate ($p > 1$) or univariate (dimension: $p = 1$) L^2-normed Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate) where a L^2-normed probability density is the original probability density function divided by its L^2-norm.

Usage

```
distl2dnormpar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1, mean2	means of the probability densities.
var1, var2	variances ($p = 1$) or covariance matrices ($p > 1$) of the probability densities.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate, before computing the inner product.
	If the variables are univariate, it checks if the variances are not zero.

Details

Given densities f_1 and f_2, the function distl2dnormpar computes the distance between the L^2-normed densities $f_1/||f_1||$ and $f_2/||f_2||$:

$$2 - 2 < f_1, f_2 > /(||f_1||||f_2||)$$

.

For some information about the method used to compute the L^2 inner product or about the arguments, see l2dpar; the norm $||f||$ of the multivariate Gaussian density f is equal to $(4\pi)^{-p/4}det(var)^{-1/4}$.

Value

The L^2 distance between the two L^2-normed Gaussian densities.

Be careful! If check = FALSE and one variance matrix is degenerated (or one variance is zero if the densities are univariate), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

distl2dpar for the distance between two probability densities.

matdistl2d in order to compute pairwise distances between several densities.

Examples

```
u1 <- c(1,1,1);
v1 <- matrix(c(4,0,0,0,16,0,0,0,25),ncol = 3);
u2 <- c(0,1,0);
v2 <- matrix(c(1,0,0,0,1,0,0,0,1),ncol = 3);
distl2dnormpar(u1,v1,u2,v2)
```

---

distl2dpar   *Lˆ2 distance between Gaussian densities given their parameters*

Description

L^2 distance between two multivariate ($p > 1$) or univariate (dimension: $p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate).

Usage

```
distl2dpar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1, mean2	means of the probability densities.
var1, var2	variances ($p = 1$) or covariance matrices ($p > 1$) of the probability densities.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate, before computing the inner product.
	If the variables are univariate, it checks if the variances are not zero.

Details

The function distl2dpar computes the distance between two densities, say f_1 and f_2, from the formula:

$$||f_1 - f_2||^2 = <f_1, f_1> + <f_2, f_2> -2<f_1, f_2>$$

.

For some information about the method used to compute the L^2 inner product or about the arguments, see l2dpar.

Value

The L^2 distance between the two densities.

Be careful! If check = FALSE and one variance matrix is degenerated (or one variance is zero if the densities are univariate), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[matdistl2d](#) in order to compute pairwise distances between several densities.

Examples

```
u1 <- c(1,1,1);
v1 <- matrix(c(4,0,0,0,16,0,0,0,25),ncol = 3);
u2 <- c(0,1,0);
v2 <- matrix(c(1,0,0,0,1,0,0,0,1),ncol = 3);
distl2dpar(u1,v1,u2,v2)
```

---

dspg   *Diploma x Socio professional group*

Description

Contingency tables of the counts of Diploma x Socio professional group of France

Usage

```
data(dspg)
```

Format

dspg is a list of 7 arrays (each one corresponding to a year: 1968, 1975, 1982, 1990, 1999, 2010, 2015) of 4 rows (each one corresponding to a level of diploma) and 6 columns (each one corresponding to a socio professional group).

csp: Socio professional group

diplome: Diploma

agri: farmer (agriculteur)

arti: craftsperson (artisan)

cadr: senior manager (cadre sup\'erieur)

pint: middle manager (profession interm\'ediaire)

empl: employee (employ\'e)

ouvr: worker (ouvrier)

```
bepc:  brevet
cap:  NVQ (cap)
bac:  baccalaureate
sup:  higher education (sup\'erieur)
```

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Source

INSEE. Population active de 25 à 54 ans ayant un emploi et chômeurs par catégorie socioprofessionnelle et diplôme par commune et département (1968 à 2015).

Examples

```
data(dspg)
names(dspg)
print(dspg[[1]])
```

---

dspgd2015   *Diploma x Socio professional group by departement in 2015*

Description

Contingency tables of the counts of Diploma x Socio professional group by metroplitan France departement in year 2015.

Usage

```
data(dspgd2015)
```

Format

dspgd2015 is a list of 96 arrays (each one corresponding to a department, designated by its official geographical code) of 4 rows (each one corresponding to a level of diploma) and 6 columns (each one corresponding to a socio professional group).

```
csp:  Socio professional group
diplome:  Diploma
agri:  farmer (agriculteur)
arti:  craftsperson (artisan)
cadr:  senior manager (cadre sup\'erieur)
pint:  middle manager (profession interm\'ediaire)
empl:  employee (employ\'e)
```

ouvr: worker (ouvrier)

bepc: brevet

cap: NVQ (cap)

bac: baccalaureate

sup: higher education (sup\'erieur)

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

Source

INSEE. Population active de 25 à 54 ans ayant un emploi et chômeurs par catégorie socioprofessionnelle et diplôme par commune et département (1968 à 2015).

Examples

```
data(dspgd2015)
names(dspgd2015)
print(dspgd2015[[1]])
```

---

dstatis.inter   *Dual STATIS method (interstructure stage)*

Description

Performs the first stage (interstructure) of the dual STATIS method in order to describe a data folder, consisting of T groups of individuals on which are observed p variables. It returns an object of class dstatis.

Usage

```
dstatis.inter(xf, normed = TRUE, centered = TRUE, data.scaled = FALSE, nb.factors = 3,
      nb.values = 10, sub.title = "", plot.eigen = TRUE, plot.score = FALSE,
      nscore = 1:3, group.name = "group", filename = NULL)
```

Arguments

xf	object of class folder. Its elements are data frames with p numeric columns. If there are non numeric columns, there is an error. The tth element ($t = 1, \ldots, T$) matches with the tth group.
normed	logical. If TRUE (default), the scalar products are normed.
centered	logical. If TRUE (default), the scalar products are centered.
data.scaled	logical. If TRUE, the data of each group are centered and scaled. The analysis is then performed on the correlation matrices. If FALSE (default), the analysis is performed on the covariance matrices.

nb.factors	numeric. Number of returned principal scores (default nb.factors = 3).
nb.values	numerical. Number of returned eigenvalues (default nb.values = 10).
sub.title	string. If provided, the subtitle for the graphs.
plot.eigen	logical. If TRUE (default), the barplot of the eigenvalues is plotted.
plot.score	logical. If TRUE, the graphs of principal scores are plotted. A new graphic device is opened for each pair of principal scores defined by nscore argument.
nscore	numeric vector. If plot.score = TRUE, the numbers of the principal scores which are plotted. By default it is equal to nscore = 1:3. Its components cannot be greater than nb.factors.
group.name	string. Name of the grouping variable. Default: groupname = "group".
filename	string. Name of the file in which the results are saved. By default (filename = NULL) the results are not saved.

Details

The covariance matrices (if data.scale is FALSE) or correlation matrices (if TRUE) per group are computed. The matrix W of the scalar products between these covariance matrices is then computed.

To perform the STATIS method, see the function DSTATIS of the multigroup package.

Value

Returns an object of class dstatis, that is a list including:

inertia	data frame of the eigenvalues and percentages of inertia.
contributions	data frame of the contributions to the first nb.factors principal components.
qualities	data frame of the qualities on the first nb.factors principal factors.
scores	data frame of the first nb.factors scores of the spectral decomposition of W.
norm	vector of the L^2 norms of the densities.
means	list of the means.
variances	list of the covariance matrices.
correlations	list of the correlation matrices.
skewness	list of the skewness coefficients.
kurtosis	list of the kurtosis coefficients.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Lavit, C., Escoufier, Y., Sabatier, R., Traissac, P. (1994). The ACT (STATIS method). Computational Statistics & Data Analysis, 18 (1994), 97-119.

See Also

[print.dstatis](#), [plot.dstatis](#), [interpret.dstatis](#).

Examples

```
data(roses)
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])

# Dual STATIS on the covariance matrices
result1 <- dstatis.inter(rosesf, data.scaled = FALSE, group.name = "rose")
print(result1)
plot(result1)

# Dual STATIS on the correlation matrices
result2 <- dstatis.inter(rosesf, data.scaled = FALSE, group.name = "rose")
print(result2)
plot(result2)
```

---

fdiscd.misclass	*Misclassification ratio in functional discriminant analysis of probability densities.*

Description

Computes the one-leave-out misclassification ratio of the rule assigning T groups of individuals, one group after another, to the class of groups (among K classes of groups) which achieves the minimum of the distances or divergences between the density function associated to the group to assign and the K density functions associated to the K classes.

Usage

```
fdiscd.misclass(xf, class.var, gaussiand = TRUE,
         distance = c("jeffreys", "hellinger", "wasserstein", "l2", "l2norm"),
           crit = 1, windowh = NULL)
```

Arguments

xf   object of class [folderh](#) with two data frames:

- The first one has at least two columns. One column contains the names of the T groups (all the names must be different). An other column is a factor with K levels partitionning the T groups into K classes.
- The second one has $(p + 1)$ columns. The first p columns are numeric (otherwise, there is an error). The last column is a factor with T levels defining T groups. Each group, say t, consists of n_t individuals.

class.var   string. The name of the class variable.

distance   The distance or dissimilarity used to compute the distance matrix between the densities. It can be:

- "jeffreys" (default) the Jeffreys measure (symmetrised Kullback-Leibler divergence),
- "hellinger" the Hellinger (Matusita) distance,
- "wasserstein" the Wasserstein distance,
- "l2" the L^2 distance,
- "l2norm" (only available when crit = 1) the densities are normed and the L^2 distance between these normed densities is used;

If gaussiand = FALSE, the densities are estimated by the Gaussian kernel method and the distance is "l2" or "l2norm".

crit   1, 2 or 3. In order to select the densities associated to the classes. See Details.

If distance is "hellinger", "jeffreys" or "wasserstein", crit is necessarily 1 (see Details).

gaussiand   logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method.

If distance is "hellinger", "jeffreys" or "wasserstein", gaussiand is necessarily TRUE.

windowh   strictly positive numeric value. If windowh = NULL (default), the bandwidths are computed using the [bandwidth.parameter](#) function.

Omitted when distance is "hellinger", "jeffreys" or "wasserstein" (see Details).

Details

The T probability densities f_t corresponding to the T groups of individuals are either parametrically estimated (gaussiand = TRUE) or estimated using the Gaussian kernel method (gaussiand = FALSE). In the latter case, the windowh argument provides the list of the bandwidths to be used. Notice that in the multivariate case ($p>1$), the bandwidths are positive-definite matrices.

The argument windowh is a numerical value, the matrix bandwidth is of the form hS, where S is either the square root of the covariance matrix ($p>1$) or the standard deviation of the estimated density.

If windowh = NULL (default), h in the above formula is computed using the [bandwidth.parameter](#) function.

To the class k consisting of T_k groups is associated the density denoted g_k. The crit argument selects the estimation method of the K densities g_k.

1. The density g_k is estimated using the whole data of this class, that is the rows of x corresponding to the T_k groups of the class k.

   The estimation of the densities g_k uses the same method as the estimation of the f_t.

2. The T_k densities f_t are estimated using the corresponding data from x. Then they are averaged to obtain an estimation of the density g_k, that is $g_k = \frac{1}{T_k} \sum f_t$.

3. Each previous density f_t is weighted by n_t (the number of rows of x corresponding to f_t). Then they are averaged, that is $g_k = \frac{1}{\sum n_t} \sum n_t f_t$.

The last two methods are only available for the L^2-distance. If the divergences between densities are computed using the Hellinger or Wasserstein distance or Jeffreys measure, only the first of these methods is available.

The distance or dissimilarity between the estimated densities is either the L^2 distance, the Hellinger distance, Jeffreys measure (symmetrised Kullback-Leibler divergence) or the Wasserstein distance.

- If it is the L^2 distance (`distance="l2"` or `distance="l2norm"`), the densities can be either parametrically estimated or estimated using the Gaussian kernel.
- If it is the Hellinger distance (`distance="hellinger"`), Jeffreys measure (`distance="jeffreys"`) or the Wasserstein distance (`distance="wasserstein"`), the densities are considered Gaussian and necessarily parametrically estimated.

Value

Returns an object of class `fdiscd.misclass`, that is a list including:

`classification`   data frame with 4 columns:

- factor giving the group name. The column name is the same as that of the column $(p + 1)$ of x,
- the prior class of the group if it is available, or NA if not,
- `alloc`: the class allocation computed by the discriminant analysis method,
- `misclassed`: boolean. TRUE if the group is misclassed, FALSE if it is well-classed, NA if the prior class of the group is unknown.

`confusion.mat`   confusion matrix,

`misalloc.per.class`

the misclassification ratio per class,

`misclassed`   the misclassification ratio,

`distances`   matrix with T rows and K columns, of the distances (d_{tk}): d_{tk} is the distance between the group t and the class k, computed with the measure given by argument `distance` (L^2-distance, Hellinger distance or Jeffreys measure),

`proximities`   matrix of the proximity indices (in percents) between the groups and the classes. The proximity of the group t to the class k is computed as so: $(1/d_{tk})/\sum_{l=1}^{l=K}(1/d_{tl})$.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (2004). Discriminant analysis with independently repeated multivariate measurements: an L^2 approach. Computational Statistics & Data Analysis, 47, 823-843.

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens. Centre de Recherches Archéologiques Médiévales de Saverne, 5, 5-38.

Examples

```
data(castles.dated)
castles.stones <- castles.dated$stones
castles.periods <- castles.dated$periods
```

```
castlesfh <- folderh(castles.periods, "castle", castles.stones)
result <- fdiscd.misclass(castlesfh, "period")
print(result)
```

---

fdiscd.predict   *Predicting the class of a group of individuals with discriminant analysis of probability densities.*

Description

Assigns several groups of individuals, one group after another, to the class of groups (among K classes of groups) which achieves the minimum of the distances or divergences between the density function associated to the group to assign and the K density functions associated to the K classes.

Usage

```
fdiscd.predict(xf, class.var, gaussiand = TRUE,
         distance = c("jeffreys", "hellinger", "wasserstein", "l2", "l2norm"),
           crit = 1, windowh = NULL, misclass.ratio = FALSE)
```

Arguments

xf   object of class folderh with two data frames:

- The first one has at least two columns. One column contains the names of the T groups (all the names must be different). An other column is a factor with K levels partitionning the T groups into K classes..
- The second one has $(p + 1)$ columns. The first p columns are numeric (otherwise, there is an error). The last column is a factor with T levels defining T groups. Each group, say t, consists of n_t individuals.

           Notice that for the versions earlier than 2.0, fdiscd.predict applied to two data frames.

class.var   string. The name of the class variable.

distance   The distance or divergence used to compute the distance matrix between the densities. It can be:

- "jeffreys" (default) Jeffreys measure (symmetrised Kullback-Leibler divergence),
- "hellinger" the Hellinger (Matusita) distance,
- "wasserstein" the Wasserstein distance,
- "l2" the L^2 distance,
- "l2norm" the densities are normed and the L^2 distance between these normed densities is used;

           If gaussiand = FALSE, the densities are estimated by the Gaussian kernel method and the distance is "l2" or "l2norm".

crit	1, 2 or 3. In order to select the densities associated to the classes. See Details.
	If distance is "hellinger", "jeffreys" or "wasserstein", crit is necessarily 1 (see Details).
gaussiand	logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method.
	If distance is "hellinger", "jeffreys" or "wasserstein", gaussiand is necessarily TRUE.
windowh	strictly positive number. If windowh = NULL (default), the bandwidths are computed using the bandwidth.parameter function.
	Omitted when distance is "hellinger", "jeffreys" or "wasserstein" (see Details).
misclass.ratio	logical (default FALSE). If TRUE, the confusion matrix and misclassification ratio are computed on the groups whose prior class is known. In order to compute the misclassification ratio by the one-leave-out method, use the fdiscd.misclass function.

Details

To the group t is associated the density denoted f_t. To the class k consisting of T_k groups is associated the density denoted g_k. The crit argument selects the estimation method of the K densities g_k.

1. The density g_k is estimated using the whole data of this class, that is the rows of x corresponding to the T_k groups of the class k.

2. The T_k densities f_t are estimated using the corresponding data from x. Then they are averaged to obtain an estimation of the density g_k, that is $g_k = (1/T_k) \sum f_t$.

3. Each previous density f_t is weighted by n_t (the number of rows of x corresponding to f_t). Then they are averaged, that is $g_k = (1/ \sum n_t) \sum n_t f_t$.

The last two methods are available only for the L^2-distance. If the divergences between densities are computed using the Hellinger or Wasserstein distance or Jeffreys measure, only the first of these methods is available.

Value

Returns an object of class fdiscd.predict, that is a list including:

prediction	data frame with 3 columns:
	• factor giving the group name. The column name is the same as that of the column $(p+1)$ of x,
	• class.known: the prior class of the group if it is available, or NA if not,
	• class.predict: the class allocation predicted by the discriminant analysis method. If misclass.ratio = TRUE, the class allocations are computed for all groups. Otherwise (default), they are computed only for the groups whose class is unknown.
distances	matrix with T rows and K columns, of the distances (d_{tk}): d_{tk} is the distance between the group t and the class k, computed with the measure given by argument distance (L^2-distance, Hellinger distance or jeffreys measure),

proximities	matrix of the proximities (in percents). The proximity of a group t to the class k is computed as so: $(1/d_{tk})/\sum_{l=1}^{l=K}(1/d_{tl})$.
confusion.mat	the confusion matrix (if misclass.ratio = TRUE)
misclassed	the misclassification ratio (if misclass.ratio = TRUE)

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (2004). Discriminant analysis with independently repeated multivariate measurements: an L^2 approach. Computational Statistics & Data Analysis, 47, 823-843.

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens. Centre de Recherches Archéologiques Médiévales de Saverne, 5, 5-38.

Examples

```
data(castles.dated)
data(castles.nondated)
castles.stones <- rbind(castles.dated$stones, castles.nondated$stones)
castles.periods <- rbind(castles.dated$periods, castles.nondated$periods)
castlesfh <- folderh(castles.periods, "castle", castles.stones)

# With the L^2-distance

# - crit=1
resultl2.1 <- fdiscd.predict(castlesfh, "period", distance="l2", crit=1)
print(resultl2.1)

# - crit=2
## Not run:
resultl2.2 <- fdiscd.predict(castlesfh, "period", distance="l2", crit=2)
print(resultl2.2)

## End(Not run)

# - crit=3
resultl2.3 <- fdiscd.predict(castlesfh, "period", distance="l2", crit=3)
print(resultl2.3)

# With the Hellinger distance
resulthelling <- fdiscd.predict(castlesfh, "period", distance="hellinger")
print(resulthelling)

# With jeffreys measure
resultjeff <- fdiscd.predict(castlesfh, "period", distance="jeffreys")
print(resultjeff)
```

---

fhclustd   *Hierarchic cluster analysis of probability densities*

Description

Performs functional hierarchic cluster analysis of probability densities. It returns an object of class [fhclustd](). It applies [hclust]() to the distance matrix between the T densities.

Usage

```
fhclustd(xf, group.name  = "group", gaussiand = TRUE, distance = c("jeffreys",
            "hellinger", "wasserstein", "l2", "l2norm"), windowh=NULL,
        data.centered = FALSE, data.scaled = FALSE, common.variance = FALSE,
          sub.title = "", filename = NULL, method.hclust = "complete")
```

Arguments

xf   object of class "[folder]()" or data.frame.

- If it is an object of class "folder", its elements are data frames with p numeric columns. If there are non numeric columns, there is an error. The tth element ($t = 1, \ldots, T$) matches with the tth group.
- If it is a data frame, the column with name given by the group.name argument is a factor giving the groups. The other columns are all numeric; otherwise, there is an error.

group.name   string.

- If xf is an object of class "[folder]()", it is the name of the grouping variable in the returned results. The default is groupname = "group".
- If xf is a data frame, it is the name of the column of xf containing the groups.

gaussiand   logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method.

                If distance is "hellinger", "jeffreys" or "wasserstein", gaussiand is necessarily TRUE (see Details).

distance   The distance or divergence used to compute the distance matrix between the densities. It can be:

- "jeffreys" (default) Jeffreys measure (symmetrised Kullback-Leibler divergence),
- "hellinger" the Hellinger (Matusita) distance,
- "wasserstein" the Wasserstein distance,
- "l2" the L^2 distance,
- "l2norm" the densities are normed and the L^2 distance between these normed densities is used;

                If gaussiand = FALSE, the densities are estimated by the Gaussian kernel method and the distance can be "l2" (default) or "l2norm".

windowh   either a list of T bandwidths (one per density associated to a group), or a strictly
                 positive number. If windowh = NULL (default), the bandwidths are automatically
                 computed. See Details.

                 Omitted when distance is "hellinger", "jeffreys" or "wasserstein" (see
                 Details).

data.centered   logical. If TRUE (default is FALSE), the data of each group are centered.

data.scaled   logical. If TRUE (default is FALSE), the data of each group are centered (even if
                 data.centered = FALSE) and scaled.

common.variance

                 logical. If TRUE (default is FALSE), a common covariance matrix (or correlation
                 matrix if data.scaled = TRUE), computed on the whole data, is used. If FALSE
                 (default), a covariance (or correlation) matrix per group is used.

sub.title   string. If provided, the subtitle for the graphs.

filename   string. Name of the file in which the results are saved. By default (filename =
                 NULL) the results are not saved.

method.hclust   the agglomeration method to be used for the clustering. See the method argu-
                 ment of the hclust function.

Details

In order to compute the distances/dissimilarities between the groups, the T probability densities
f_t corresponding to the T groups of individuals are either parametrically estimated (gaussiand =
TRUE) or estimated using the Gaussian kernel method (gaussiand = FALSE). In the latter case, the
windowh argument provides the list of the bandwidths to be used. Notice that in the multivariate
case ($p>1$), the bandwidths are positive-definite matrices. The distances between the T groups of
individuals are given by the L^2-distances between the T probability densities f_t corresponding to
these groups. The hclust function is then applied to the distance matrix to perform the hierarchical
clustering on the T groups.

If windowh is a numerical value, the matrix bandwidth is of the form hS, where S is either the
square root of the covariance matrix ($p>1$) or the standard deviation of the estimated density.

If windowh = NULL (default), h in the above formula is computed using the bandwidth.parameter
function.

The distance or dissimilarity between the estimated densities is either the L^2 distance, the Hellinger
distance, Jeffreys measure (symmetrised Kullback-Leibler divergence) or the Wasserstein distance.

- If it is the L^2 distance (distance="l2" or distance="l2norm"), the densities can be either
  parametrically estimated or estimated using the Gaussian kernel.

- If it is the Hellinger distance (distance="hellinger"), Jeffreys measure (distance="jeffreys")
  or the Wasserstein distance (distance="wasserstein"), the densities are considered Gaus-
  sian and necessarily parametrically estimated.

Value

Returns an object of class fhclustd, that is a list including:

distances   matrix of the L^2-distances between the estimated densities.

clust   an object of class hclust.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

fdiscd.predict, fdiscd.misclass

Examples

```
data(castles.dated)
stones <- castles.dated$stones
periods <- castles.dated$periods

periods123 <- periods[periods$period %in% 1:3, "castle"]
stones123 <- stones[stones$castle %in% periods123, ]
stones123$castle <- as.factor(as.character(stones123$castle))
yf <- as.folder(stones123)


# Jeffreys measure (default):

resultjef <- fhclustd(yf)
print(resultjef)
print(resultjef, dist.print = TRUE)
plot(resultjef)
plot(resultjef, hang = -1)

# Use cutree (stats package) to get the partition
cutree(resultjef$clust, k = 1:4)
cutree(resultjef$clust, k = 5)
cutree(resultjef$clust, h = 0.041)


# Applied to a data frame (Jeffreys measure):

fhclustd(stones123, group.name = "castle")

# Use cutree (stats package) to get the partition
cutree(resultjef$clust, k = 1:4)
cutree(resultjef$clust, k = 5)
cutree(resultjef$clust, h = 0.041)


# Hellinger distance:

resulthel <- fhclustd(yf, distance = "hellinger")
print(resulthel)
print(resulthel, dist.print = TRUE)
plot(resulthel)
plot(resulthel, hang = -1)

# Use cutree (stats package) to get the partition
```

```
cutree(resulthel$clust, k = 1:4)
cutree(resulthel$clust, k = 5)
cutree(resulthel$clust, h = 0.041)


## Not run:
# L2-distance:

xf <- as.folder(stones)
result <- fhclustd(xf, distance = "l2")
print(result)
print(result, dist.print = TRUE)
plot(result)
plot(result, hang = -1)

# Use cutree (stats package) to get the partition
cutree(result$clust, k = 1:5)
cutree(result$clust, k = 5)
cutree(result$clust, h = 0.18)

## End(Not run)

periods123 <- periods[periods$period %in% 1:3, "castle"]
stones123 <- stones[stones$castle %in% periods123, ]
stones123$castle <- as.factor(as.character(stones123$castle))
yf <- as.folder(stones123)
result123 <- fhclustd(yf, distance = "l2")
print(result123)
print(result123, dist.print = TRUE)
plot(result123)
plot(result123, hang = -1)

# Use cutree (stats package) to get the partition
cutree(result123$clust, k = 1:4)
cutree(result123$clust, k = 5)
cutree(result123$clust, h = 0.041)
```

---

floribundity	*Rose flowering*

Description

These data are collected on eight rosebushes from four varieties, during summer 2010 in Angers, France. They give measures of the flowering.

Usage

```
data("floribundity")
```

Format

floribundity is a list of 16 data frames, each corresponding to an observation date. Each one of these data frames has 3 or 4 columns:

- rose: the number of the rosebush, that is an identifier.
- variety: factor. The variety of the rosebush.
- area (when available): numeric. The ratio of flowering area to the whole plant area, measured on the photograph of the rosebush.
- nflowers (when available): integer. The number of flowers on the rosebush.

The row names of these data frames are the rose identifiers.

Examples

```
data(floribundity)
foldt <- foldert(floribundity, times = as.Date(names(floribundity)), rows.select = "union")
summary(foldt)
```

---

fmdsd   *Multidimensional scaling of probability densities*

Description

Applies the multidimensional scaling (MDS) method to probability densities in order to describe a data folder, consisting of T groups of individuals on which are observed p variables. It returns an object of class fmdsd. It applies cmdscale to the distance matrix between the T densities.

Usage

```
fmdsd(xf, group.name = "group", gaussiand = TRUE, distance = c("jeffreys", "hellinger",
    "wasserstein", "l2", "l2norm"), windowh=NULL, data.centered = FALSE,
    data.scaled = FALSE, common.variance = FALSE, add = TRUE, nb.factors = 3,
  nb.values = 10, sub.title = "", plot.eigen = TRUE, plot.score = FALSE, nscore = 1:3,
    filename = NULL)
```

Arguments

xf   object of class "folder" or data.frame.

- If it is an object of class "folder", its elements are data frames with p numeric columns. If there are non numeric columns, there is an error. The tth element ($t = 1, \ldots, T$) matches with the tth group.
- If it is a data frame, the column with name given by the group.name argument is a factor giving the groups. The other columns are all numeric; otherwise, there is an error.

group.name   string.

- If xf is an object of class `"folder"`, it is the name of the grouping variable in the returned results. The default is groupname = "group".
- If xf is a data frame, it is the name of the column of xf containing the groups.

gaussiand   logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method.

distance   The distance or divergence used to compute the distance matrix between the densities.

If gaussiand = TRUE, the densities are parametrically estimated and the distance can be:

- `"jeffreys"` (default) Jeffreys measure (symmetrised Kullback-Leibler divergence),
- `"hellinger"` the Hellinger (Matusita) distance,
- `"wasserstein"` the Wasserstein distance,
- `"l2"` the L^2 distance,
- `"l2norm"` the densities are normed and the L^2 distance between these normed densities is used;

If gaussiand = FALSE, the densities are estimated by the Gaussian kernel method and the distance can be `"l2"` (default) or `"l2norm"`.

windowh   either a list of T bandwidths (one per density associated to a group), or a strictly positive number. If windowh = NULL (default), the bandwidths are automatically computed. See Details.

Omitted when distance is `"hellinger"`, `"jeffreys"` or `"wasserstein"` (see Details).

data.centered   logical. If TRUE (default is FALSE), the data of each group are centered.

data.scaled   logical. If TRUE (default is FALSE), the data of each group are centered (even if data.centered = FALSE) and scaled.

common.variance

logical. If TRUE (default is FALSE), a common covariance matrix (or correlation matrix if data.scaled = TRUE), computed on the whole data, is used. If FALSE (default), a covariance (or correlation) matrix per group is used.

add   logical indicating if an additive constant should be computed and added to the non diagonal dissimilarities such that the modified dissimilarities are Euclidean (default TRUE; see add argument of `cmdscale`).

nb.factors   numeric. Number of returned principal coordinates (default nb.factors = 3).

Warning: The `plot.fmdsd` and `interpret.fmdsd` functions cannot take into account more than nb.factors principal factors.

nb.values   numeric. Number of returned eigenvalues (default nb.values = 10).

sub.title   string. Subtitle for the graphs (default NULL).

plot.eigen   logical. If TRUE (default), the barplot of the eigenvalues is plotted.

plot.score   logical. If TRUE, the graphs of new coordinates are plotted. A new graphic device is opened for each pair of coordinates defined by nscore argument.

nscore   numeric vector. If plot.score = TRUE, the numbers of the principal coordinates
                which are plotted. By default it is equal to nscore = 1:3. Its components cannot
                be greater than nb.factors.

filename   string. Name of the file in which the results are saved. By default (filename =
                NULL) they are not saved.

Details

In order to compute the distances/dissimilarities between the groups, the T probability densities
f_t corresponding to the T groups of individuals are either parametrically estimated (gaussiand =
TRUE) or estimated using the Gaussian kernel method (gaussiand = FALSE). In the latter case, the
windowh argument provides the list of the bandwidths to be used. Notice that in the multivariate
case (p>1), the bandwidths are positive-definite matrices.

If windowh is a numerical value, the matrix bandwidth is of the form hS, where S is either the
square root of the covariance matrix (p>1) or the standard deviation of the estimated density.

If windowh = NULL (default), h in the above formula is computed using the bandwidth.parameter
function.

The distance or dissimilarity between the estimated densities is either the L^2 distance, the Hellinger
distance, Jeffreys measure (symmetrised Kullback-Leibler divergence) or the Wasserstein distance.

- If it is the L^2 distance (distance="l2" or distance="l2norm"), the densities can be either
  parametrically estimated or estimated using the Gaussian kernel.

- If it is the Hellinger distance (distance="hellinger"), Jeffreys measure (distance="jeffreys")
  or the Wasserstein distance (distance="wasserstein"), the densities are considered Gaus-
  sian and necessarily parametrically estimated.

Value

Returns an object of class fmdsd, i.e. a list including:

inertia   data frame of the eigenvalues and percentages of inertia.

scores   data frame of the nb.factors first principal coordinates.

means   list of the means.

variances   list of the covariance matrices.

correlations   list of the correlation matrices.

skewness   list of the skewness coefficients.

kurtosis   list of the kurtosis coefficients.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Delicado, P. (2011). Dimensionality reduction when data are density functions. Computational Statistics & Data Analysis, 55, 401-420.

Yousfi, S., Boumaza, R., Aissani, D., Adjabi, S. (2014). Optimal bandwith matrices in functional principal component analysis of density function. Journal of Statistical Computation and Simulation, 85 (11), 2315-2330.

Cox, T.F., Cox, M.A.A. (2001). Multimensional Scaling, second ed. Chapman & Hall/CRC.

See Also

fpcad print.fmdsd, plot.fmdsd, interpret.fmdsd, bandwidth.parameter

Examples

```
data(roses)
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])

# MDS on Gaussian densities (on sensory data)

# using jeffreys measure (default):
resultjeff <- fmdsd(rosesf, distance = "jeffreys")
print(resultjeff)
plot(resultjeff)

## Not run:
# Applied to a data frame:
resultjeffdf <- fmdsd(roses[,c("Sha","Den","Sym","rose")],
                      distance = "jeffreys", group.name = "rose")
print(resultjeffdf)
plot(resultjeffdf)

## End(Not run)

# using the Hellinger distance:
resulthellin <- fmdsd(rosesf, distance = "hellinger")
print(resulthellin)
plot(resulthellin)

# using the Wasserstein distance:
resultwass <- fmdsd(rosesf, distance = "wasserstein")
print(resultwass)
plot(resultwass)

# Gaussian case, using the L2-distance:
resultl2 <- fmdsd(rosesf, distance = "l2")
print(resultl2)
plot(resultl2)
```

```
# Gaussian case, using the L2-distance between normed densities:
resultl2norm <- fmdsd(rosesf, distance = "l2norm")
print(resultl2norm)
plot(resultl2norm)

## Not run:
# Non Gaussian case, using the L2-distance,
# the densities are estimated using the Gaussian kernel method:
result <- fmdsd(rosesf, distance = "l2", gaussiand = FALSE, group.name = "rose")
print(result)
plot(result)

## End(Not run)
```

---

folder   *Folder of data sets*

Description

Creates an object of class "folder" (called folder below), that is a list of data frames with the same
column names. Thus, these data sets are on the same variables. They can be on the same individuals
or not.

Usage

```
folder(x1, x2 = NULL, ..., cols.select = "intersect", rows.select = "")
```

Arguments

x1	data frame (can also be a tibble) or list of data frames.
	• If x1 is a data frame, x2 must be provided.
	• If x1 is a list of data frames, its elements are the datasets of the folder. In this case, there is no x2 argument.
x2	data frame. Must be provided if x1 is a data frame.
...	optional. One or several data frames. When x1 and x2 are data frames, these are the other data frames.
cols.select	string. Gives the method used to choose the column names of the data frames of the folder. This argument can be:
	"intersect" (default) the column names of the data frames in the folder are the intersection of the column names of all the data frames given as arguments.
	"union" the column names of the data frames in the folder are the union of the column names of all the data frames given as arguments. When necessary, the rows of the returned data frames are completed by NA. If cols.select is a character vector, it gives the column names selected in the data frames given as arguments. The corresponding columns constitute the columns of

the elements of the returned folder. Notice that when a column name is not present in all data frames (given as arguments), the data are completed by NA.

rows.select string. Gives the method used to choose the row names of the data frames of the folder. This argument can be:

 `""` (default) the data frames of the folder have the same rows as those which were passed as arguments.

 `"intersect"` the row names of the data frames in the folder are the intersection of the row names of all the data frames given as arguments.

 `"union"` the row names of the data frames in the folder are the union of the row names of all the data frames given as arguments. When necessary, the columns of the data frames returned are completed by NA.

Details

The class folder has a logical attributes attr(,"same.rows").

The data frames in the returned folder all have the same column names. That means that the same variables are observed in every data sets.

If the rows.select argument is "union" or "intersect", the elements of the returned folder have the same rows. That means that the same individuals are present in every data sets. This allows to consider the evolution of each individual among time.

If rows.select is "", every rows of this folder are different, and the row names are made unique by adding the name of the data frame to the row names. In this case, The individuals of the data sets are assumed to be all different. Or, at least, the user does not mind if they are the same or not.

Value

Returns an object of class "folder", that is a list of data frames.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

is.folder to test if an object is of class folder. folderh to build a folder of several data frames with a hierarchic relation between each pair of consecutive data frames.

Examples

```
# First example
x1 <- data.frame(x = rnorm(10), y = 1:10)
x2 <- data.frame(x = rnorm(10), z = runif(10, 1, 10))
f1 <- folder(x1, x2)
print(f1)

f2 <- folder(x1, x2, cols.select = "union")
print(f2)
```

```
#Second example
data(iris)
iris.set <- iris[iris$Species == "setosa", 1:4]
iris.ver <- iris[iris$Species == "versicolor", 1:4]
iris.vir <- iris[iris$Species == "virginica", 1:4]
irisf1 <- folder(iris.set, iris.ver, iris.vir)
print(irisf1)

listofdf <- list(df1 = iris.set,df2 = iris.ver,df3 = iris.vir)
irisf2 <- folder(listofdf,x2 = NULL)
print(irisf2)
```

---

folderh   *Hierarchic folder of n data frames related in pairs by (n-1) keys*

Description

Creates an object of class folderh, that is a list of $n > 1$ data frames whose rows are related by (n-1) keys, each key defining a relation "1 to N" between the two adjacent data frames passed as arguments of the function.

Usage

```
folderh(df1, key1, df2, ..., na.rm = TRUE)
```

Arguments

df1	data frame (can also be a tibble) with at least two columns. It contains a factor (whose name is given by key1 argument) whose levels are taken exactly once.
key1	character string. The name of the factor of the data frames df1 and df2 which contains the key of the relations "1 to N" between the two datasets.
df2	data frame (or tibble) with at least two columns. It contains a factor column (named by keys argument) with the same levels as df1[, key1] (see Details).
...	optional. One or several supplementary character strings and data frames, ordered as follows: key2, df3, ... The argument key2 indicates the key defining the relation "1 to N" between the data frames df2 and df3, and so on.
na.rm	logical. If TRUE, the rows of each data frame for which the key is NA are removed.

Details

The object of class folderh is a list of $n \geq 2$ data frames.

- If no optional arguments are given via ..., that is $n = 2$, the two data frames of the list have a column named by the attribute attr(, "keys") (argument key1), which is a factor with the same levels. Each one of these levels occur exactly once in the first data frame of the list.

- If some supplementary data frames and supplementary strings key2, df3, ... are given as optional arguments, n is the number of data frames given as arguments. Then, the attribute attr(, "keys") is a vector of $n - 1$ character strings. For $i = 1, \ldots, N - 1$, its i-th element is the name of a column of the i-th and $(i + 1)$-th data frames of the folderh, which are factors with the same levels. Each one of these levels occur exactly once in the i-th data frame.

If there are more than two data frames, folderh computes a folderh with the two last data frames, and then uses the function appendtofolderh to append each one of the other data frames to the folderh.

Value

Returns an object of class folderh. Its elements are the data frames passed as arguments, and the attribute attr(, "keys") contains the character arguments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

is.folderh to test if an object is of class folderh. folder for a folder of data frames with no hierarchic relation between them. as.folder.folderh (or as.data.frame.folderh) to build an object of class folder (or a data frame) from an object of class folderh,

Examples

```
# First example: rose flowers
data(roseflowers)
df1 <- roseflowers$variety
df2 <- roseflowers$flower
fh1 <- folderh(df1, "rose", df2)
print(fh1)

# Second example
data(roseleaves)
roses <- roseleaves$rose
stems <- roseleaves$stem
leaves <- roseleaves$leaf
leaflets <- roseleaves$leaflet
fh2 <- folderh(roses, "rose", stems, "stem", leaves, "leaf", leaflets)
print(fh2)
```

---

foldermtg	*foldermtg*

Description

An object of S3 class "foldermtg" is built and returned by the function read.mtg.

Value

An object of this S3 class is a list of at least 5 data frames (see the Value section in `read.mtg`):
`classes`, `description`, `features`, `topology`, `coordinates`...

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

`read.mtg` `print.foldermtg` `mtgorder`

Examples

```
mtgfile1 <- system.file("extdata/plant1.mtg", package = "dad")
x1 <- read.mtg(mtgfile1)
print(x1)

mtgfile2 <- system.file("extdata/plant2.mtg", package = "dad")
x2 <- read.mtg(mtgfile2)
print(x2)
```

---

foldert	*Folder of data sets among time*

Description

Creates an object of class `"foldert"` (called foldert below), that is a list of data frames, each of
them corresponding to a time of observation. These data sets are on the same variables. They can
be on the same individuals or not.

Usage

```
foldert(x1, x2 = NULL, ..., times = NULL, cols.select = "intersect", rows.select = "")
```

Arguments

x1	data frame (can also be a tibble) or list of data frames.

- If x1 is a data frame, x2 must be provided.
- If x1 is a list of data frames, its elements are the datasets of the folder. In this case, there is no x2 argument.

x2	data frame. Must be provided if x1 is a data frame. Omitted if x1 is a list of data frames.

...	optional. One or several data frames when x1 is a data frame. These supplementary data frames are added to the list of data frames constituting the returned foldert.
times	Vector of the "times" of observations. It can be either numeric, or an ordered factor or an object of class "Date", "POSIXlt" or "POSIXct". If omitted, it is 1:N where N is the number of data frame arguments (if x1 is a data frame) or the length of x1 (if it is a list).

So there is an order relationship between these times.

cols.select   string or character vector. Gives the method used to choose the column names of the data frames of the foldert. This argument can be:

"intersect" (default) the column names of the data frames in the foldert are the intersection of the column names of all the data frames given as arguments.

"union" the column names of the data frames in the foldert are the union of the column names of all the data frames given as arguments. When necessary, the rows of the returned data frames are completed by NA.

If cols.select is a character vector, it gives the column names selected in the data frames given as arguments. The corresponding columns constitute the columns of the elements of the returned foldert. Notice that when a column name is not present in all data frames (given as arguments), the data are completed by NA.

rows.select   string. Gives the method used to choose the row names of the data frames of the foldert. This argument can be:

"" (default) the data frames of the foldert have the same rows as those which were passed as arguments.

"intersect" the row names of the data frames in the foldert are the intersection of the row names of all the data frames given as arguments.

"union" the row names of the data frames in the foldert are the union of the row names of all the data frames given as arguments. When necessary, the columns of the data frames returned are completed by NA.

Details

The class "foldert" has an attribute attr(,"times") (the times argument, when provided) and a logical attributes attr(,"same.rows").

The data frames in the returned foldert all have the same column names. That means that the same variables are observed in every data sets.

If the rows.select argument is "union" or "intersect", the elements of the returned foldert have the same rows. That means that the same individuals are present in every data sets. This allows to consider the evolution of each individual among time.

If rows.select is "", every rows of this foldert are different, and the row names are made unique by adding the name of the data frame to the row names. In this case, The individuals of the data sets are assumed to be all different. Or, at least, the user does not mind if they are the same or not.

Value

Returns an object of class `"foldert"`, that is a list of data frames. The elements of this list are ordered according to time.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

`is.foldert` to test if an object is of class `foldert`. `as.foldert.data.frame`: build an object of class `foldert` from a data frame. `as.foldert.array`: build an object of class `foldert` from a $3d$-array.

Examples

```
x <- data.frame(xyz = rep(c("A", "B", "C"), each = 2),
                xy = letters[1:6],
                x1 = rnorm(6),
                x2 = rnorm(6, 2, 1),
                row.names = paste0("i", 1:6),
                stringsAsFactors = TRUE)
y <- data.frame(xyz = c("A", "A", "B", "C"),
                xy = c("a", "b", "a", "c"),
                y1 = rnorm(4, 4, 2),
                row.names = c(paste0("i", c(1, 2, 4, 6))),
                stringsAsFactors = TRUE)
z <- data.frame(xyz = c("A", "B", "C"),
                z1 = rnorm(3),
                row.names = c("i1", "i2", "i5"),
                stringsAsFactors = TRUE)

# Columns selected by the user
ftc. <- foldert(x, y, z, cols.select = c("xyz", "x1", "y1", "z1"))
print(ftc.)

# cols.select = "union": all the variables (columns) of each data frame are kept
ftcun <- foldert(x, y, z, cols.select = "union")
print(ftcun)

# cols.select = "intersect": only variables common to all data frames
ftcint <- foldert(x, y, z, cols.select = "intersect")
print(ftcint)

# rows.select = "": the rows of the data frames are unchanged
# and the rownames are made unique
ftr. <- foldert(x, y, z, rows.select = "")
print(ftr.)

# rows.select = "union": all the individuals (rows) of each data frame are kept
ftrun <- foldert(x, y, z, rows.select = "union")
print(ftrun)
```

```
# rows.select = "intersect": only individuals common to all data frames
ftrint <- foldert(x, y, z, rows.select = "intersect")
print(ftrint)

# Define the times (times argument)
ftimes <- foldert(x, y, z, times = as.Date(c("2018-03-01", "2018-04-01", "2018-05-01")))
print(ftimes)
```

---

fpcad   *Functional PCA of probability densities*

Description

Performs functional principal component analysis of probability densities in order to describe a data folder, consisting of T groups of individuals on which are observed p variables. It returns an object of class fpcad.

Usage

```
fpcad(xf, group.name = "group", gaussiand = TRUE, windowh = NULL, normed = TRUE,
    centered = TRUE, data.centered = FALSE, data.scaled = FALSE,
    common.variance = FALSE, nb.factors = 3, nb.values = 10, sub.title = "",
    plot.eigen = TRUE, plot.score = FALSE, nscore = 1:3,
    filename = NULL)
```

Arguments

xf   object of class "[folder](#)" or data.frame.

- If it is an object of class "folder", its elements are data frames with p numeric columns. If there are non numeric columns, there is an error. The tth element ($t = 1, \ldots, T$) matches with the tth group.

- If it is a data frame, the column with name given by the group.name argument is a factor giving the groups. The other columns are all numeric; otherwise, there is an error.

group.name   string.

- If xf is an object of class "[folder](#)", name of the grouping variable in the returned results. The default is groupname = "group".

- If xf is a data frame, group.name is the name of the column of xf containing the groups.

gaussiand   logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method.

windowh   either a list of T bandwidths (one per density associated to a group), or a strictly positive number. If windowh = NULL (default), the bandwidths are automatically computed. See Details.

normed	logical. If TRUE (default), the densities are normed before computing the distances.
centered	logical. If TRUE (default), the densities are centered.
data.centered	logical. If TRUE (default is FALSE), the data of each group are centered.
data.scaled	logical. If TRUE (default is FALSE), the data of each group are centered (even if data.centered = FALSE) and scaled.
common.variance	
	logical. If TRUE (default is FALSE), a common covariance matrix (or correlation matrix if data.scaled = TRUE), computed on the whole data, is used. If FALSE (default), a covariance (or correlation) matrix per group is used.
nb.factors	numeric. Number of returned principal scores (default nb.factors = 3).
	Warning: The plot.fpcad and interpret.fpcad functions cannot take into account more than nb.factors principal factors.
nb.values	numerical. Number of returned eigenvalues (default nb.values = 10).
sub.title	string. If provided, the subtitle for the graphs.
plot.eigen	logical. If TRUE (default), the barplot of the eigenvalues is plotted.
plot.score	logical. If TRUE, the graphs of principal scores are plotted. A new graphic device is opened for each pair of principal scores defined by nscore argument.
nscore	numeric vector. If plot.score = TRUE, the numbers of the principal scores which are plotted. By default it is equal to nscore = 1:3. Its components cannot be greater than nb.factors.
filename	string. Name of the file in which the results are saved. By default (filename = NULL) the results are not saved.

Details

The T probability densities f_t corresponding to the T groups of individuals are either parametrically estimated (gaussiand = TRUE) or estimated using the Gaussian kernel method (gaussiand = FALSE). In the latter case, the windowh argument provides the list of the bandwidths to use. Notice that in the multivariate case ($p>1$) the bandwidths are positive-definite matrices.

If windowh is a numerical value, the matrix bandwidth is of the form hS, where S is either the square root of the covariance matrix ($p>1$) or the standard deviation of the estimated density.

If windowh = NULL (default), h in the above formula is computed using the bandwidth.parameter function.

Value

Returns an object of class fpcad, that is a list including:

inertia	data frame of the eigenvalues and percentages of inertia.
contributions	data frame of the contributions to the first nb.factors principal components.
qualities	data frame of the qualities on the first nb.factors principal factors.
scores	data frame of the first nb.factors principal scores.
norm	vector of the L^2 norms of the densities.

means	list of the means.
variances	list of the covariance matrices.
correlations	list of the correlation matrices.
skewness	list of the skewness coefficients.
kurtosis	list of the kurtosis coefficients.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (1998). Analyse en composantes principales de distributions gaussiennes multidimensionnelles. Revue de Statistique Appliqu?e, XLVI (2), 5-20.

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Delicado, P. (2011). Dimensionality reduction when data are density functions. Computational Statistics & Data Analysis, 55, 401-420.

Yousfi, S., Boumaza, R., Aissani, D., Adjabi, S. (2014). Optimal bandwith matrices in functional principal component analysis of density functions. Journal of Statistical Computation and Simulation, 85 (11), 2315-2330.

See Also

print.fpcad, plot.fpcad, interpret.fpcad, bandwidth.parameter

Examples

```
data(roses)
# Case of a normed non-centred PCA of Gaussian densities (on 3 architectural
# characteristics of roses: shape (Sha), foliage density (Den) and symmetry (Sym))
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])
result3 <- fpcad(rosesf, group.name = "rose")
print(result3)
plot(result3)

# Applied to a data frame:
result3df <- fpcad(roses[,c("Sha","Den","Sym","rose")], group.name = "rose")
print(result3df)
plot(result3df)

# Flower colors of the roses
scores <- result3$scores
scores <- data.frame(scores, color = scores$rose, stringsAsFactors = TRUE)
colours <- scores$rose
colours <- factor(c(A = "yellow", B = "yellow", C = "pink", D = "yellow", E = "red",
                    F = "yellow", G = "pink", H = "pink", I = "yellow", J = "yellow"))
levels(scores$color) <- c(A = "yellow", B = "yellow", C = "pink", D = "yellow", E = "red",
```

```
                              F = "yellow", G = "pink", H = "pink", I = "yellow", J = "yellow")
# Scores according to the first two principal components, per color
plot(result3, nscore = 1:2, color = colours)
```

---

fpcat   *Functional PCA of probability densities among time*

Description

Performs functional principal component analysis of probability densities in order to describe a data "foldert", consisting of individuals on which are observed p variables on T times. It returns an object of class fpcat.

Usage

```
fpcat(xf, group.name="time", method = 1, ind = 1, nvar = NULL, gaussiand = TRUE,
    windowh = NULL, normed=TRUE, centered=TRUE, data.centered = FALSE,
    data.scaled = FALSE, common.variance = FALSE, nb.factors = 3, nb.values = 10,
    sub.title = "", plot.eigen = TRUE, plot.score = FALSE, nscore = 1:3,
    filename = NULL)
```

Arguments

xf   object of class "foldert" or data.frame.

- An object of class "foldert" is a list of data frames with the same column names, each of them corresponding to a time of observation. Its elements are data frames with p numeric columns. If there are non numeric columns, there is an error. The tth element ($t = 1, \ldots, T$) matches with the tth time of observation.
- If it is a data frame:
    - If method=1: the column with name given by the group.name argument is a factor giving the groups. The other columns are all numeric; otherwise, there is an error.
    - If method=2: the column named after the ind argument contains the identifiers of the measured objects, and the observations are organized as follows:
      Given timecol the number of the column named by the group.name argument,
      the observations corresponding to the 1st time are on columns timecol : (timecol + nvar - 1)
      the observations corresponding to the 2nd time are on columns (timecol + nvar) : (timecol + 2 * nvar - 1)
      and so on.

group.name   string or numeric.

- If xf is an object of class "foldert", string. Name of the grouping variable, that is the observation times. The default is groupname = "time".

- If xf is a data frame, string or numeric, as the ind argument of `as.foldert.data.frame`.
  - If method = 1, timecol is the name or the number of the column of x containing the times of observation, or the number of this column. x[, timecol] must be of class "numeric", "ordered", "Date", "POSIXlt" or "POSIXct", otherwise, there is an error.
  - If method=2, timecol is the name or the number of the first column corresponding to the first observation. If there are duplicated column names and several columns are named by timecol, the first one is considered.

method
:   if xf is a data frame, 1 or 2. Omitted if xf is an object of class "foldert".

    If xf is a data frame, method indicates the layout of this data frame and, therefore, the method used to extract the data and build the foldert.

    - If method = 1, there is a column containing the identifiers of the measured objects and a column containing the times. The other columns contain the observations.
    - If method = 2, there is a column containing the identifiers of the measured objects, and the observations are organized as follows:
      - the observations corresponding to the 1st time are on columns timecol : (timecol + nvar - 1)
      - the observations corresponding to the 2nd time are on columns (timecol + nvar) : (timecol + 2 * nvar - 1)
      - and so on.

ind
:   if xf is a data frame, string or numeric. Omitted if xf is an object of class "foldert".

    The name of the column of x containing the indentifiers of the measured objects, or the number of this column. See the ind argument of `as.foldert.data.frame`.

nvar
:   if xf is a data frame and mathod=2, string or numeric. Omitted if xf is an object of class "foldert" or if method=1.

    The number of variable measured at each observation time. See the ind argument of `as.foldert.data.frame`.

All other arguments are the same as for `fpcad`.

gaussiand
:   logical. If TRUE (default), the probability densities are supposed Gaussian. If FALSE, densities are estimated using the Gaussian kernel method (as `fpcad`).

windowh
:   either a list of T bandwidths (one per density associated to a group), or a strictly positive number. If windowh = NULL (default), the bandwidths are automatically computed (as `fpcad`). See Details.

normed
:   logical. If TRUE (default), the densities are normed before computing the distances (as `fpcad`).

centered
:   logical. If TRUE (default), the densities are centered (as `fpcad`).

data.centered
:   logical. If TRUE (default is FALSE), the data of each group are centered (as `fpcad`).

data.scaled
:   logical. If TRUE (default is FALSE), the data of each group are centered (even if data.centered = FALSE) and scaled (as `fpcad`).

common.variance

        logical. If TRUE (default is FALSE), a common covariance matrix (or correlation matrix if data.scaled = TRUE), computed on the whole data, is used. If FALSE (default), a covariance (or correlation) matrix per group is used (as [fpcad](#)).

nb.factors   numeric. Number of returned principal scores (default nb.factors = 3) (as [fpcad](#)).

        Warning: The [plot.fpcad](#) and [interpret.fpcad](#) functions cannot take into account more than nb.factors principal factors (as [fpcad](#)).

nb.values   numerical. Number of returned eigenvalues (default nb.values = 10) (as [fpcad](#)).

sub.title   string. Subtitle for the graphs (default NULL) (as [fpcad](#)).

plot.eigen   logical. If TRUE (default), the barplot of the eigenvalues is plotted (as [fpcad](#)).

plot.score   logical. If TRUE, the graphs of principal scores are plotted. A new graphic device is opened for each pair of principal scores defined by nscore argument (as [fpcad](#)).

nscore   numeric vector. If plot.score = TRUE, the numbers of the principal scores which are plotted. By default it is equal to nscore = 1:3. Its components cannot be greater than nb.factors (as [fpcad](#)).

filename   string. Name of the file in which the results are saved. By default (filename = NULL) the results are not saved (as [fpcad](#)).

Details

The T probability densities f_t corresponding to the T times of observation are either parametrically estimated or estimated using the Gaussian kernel method (see [fpcad](#) for the use of the arguments indicating the method used to estimate these densities).

Value

Returns an object of class fpcat, that is a list including:

times	vector of the times of observation.
inertia	data frame of the eigenvalues and percentages of inertia.
contributions	data frame of the contributions to the first nb.factors principal components.
qualities	data frame of the qualities on the first nb.factors principal factors.
scores	data frame of the first nb.factors principal scores.
norm	vector of the L^2 norms of the densities.
means	list of the means.
variances	list of the covariance matrices.
correlations	list of the correlation matrices.
skewness	list of the skewness coefficients.
kurtosis	list of the kurtosis coefficients.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (1998). Analyse en composantes principales de distributions gaussiennes multidimensionnelles. Revue de Statistique Appliqu?e, XLVI (2), 5-20.

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Delicado, P. (2011). Dimensionality reduction when data are density functions. Computational Statistics & Data Analysis, 55, 401-420.

Yousfi, S., Boumaza, R., Aissani, D., Adjabi, S. (2014). Optimal bandwith matrices in functional principal component analysis of density functions. Journal of Statistical Computation and Simulation, 85 (11), 2315-2330.

See Also

print.fpcat, plot.fpcat, bandwidth.parameter

Examples

```
times <- as.Date(c("2017-03-01", "2017-04-01", "2017-05-01", "2017-06-01"))
x1 <- data.frame(z1=rnorm(6,1,5), z2=rnorm(6,3,3))
x2 <- data.frame(z1=rnorm(6,4,6), z2=rnorm(6,5,2))
x3 <- data.frame(z1=rnorm(6,7,2), z2=rnorm(6,8,4))
x4 <- data.frame(z1=rnorm(6,9,3), z2=rnorm(6,10,2))
ft <- foldert(x1, x2, x3, x4, times = times, rows.select="intersect")
print(ft)
result <- fpcat(ft)
print(result)
plot(result)
```

---

getcol.folder   *Select columns in all elements of a folder*

Description

Select columns in all data frames of a folder.

Usage

```
getcol.folder(object, name)
```

Arguments

object	object of class folder that is a list of data frames with the same column names.
name	character vector. The names of the columns to be selected in each data frame of the folder.

Value

A folder with the same number of elements as object. Its kth element is a data frame, and its columns are the columns of object[[k]] given by name.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder: object of class folder.

rmcol.folder: remove columns in all elements of a folder.

getrow.folder: select rows in all elements of a folder.

rmrow.folder: remove rows in all elements of a folder.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
getcol.folder(iris.fold, "Sepal.Length")
getcol.folder(iris.fold, c("Petal.Length", "Petal.Width"))
```

---

getcol.foldert   *Select columns in all elements of a foldert*

Description

Select columns in all data frames of a foldert.

Usage

```
getcol.foldert(object, name)
```

Arguments

object	object of class foldert that is a list of data frames with the same column names, each of them corresponding to a time of observation.
name	character vector. The names of the columns to be selected in each data frame of the foldert.

Value

A foldert with the same number of elements as object. Its kth element is a data frame, and its columns are the columns of object[[k]] given by name.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](): object of class `foldert`.

[rmcol.foldert](): remove columns in all elements of a foldert.

[getrow.foldert](): select rows in all elements of a foldert.

[rmrow.foldert](): remove rows in all elements of a foldert.

Examples

```
data(floribundity)

ft0 <- foldert(floribundity, cols.select = "union")
getcol.foldert(ft0, c("rose", "variety"))
```

---

getrow.folder   *Select rows in all elements of a folder*

Description

Select rows in all data frames of a folder.

Usage

```
getrow.folder(object, name)
```

Arguments

object	object of class [folder]() that is a list of data frames with the same column names.
name	character vector. The names of the rows to be selected in each data frame of the folder.

Value

A folder with the same number of elements as `object`. Its kth element is a data frame, and its rows are the rows of `object[[k]]` given by `name`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder: object of class `folder`.

rmrow.folder: remove rows in all elements of a folder.

getcol.folder: select rows in all elements of a folder.

rmcol.folder: remove rows in all elements of a folder.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
getrow.folder(iris.fold, c(1:5, 51:55, 101:105))
```

---

getrow.foldert   *Select rows in all elements of a foldert*

Description

Select rows in all data frames of a foldert.

Usage

```
getrow.foldert(object, name)
```

Arguments

object   object of class foldert that is a list of data frames with the same column names, each of them corresponding to a time of observation.

name   character vector. The names of the rows to be selected in each data frame of the foldert.

Value

A foldert with the same number of elements as `object`. Its kth element is a data frame, and its rows are the rows of `object[[k]]` given by `name`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

foldert: object of class `foldert`.

rmrow.foldert: remove rows in all elements of a foldert.

getcol.foldert: select columns in all elements of a foldert.

rmcol.foldert: remove columns in all elements of a foldert.

Examples

```
data(floribundity)

ft0 <- foldert(floribundity, cols.select = "union", rows.select = "union")
getrow.foldert(ft0, c("16", "51"))
```

---

hclustdd	*Hierarchic cluster analysis of discrete probability distributions*

Description

Performs functional hierarchic cluster analysis of discrete probability distributions. It returns an object of class [hclustdd](). It applies [hclust]() to the distance matrix between the T distributions.

Usage

```
hclustdd(xf, group.name = "group", distance = c("l1", "l2", "chisqsym", "hellinger",
              "jeffreys", "jensen", "lp"),
              sub.title = "", filename = NULL,
              method.hclust = "complete")
```

Arguments

xf   object of class [folder](), or list of arrays (or tables).

- If it is a folder, its elements are data frames with q columns (considered as factors). The tth element ($t = 1, \ldots, T$) matches with the tth group.
- If it is a data frame, the columns with name given by the group.name argument is a factor giving the groups. The other columns are all considered as factors.
- If it is a list of arrays (or tables), the tth element ($t = 1, \ldots, T$) is the table of the joint frequency distribution of q variables within the tth group. The frequency distribution is expressed with relative or absolute frequencies. These arrays have the same shape.

  Each array (or table) xf[[i]] has:
  - the same dimension(s). If $q = 1$ (univariate), dim(xf[[i]]) is an integer. If $q > 1$ (multivariate), dim(xf[[i]]) is an integer vector of length q.
  - the same dimension names dimnames(xf[[i]]) (is non NULL). These dimnames are the names of the variables.

  The elements of the arrays are non-negative numbers (if they are not, there is an error).

group.name string. Name of the grouping variable. Default: group.name = "group".

distance   The distance or divergence used to compute the distance matrix between the discrete distributions (see Details). It can be:

- "l1" (default) the L^p distance with $p = 1$

- "l2" the L^p distance with $p = 2$
- "chisqsym" the symmetric Chi-squared distance
- "hellinger" the Hellinger metric (Matusita distance)
- "jeffreys" Jeffreys distance (symmetrised Kullback-Leibler divergence)
- "jensen" the Jensen-Shannon distance
- "lp" the L^p distance with p given by the argument p of the function.

sub.title	string. If provided, the subtitle for the graphs.
filename	string. Name of the file in which the results are saved. By default (filename = NULL) the results are not saved.
method.hclust	the agglomeration method to be used for the clustering. See the method argument of the [hclust](#) function.

Details

In order to compute the distances/dissimilarities between the groups, the T probability distributions f_t corresponding to the T groups of individuals are estimated from observations. Then the distances/dissimilarities between the estimated distributions are computed, using the distance or divergence defined by the distance argument:

If the distance is "l1", "l2" or "lp", the distances are computed by the function [matddlppar](#). Otherwise, it can be computed by [matddchisqsympar](#) ("chisqsym"), [matddhellingerpar](#) ("hellinger"), [matddjeffreyspar](#) ("jeffreys") or [matddjensenpar](#) ("jensen").

Value

Returns an object of class hclustdd, that is a list including:

distances	matrix of the L^2-distances between the estimated densities.
clust	an object of class [hclust](#).

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[hclustdd](#)

Examples

```
# Example 1 with a folder (10 groups) of 3 factors
# obtained by converting numeric variables
data(roses)
xr = roses[,c("Sha", "Den", "Sym", "rose")]
xr = cut(xr, breaks = list(c(0, 5, 7, 10), c(0, 4, 6, 10), c(0, 6, 8, 10)))
xf = as.folder(xr, groups = "rose")
af = hclustdd(xf)
print(af)
print(af, dist.print = TRUE)
```

```
plot(af)
plot(af, hang = -1)

# Example 2 with a data frame obtained by converting numeric variables
ar = hclustdd(xr, group.name = "rose")
print(ar)
print(ar, dist.print = TRUE)
plot(ar)
plot(ar, hang = -1)

# Example 3 with a list of 7 arrays
data(dspg)
xl = dspg
hclustdd(xl)
```

---

hellinger *Hellinger distance between Gaussian densities*

Description

Hellinger distance between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities (see Details).

Usage

```
hellinger(x1, x2, check = FALSE)
```

Arguments

x1	a matrix or data frame of n_1 rows (observations) and p columns (variables) (can also be a tibble) or a vector of length n_1.
x2	matrix or data frame (or tibble) of n_2 rows and p columns or vector of length n_2.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

The Hellinger distance between the two Gaussian densities is computed by using the `hellingerpar` function and the density parameters estimated from samples.

Value

Returns the $Hellinger$ distance between the two probability densities.

Be careful! If check = FALSE and one smoothing bandwidth matrix is degenerate, the result returned can not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

McLachlan, G.J. (1992). Discriminant analysis and statistical pattern recognition. John Wiley & Sons, New York .

See Also

hellingerpar: Hellinger distance between Gaussian densities, given their parameters.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
hellinger(x1, x2)
```

---

hellingerpar	*Hellinger distance between Gaussian densities given their parameters*

Description

Hellinger distance between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate) (see Details).

Usage

```
hellingerpar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1	p-length numeric vector: the mean of the first Gaussian density.
var1	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the first Gaussian density.
mean2	p-length numeric vector: the mean of the second Gaussian density.
var2	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the second Gaussian density.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

The mean vectors ($m1$ and $m2$) and variance matrices ($v1$ and $v2$) given as arguments (mean1, mean2, var1 and var2) are used to compute the Hellinger distance between the two Gaussian densities, equal to:

$$(2(1 - 2^{p/2}det(v1v2)^{1/4}det(v1+v2)^{-1/2}exp((-1/4)t(m1-m2)(v1+v2)^{-1}(m1-m2))))^{1/2}$$

If $p = 1$ the means and variances are numbers, the formula is the same ignoring the following operators: t (transpose of a matrix or vector) and det (determinant of a square matrix).

Value

The Hellinger distance between two Gaussian densities.

Be careful! If check = FALSE and one covariance matrix is degenerated (multivariate case) or one variance is zero (univariate case), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

McLachlan, G.J. (1992). Discriminant analysis and statistical pattern recognition. John Wiley & Sons, New York .

See Also

[hellinger](): Hellinger distance between Gaussian densities estimated from samples.

Examples

```
m1 <- c(1,1)
v1 <- matrix(c(4,1,1,9),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(1,0,0,1),ncol = 2)
hellingerpar(m1,v1,m2,v2)
```

---

interpret | *Scores of* fmdsd*,* dstatis*,* fpcad*, or* fpcat *vs. moments, or scores of* mdsdd *vs. marginal distributions or association measures*

Description

This generic function provides a tool for the interpretation of the results of fmdsd, dstatis, fpcad, fpcat or mdsdd function.

Usage

```
interpret(x, nscore = 1:3, ...)
```

Arguments

x   object of class fmdsd, dstatis, fpcad, fpcat or mdsdd.

> - fmdsd: see `interpret.fmdsd`
> - dstatis: see `interpret.dstatis`
> - fpcad: see `interpret.fpcad`
> - fpcat: see `interpret.fpcat`
> - mdsdd: see `interpret.mdsdd`

nscore   numeric vector. Selects the columns of the data frame x$scores to be interpreted.

> Warning: Its components cannot be greater than the nb.factors argument in the call of the `fpcad` or `fpcat` function.

...   Arguments to be passed to the methods, such as moment (for `interpret.fmdsd`, `interpret.dstatis`, `interpret.fpcad` and `interpret.fpcat`), or mma (for `interpret.mdsdd`).

Value

Returns a list including:

pearson   matrix of Pearson correlations between selected scores and moments, probabilities or associations.

spearman   matrix of Spearman correlations between selected scores and moments, probabilities or associations.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

interpret.fmdsd; interpret.dstatis; interpret.fpcad; interpret.fpcat; interpret.mdsdd.

---

interpret.dstatis   *Scores of the* dstatis *function vs. moments of the densities*

Description

Applies to an object of class "dstatis", plots the principal scores vs. the moments of the densities (means, standard deviations, variances, correlations, skewness and kurtosis coefficients), and computes the correlations between these scores and moments.

Usage

```
## S3 method for class 'dstatis'
interpret(x, nscore = 1, moment=c("mean", "sd", "var", "cov", "cor",
    "skewness", "kurtosis"), ...)
```

Arguments

x	object of class "dstatis" (returned by the dstatis.inter function).
nscore	numeric. Selects the column of the data frame x$scores consisting of a score vector.
	Note that since dad-4, nscore can only be a single value (in earlier versions, it could be a vector of length > 1).
	Warning: nscore cannot be greater than the nb.factors argument in the call of the dstatis.inter function.
moment	characters string. Selects the moments to cross with scores:

- "mean" (means)
- "sd" (standard deviations)
- "cov" (covariances)
- "cor" (correlation coefficients)
- "skewness" (skewness coefficients)
- "kurtosis" (kurtosis coefficients)

...	Arguments to be passed to methods.

Details

A graphics device can contain up to 9 graphs. If there are too many (more than 36) graphs for each score, one can display the graphs in a multipage PDF file.

The number of principal scores to be interpreted cannot be greater than nb.factors of the data frame x$scores returned by the function dstatis.inter.

Value

Returns a list including:

pearson	matrix of Pearson correlations between selected scores and moments.
spearman	matrix of Spearman correlations between selected scores and moments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Lavit, C., Escoufier, Y., Sabatier, R., Traissac, P. (1994). The ACT (STATIS method). Computational Statistics & Data Analysis, 18 (1994), 97-119.

See Also

dstatis.inter; plot.dstatis.

Examples

```
data(roses)
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])

# Dual STATIS on the covariance matrices
## Not run:
result <- dstatis.inter(rosesf, group.name = "rose")
interpret(result)
interpret(result, moment = "var")
interpret(result, moment = "cor")
interpret(result, nscore = 2)

## End(Not run)
```

---

interpret.fmdsd   *Scores of the* fmdsd *function vs. moments of the densities*

Description

Applies to an object of class "fmdsd", plots the scores vs. the moments of the densities (means, standard deviations, variances, correlations, skewness and kurtosis coefficients), and computes the correlations between these scores and moments.

Usage

```
## S3 method for class 'fmdsd'
interpret(x, nscore = 1, moment=c("mean", "sd", "var", "cov", "cor",
    "skewness", "kurtosis"), ...)
```

Arguments

x   object of class "fmdsd" (returned by the fmdsd function).

nscore   numeric. Selects the column of the data frame x$scores consisting of a score
                vector.

                Note that since dad-4, nscore can only be a single value (in earlier versions, it
                could be a vector of length > 1).

                Warning: nscore cannot be greater than the nb.factors argument in the call
                of the [fmdsd](#) function.

moment   character string. Selects the moments to cross with scores:

                - "mean" (means, which is the default value)
                - "sd" (standard deviations)
                - "cov" (covariances)
                - "cor" (correlation coefficients)
                - "skewness" (skewness coefficients)
                - "kurtosis" (kurtosis coefficients)

...   Arguments to be passed to methods.

Details

A graphics device can contain up to 9 graphs. If there are too many (more than 36) graphs for each
score, one can display the graphs in a multipage PDF file.

The number of principal scores to be interpreted cannot be greater than nb.factors of the data
frame x$scores returned by the function [fmdsd](#).

Value

Returns a list including:

pearson   matrix of Pearson correlations between selected scores and moments.

spearman   matrix of Spearman correlations between selected scores and moments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analy-
sis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16),
3321-3339.

Delicado, P. (2011). Dimensionality reduction when data are density functions. Computational
Statistics & Data Analysis, 55, 401-420.

See Also

[fmdsd](#); [plot.fmdsd](#).

Examples

```
data(roses)
x <- roses[,c("Sha","Den","Sym","rose")]
rosesfold <- as.folder(x)
result1 <- fmdsd(rosesfold)
interpret(result1)
## Not run:
interpret(result1, moment = "var")

## End(Not run)
interpret(result1, nscore = 2)
```

---

interpret.fpcad   *Scores of the* fpcad *function vs. moments of the densities*

Description

Applies to an object of class "fpcad", plots the principal scores vs. the moments of the densities (means, standard deviations, variances, correlations, skewness and kurtosis coefficients), and computes the correlations between these scores and moments.

Usage

```
## S3 method for class 'fpcad'
interpret(x, nscore = 1, moment=c("mean", "sd", "var", "cov", "cor",
    "skewness", "kurtosis"), ...)
```

Arguments

x	object of class "fpcad" (returned by the fpcad function).
nscore	numeric. Selects the column of the data frame x$scores consisting of a score vector.
	Note that since dad-4, nscore can only be a single value (in earlier versions, it could be a vector of length > 1).
	Warning: nscore cannot be greater than the nb.factors argument in the call of the fpcad function.
moment	characters string. Selects the moments to cross with scores:
	• "mean" (means)
	• "sd" (standard deviations)
	• "cov" (covariances)
	• "cor" (correlation coefficients)
	• "skewness" (skewness coefficients)
	• "kurtosis" (kurtosis coefficients)
...	Arguments to be passed to methods.

Details

A graphics device can contain up to 9 graphs. If there are too many (more than 36) graphs for each score, one can display the graphs in a multipage PDF file.

The number of principal scores to be interpreted cannot be greater than nb.factors of the data frame x$scores returned by the function fpcad.

Value

Returns a list including:

pearson   matrix of Pearson correlations between selected scores and moments.

spearman   matrix of Spearman correlations between selected scores and moments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcad; plot.fpcad.

Examples

```
data(roses)
rosefold <- as.folder(roses[,c("Sha","Den","Sym","rose")])
result1 <- fpcad(rosefold)
interpret(result1)
## Not run:
interpret(result1, moment = "var")

## End(Not run)
interpret(result1, moment = "cor")
interpret(result1, nscore = 2)
```

---

interpret.fpcat   *Scores of the* "fpcat" *function vs. moments of the densities*

Description

This function applies to an object of class "fpcat" and does the same as for an object of class
"fpcad": it plots the principal scores vs. the moments of the densities (means, standard deviations,
variances, correlations, skewness and kurtosis coefficients), and computes the correlations between
these scores and moments.

Usage

```
## S3 method for class 'fpcat'
interpret(x, nscore = 1, moment=c("mean", "sd", "var", "cov", "cor",
    "skewness", "kurtosis"), ...)
```

Arguments

x	object of class "fpcat" (returned by the fpcat function).
nscore	numeric. Selects the column of the data frame x$scores consisting of a score vector.
	Note that since dad-4, nscore can only be a single value (in earlier versions, it could be a vector of length > 1).
	Warning: nscore cannot be greater than the nb.factors argument in the call of the fpcat function.
moment	characters string. Selects the moments to cross with scores:
	• "mean" (means)
	• "sd" (standard deviations)
	• "cov" (covariances)
	• "cor" (correlation coefficients)
	• "skewness" (skewness coefficients)
	• "kurtosis" (kurtosis coefficients)
...	Arguments to be passed to methods.

Details

A graphics device can contain up to 9 graphs. If there are too many (more than 36) graphs for each
score, one can display the graphs in a multipage PDF file.

The number of principal scores to be interpreted cannot be greater than nb.factors of the data
frame x$scores returned by the function fpcat.

Value

Returns a list including:

pearson	matrix of Pearson correlations between selected scores and moments.
spearman	matrix of Spearman correlations between selected scores and moments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcat; plot.fpcat.

Examples

```
# Alsacian castles with their building year
data(castles)
castyear <- foldert(lapply(castles, "[", 1:4))
fpcayear <- fpcat(castyear, group.name = "year")
interpret(fpcayear)
## Not run:
interpret(fpcayear, moment="var")

## End(Not run)
```

---

interpret.mdsdd   *Scores of the* mdsdd *function vs. marginal probability distributions or association measures*

Description

Applies to an object of class "mdsdd", plots the scores vs. the marginal probability distributions or pairwise association measures of the discrete variables, and computes the correlations between these scores and probabilities or association measures (see Details).

Usage

```
## S3 method for class 'mdsdd'
interpret(x, nscore = 1, mma = c("marg1", "marg2", "assoc"), ...)
```

Arguments

x	object of class "mdsdd" (returned by the mdsdd function).
nscore	numeric. Selects the column of the data frame x$scores consisting of a score vector.
	Note that since dad-4, nscore can only be a single value (in earlier versions, it could be a vector of length > 1).
	Warning: nscore cannot be greater than the nb.factors argument in the call of the mdsdd function.

mma   character. Indicates which measures will be considered:

- "marg1": the probability distribution of each variable.
- "marg2": the joint probability distribution of each pair of variables.
- "assoc": the pairwise association measures of the variables.

...   Arguments to be passed to methods.

Details

A graphics device can contain up to 9 graphs. If there are too many (more than 36) graphs for each
score, one can display the graphs in a multipage PDF file.

The number of principal scores to be interpreted cannot be greater than nb.factors of the data
frame x$scores returned by the function [mdsdd](#).

Value

Returns a list including:

pearson   matrix of Pearson correlations between selected scores and probabilities or as-
                             sociation measures.

spearman   matrix of Spearman correlations between selected scores and probabilities or
                             association measures.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

See Also

[mdsdd](#); [plot.mdsdd](#).

Examples

```
# INSEE (France): Diploma x Socio professional group, seven years.
data(dspg)
xlista = dspg
a <- mdsdd(xlista)
interpret(a)

# Example 3 with a list of 96 arrays (departments)
## Not run:
data(dspgd2015)
xd = dspgd2015
res = mdsdd(xd, group.name = "coded")
interpret(res)
plot(res, fontsize.points = 0.7)

# Each department is represented by its name
data(departments)
coor = merge(res$scores, departments, by = "coded")
dev.new()
```

```
plot(coor$PC.1, coor$PC.2, type ="n")
text(coor$PC.1, coor$PC.2, coor$named, cex = 0.5)

# Each department is represented by its region
dev.new()
plot(coor$PC.1, coor$PC.2, type ="n")
text(coor$PC.1, coor$PC.2, coor$coder, cex = 0.7)

## End(Not run)
```

---

is.discdd.misclass   *Class* discdd.misclass

Description

Tests if its argument is an object of class discdd.misclass (see Details of the function [discdd.misclass](#)).

Usage

```
is.discdd.misclass(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class discdd.misclass, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[discdd.misclass](#).

---

is.discdd.predict   *Class* discdd.predict

Description

Tests if its argument is an object of class discdd.predict (see Details of the function [discdd.predict](#)).

Usage

```
is.discdd.predict(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class discdd.predict, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[discdd.predict](#).

---

is.dstatis   *Class* dstatis

Description

Tests if its argument is an object of class dstatis (see Details of the function [dstatis.inter](#)).

Usage

```
is.dstatis(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class dstatis, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[dstatis.inter](#).

---

is.fdiscd.misclass   *Class* fdiscd.misclass

Description

Tests if its argument is an object of class fdiscd.misclass (see Details of the function [fdiscd.misclass](#)).

Usage

```
is.fdiscd.misclass(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class fdiscd.misclass, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[fdiscd.misclass](#).

---

is.fdiscd.predict   *Class* fdiscd.predict

Description

Tests if its argument is an object of class fdiscd.predict (see Details of the function fdiscd.predict)..

Usage

```
is.fdiscd.predict(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class fdiscd.predict, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

fdiscd.predict.

---

is.fhclustd   *Class* fhclustd

Description

Tests if its argument is an object of class fhclustd (see Details of the function fhclustd).

Usage

```
is.fhclustd(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class fhclustd, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[fhclustd.](#)

---

is.fmdsd *Class* fmdsd

Description

Tests if its argument is an object of class fmdsd (see Details of the function [fmdsd](#)).

Usage

```
is.fmdsd(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class fmdsd, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[fmdsd.](#)

---

is.folder   *Class* folder

Description

Tests if its argument is an object of class folder (see [folder](#)).

Usage

```
is.folder(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class folder, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#) to create an object of class folder.

---

is.folderh   *Class* folderh

Description

Tests if its argument is an object of class folderh (see [folderh](#)).

Usage

```
is.folderh(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class folderh, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folderh](#) to create an object of class folderh.

---

is.foldermtg	*Class* foldermtg

Description

Tests if its argument is an object of class foldermtg (see [read.mtg](#)).

Usage

```
is.foldermtg(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class foldermtg, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[read.mtg](#) to read a MTG file and create an object of class foldermtg.

---

is.foldert   *Class* foldert

Description

Tests if its argument is an object of class foldert (see [foldert](#)).

Usage

```
is.foldert(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class foldert, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](#) to create an object of class foldert.

---

is.fpcad   *Class* fpcad

Description

Tests if its argument is an object of class fpcad (see Details of the function [fpcad](#)).

Usage

```
is.fpcad(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class fpcad, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

fpcad.

---

is.mdsdd   *Class* mdsdd

Description

Tests if its argument is an object of class mdsdd (see Details of the function mdsdd).

Usage

```
is.mdsdd(x)
```

Arguments

x   object to be tested.

Value

TRUE if its argument is of class mdsdd, and FALSE otherwise.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

mdsdd.

---

jeffreys   *Jeffreys measure between Gaussian densities*

Description

Jeffreys measure (or symmetrised Kullback-Leibler divergence) between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities given samples (see Details).

Usage

```
jeffreys(x1, x2, check = FALSE)
```

Arguments

x1	a matrix or data frame of n_1 rows (observations) and p columns (variables) (can also be a tibble) or a vector of length n_1.
x2	matrix or data frame (or tibble) of n_2 rows and p columns or vector of length n_2.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

The Jeffreys measure between the two Gaussian densities is computed by using the jeffreyspar function and the density parameters estimated from samples.

Value

Returns the Jeffrey's measure between the two probability densities.

Be careful! If check = FALSE and one smoothing bandwidth matrix is degenerate, the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Thabane, L., Safiul Haq, M. (1999). On Bayesian selection of the best population using the Kullback-Leibler divergence measure. Statistica Neerlandica, 53(3): 342-360.

See Also

jeffreyspar: Jeffreys measure between Gaussian densities, given their parameters.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
jeffreys(x1, x2)
```

---

jeffreyspar	*Jeffreys measure between Gaussian densities given their parameters*

Description

Jeffreys measure (or symmetrised Kullback-Leibler divergence) between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if they are multivariate, means and variances if univariate) (see Details).

Usage

```
jeffreyspar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1	p-length numeric vector: the mean of the first Gaussian density.
var1	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the first Gaussian density.
mean2	p-length numeric vector: the mean of the second Gaussian density.
var2	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the second Gaussian density.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

Let $m1$ and $m2$ the mean vectors, $v1$ and $v2$ the covariance matrices, Jeffreys measure of the two Gaussian densities is equal to:

$$(1/2)t(m1 - m2)(v1^{-1} + v2^{-1})(m1 - m2) - (1/2)tr((v1 - v2)(v1^{-1} - v2^{-1}))$$

.

If $p = 1$ the means and variances are numbers, the formula is the same ignoring the following operators: t (transpose of a matrix or vector) and tr (trace of a square matrix).

Value

Jeffreys measure between two Gaussian densities.

Be careful! If check = FALSE and one covariance matrix is degenerated (multivariate case) or one variance is zero (univariate case), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

McLachlan, G.J. (1992). Discriminant analysis and statistical pattern recognition. John Wiley & Sons, New York .

Thabane, L., Safiul Haq, M. (1999). On Bayesian selection of the best population using the Kullback-Leibler divergence measure. Statistica Neerlandica, 53(3): 342-360.

See Also

jeffreys: Jeffreys measure of two parametrically estimated Gaussian densities, given samples.

Examples

```
m1 <- c(1,1)
v1 <- matrix(c(4,1,1,9),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(1,0,0,1),ncol = 2)
jeffreyspar(m1,v1,m2,v2)
```

---

kurtosis.folder   *Kurtosis coefficients of a folder of data sets*

Description

Computes the kurtosis coefficient by column of the elements of an object of class folder.

Usage

```
kurtosis.folder(x, na.rm = FALSE, type = 3)
```

Arguments

x	an object of class folder.
na.rm	logical. Should missing values be omitted from the calculations? (see kurtosis)
type	an integer between 1 and 3 (see kurtosis).

Details

It uses [kurtosis](#) to compute the mean by numeric column of each element of the folder. If some columns of the data frames are not numeric, there is a warning, and the means are computed on the numeric columns only.

Value

A list whose elements are the kurtosis coefficients by column of the elements of the folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#) to create an object is of class folder. [mean.folder](#), [var.folder](#), [cor.folder](#), [skewness.folder](#) for other statistics for folder objects.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
iris.kurtosis <- kurtosis.folder(iris.fold)
print(iris.kurtosis)

# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
roses.kurtosis <- kurtosis.folder(roses.fold)
print(roses.kurtosis)
```

---

l2d   *Lˆ2 inner product of probability densities*

Description

L^2 inner product of two multivariate ($p > 1$) or univariate ($p = 1$) probability densities, estimated from samples.

Usage

```
l2d(x1, x2, method = "gaussiand", check = FALSE, varw1 = NULL, varw2 = NULL)
```

Arguments

x1	a matrix or data frame of n_1 rows (observations) and p columns (variables) (can also be a tibble) or a vector of length n_1.
x2	matrix or data frame (or tibble) of n_2 rows and p columns or vector of length n_2.
method	string. It can be:

- "gaussiand" if the densities are considered to be Gaussian.
- "kern" if they are estimated using the Gaussian kernel method.

check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices (if method = "gaussiand") or smoothing bandwidth matrices (if method = "kern") are not degenerate, before computing the inner product.

Notice that if $p = 1$, it checks if the variances or smoothing parameters are not zero.

varw1, varw2	p x p symmetric matrices: the smoothing bandwidths for the estimation of the probability densities. If they are omitted, the smoothing bandwidths are computed using the normal reference rule matrix bandwidth (see details).

Details

- If method = "gaussiand", the mean vectors and the variance matrices ($v1$ and $v2$) of the two samples are computed, and they are used to compute the inner product using the [l2dpar](#) function.

- If method = "kern", the densities of both samples are estimated using the Gaussian kernel method. These estimations are then used to compute the inner product. if varw1 and varw2 arguments are omitted, the smoothing bandwidths are computed using the normal reference rule matrix bandwidth:

$$h_1 v_1^{1/2}$$

  where

$$h_1 = (4/(n_1(p+2)))^{1/(p+4)}$$

  for the first density. Idem for the second density after making the necessary changes.

Value

The L^2 inner product of the two probability densities.

Be careful! If check = FALSE and one smoothing bandwidth matrix is degenerate, the result returned can not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

Wand, M., Jones, M. (1995). Kernel smoothing. Chapman and Hall/CRC, London.

Yousfi, S., Boumaza R., Aissani, D., Adjabi, S. (2014). Optimal bandwith matrices in functional principal component analysis of density functions. Journal of Statistical Computational and Simulation, 85 (11), 2315-2330.

See Also

l2dpar for Gaussian densities whose parameters are given.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
l2d(x1, x2, method = "gaussiand")
l2d(x1, x2, method = "kern")
l2d(x1, x2, method = "kern", varw1 = v1, varw2 = v2)
```

---

l2dpar   *Lˆ2 inner product of Gaussian densities given their parameters*

Description

L^2 inner product of multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate).

Usage

```
l2dpar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1	p-length numeric vector: the mean of the first Gaussian density.
var1	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the first Gaussian density.
mean2	p-length numeric vector: the mean of the second Gaussian density.
var2	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the second Gaussian density.

| check | logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case). |

Details

Computes the inner product of two Gaussian densities, equal to:

$$(2\pi)^{-p/2} det(var1+var2)^{-1/2} exp(-(1/2)t(mean1-mean2)(var1+var2)^{-1}(mean1-mean2))$$

If $p = 1$ the means and variances are numbers, the formula is the same ignoring the following operators: t (transpose of a matrix or vector) and det (determinant of a square matrix).

Value

The L^2 inner product between two Gaussian densities.

Be careful! If check = FALSE and one covariance matrix is degenerated (multivariate case) or one variance is zero (univariate case), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

M. Wand and M. Jones (1995). Kernel Smoothing. Chapman and Hall, London.

See Also

l2d for parametrically estimated Gaussian densities or nonparametrically estimated densities, given samples;

Examples

```
m1 <- c(1,1)
v1 <- matrix(c(4,1,1,9),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(1,0,0,1),ncol = 2)
l2dpar(m1,v1,m2,v2)
```

---

matddchisqsym | *Matrix of distances between discrete probability densities given samples*

Description

Computes the matrix of the symmetric Chi-squared distances between several multivariate or univariate discrete probability distributions, estimated from samples.

Usage

```
matddchisqsym(x)
```

Arguments

x   object of class `"folder"` containing the data. Its elements are data frames (one data frame per distribution) whose columns are factors.

Value

Positive symmetric matrix whose order is equal to the number of data frames (or distributions), consisting of the pairwise symmetric chi-squared distances between the distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddchisqsym](#).

[matddchisqsympar](#) for discrete probability densities, given the probabilities on the same support.

Examples

```
# Example 1
x1 <- data.frame(x = factor(c("A", "A", "B", "B")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")))
xf <- folder(x1, x2, x3)
matddchisqsym(xf)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
```

```
                    y = factor(c("a", "a", "b", "a", "b")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")),
                    y = factor(c("a", "b", "a", "b", "a", "b")))
xf <- folder(x1, x2, x3)
matddchisqsym(xf)
```

---

matddchisqsympar   *Matrix of distances between discrete probability densities given the*
                                  probabilities on their common support

---

Description

Computes the matrix of the symmetric Chi-squared distances between several multivariate or uni-
variate discrete probability distributions on the same support (which can be a Cartesian product of
q sets), given the probabilities of the states (which are q-tuples) of the support.

Usage

```
matddchisqsympar(freq)
```

Arguments

freq   list of arrays. Their dim attribute is a vector with length q, its elements contain-
                     ing the numbers of levels of the *sets*. Each array contains the probabilities of
                     the discrete distribution on the same support.

Value

Positive symmetric matrix whose order is equal to the number of distributions, consisting of the
pairwise symmetric chi-squared distances between these distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddchisqsympar](#).

[matddchisqsym](#) for discrete probability densities which are estimated from the data.

---

matddhellinger	*Matrix of distances between discrete probability densities given samples*

Description

Computes the matrix of the Hellinger (or Matusita) distances between several multivariate or univariate discrete probability distributions, estimated from samples.

Usage

```
matddhellinger(x)
```

Arguments

x   object of class `"folder"` containing the data. Its elements are data frames (one data frame per distribution) whose columns are factors.

Value

Positive symmetric matrix whose order is equal to the number of data frames (or distributions), consisting of the pairwise Hellinger distances between the distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddhellinger.

matddhellingerpar for discrete probability densities, given the probabilities on the same support.

Examples

```
# Example 1
x1 <- data.frame(x = factor(c("A", "A", "B", "B")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
```

```
                    y = factor(c("a", "a", "b", "a", "b")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")),
                    y = factor(c("a", "b", "a", "b", "a", "b")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)
```

---

matddhellingerpar   *Matrix of distances between discrete probability densities given the*
                           probabilities on their common support

---

Description

Computes the matrix of the Hellinger (or Matusita) distances between several multivariate or univariate discrete probability distributions on the same support (which can be a Cartesian product of q sets), given the probabilities of the states (which are q-tuples) of the support.

Usage

```
matddhellingerpar(freq)
```

Arguments

freq   list of arrays. Their dim attribute is a vector with length q, its elements containing the numbers of levels of the *sets*. Each array contains the probabilities of the discrete distribution on the same support.

Value

Positive symmetric matrix whose order is equal to the number of distributions, consisting of the pairwise Hellinger distances between these distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddhellingerpar](#).

[matddhellinger](#) for discrete probability densities which are estimated from the data.

---

matddjeffreys   *Matrix of distances between discrete probability densities given samples*

Description

Computes the matrix of Jeffreys divergences between several multivariate or univariate discrete probability distributions, estimated from samples.

Usage

```
matddjeffreys(x)
```

Arguments

x   object of class `"folder"` containing the data. Its elements are data frames (one data frame per distribution) whose columns are factors.

Value

Positive symmetric matrix whose order is equal to the number of data frames (or distributions), consisting of the pairwise Jeffreys divergences between the distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Dezaz E. (2013). Encyclopedia of distances. Springer.

See Also

[ddjeffreys](#).

[matddjeffreyspar](#) for discrete probability densities, given the probabilities on the same support.

Examples

```
# Example 1
x1 <- data.frame(x = factor(c("A", "A", "B", "B")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
```

```
                    y = factor(c("a", "a", "b", "a", "b")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")),
                    y = factor(c("a", "b", "a", "b", "a", "b")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)
```

---

matddjeffreyspar   *Matrix of divergences between discrete probability densities given the*
                          probabilities on their common support

---

Description

Computes the matrix of Jeffreys divergences between several multivariate or univariate discrete
probability distributions on the same support (which can be a Cartesian product of q sets), given the
probabilities of the states (which are q-tuples) of the support.

Usage

```
matddjeffreyspar(freq)
```

Arguments

freq   list of arrays. Their dim attribute is a vector with length q, its elements contain-
                  ing the numbers of levels of the *sets*. Each array contains the probabilities of
                  the discrete distribution on the same support.

Value

Positive symmetric matrix whose order is equal to the number of distributions, consisting of the
pairwise Jeffreys divergences between these distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddjeffreyspar.

matddjeffreys for discrete probability densities which are estimated from the data.

---

matddjensen *Matrix of divergences between discrete probability densities given samples*

Description

Computes the matrix of the Jensen-Shannon divergences between several multivariate or univariate discrete probability distributions, estimated from samples.

Usage

```
matddjensen(x)
```

Arguments

x   object of class `"folder"` containing the data. Its elements are data frames (one data frame per distribution) whose columns are factors.

Value

Positive symmetric matrix whose order is equal to the number of data frames (or distributions), consisting of the pairwise Jensen-Shannon divergences between the distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddjensen](#).

[matddjensenpar](#) for discrete probability densities, given the probabilities on the same support.

Examples

```
# Example 1
x1 <- data.frame(x = factor(c("A", "A", "B", "B")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)

# Example 2
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
```

```
                y = factor(c("a", "a", "b", "a", "b")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")),
                y = factor(c("a", "b", "a", "b", "a", "b")))
xf <- folder(x1, x2, x3)
matddhellinger(xf)
```

---

matddjensenpar   *Matrix of divergences between discrete probability densities given the*
                               probabilities on their common support

---

Description

Computes the matrix of the Jensen-Shannon divergences between several multivariate or univariate discrete probability distributions on the same support (which can be a Cartesian product of q sets), given the probabilities of the states (which are q-tuples) of the support.

Usage

```
matddjensenpar(freq)
```

Arguments

freq   list of arrays. Their dim attribute is a vector with length q, its elements containing the numbers of levels of the *sets*. Each array contains the probabilities of the discrete distribution on the same support.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise Jensen-Shannon divergences between the discrete probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

ddjensenpar.

matddjensen for discrete probability densities which are estimated from the data.

---

matddlp   *Matrix of distances between discrete probability distributions given samples*

Description

Computes the matrix of the L^p distances between several multivariate or univariate discrete probability distributions, estimated from samples.

Usage

```
matddlp(x, p = 1)
```

Arguments

x   object of class `"folder"` containing the data. Its elements are data frames (one data frame per distribution) whose columns are factors.

p   integer. Parameter of the distance.

Value

Positive symmetric matrix whose order is equal to the number of data frames (or distributions), consisting of the pairwise L^p distances between the distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddlp](#).

[matddlppar](#) for discrete probability distributions, given the probabilities on the same support.

Examples

```
# Example 1
x1 <- data.frame(x = factor(c("A", "A", "B", "B")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")))
xf <- folder(x1, x2, x3)
matddlp(xf)
matddlp(xf, p = 2)

# Example 2
```

```
x1 <- data.frame(x = factor(c("A", "A", "A", "B", "B", "B")),
                 y = factor(c("a", "a", "a", "b", "b", "b")))
x2 <- data.frame(x = factor(c("A", "A", "A", "B", "B")),
                 y = factor(c("a", "a", "b", "a", "b")))
x3 <- data.frame(x = factor(c("A", "A", "B", "B", "B", "B")),
                 y = factor(c("a", "b", "a", "b", "a", "b")))
xf <- folder(x1, x2, x3)
matddlp(xf, p = 1)
```

---

matddlppar   *Matrix of distances between discrete probability densities given the*
                           probabilities on their common support

---

Description

Computes the matrix of the L^p distances between several multivariate or univariate discrete probability distributions on the same support (which can be a Cartesian product of q sets), given the probabilities of the states (which are q-tuples) of the support.

Usage

```
matddlppar(freq, p = 1)
```

Arguments

freq   list of arrays. Their dim attribute is a vector with length q, its elements contain-
             ing the numbers of levels of the $sets$. Each array contains the probabilities of
             the discrete distribution on the same support.

p   integer. Parameter of the distance.

Value

Positive symmetric matrix whose order is equal to the number of distributions, consisting of the pairwise L^p distances between these distributions.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Deza, M.M. and Deza E. (2013). Encyclopedia of distances. Springer.

See Also

[ddlppar](#).

[matddlp](#) for discrete probability distributions which are estimated from samples.

---

matdistl2d   *Matrix of $L\hat{\ }2$ distances between probability densities*

Description

Computes the matrix of the L^2 distances between several multivariate ($p > 1$) or univariate ($p = 1$) probability densities, estimated from samples.

Usage

```
matdistl2d(x, method = "gaussiand", varwL = NULL)
```

Arguments

x	object of class "folder" containing the data. Its elements have only numeric variables (observations of the probability densities). If there are non numeric variables, there is an error.
method	string. It can be:

- "gaussiand" if the densities are considered to be Gaussian.
- "kern" if they are estimated using the Gaussian kernel method.

varwL	list of matrices. The smoothing bandwidths for the estimation of each probability density. If they are omitted, the smoothing bandwidths are computed using the normal reference rule matrix bandwidth (see details of the l2d function).

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise distances between the probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

distl2d.

matdistl2dpar when the probability densities are Gaussian, given the parameters (means and variances).

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
mean.X <- mean(X)
```

```
    var.X <- var.folder(X)

    # Parametrically estimated Gaussian densities:
    matdistl2d(X)

    ## Not run:
  # Estimated densities using the Gaussian kernel method )normal reference rule bandwidth):
    matdistl2d(X, method = "kern")

    # Estimated densities using the Gaussian kernel method (bandwidth provided):
    matdistl2d(X, method = "kern", varwL = var.X)

  ## End(Not run)

    # Univariate :
    X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
    summary(X1)
    mean.X1 <- mean(X1)
    var.X1 <- var.folder(X1)

    # Parametrically estimated Gaussian densities:
    matdistl2d(X1)

  # Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
    matdistl2d(X1, method = "kern")

  # Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
    matdistl2d(X1, method = "kern", varwL = var.X1)
```

---

matdistl2dnorm   *Matrix of L^2 distances between L^2-normed probability densities*

Description

Computes the matrix of the L^2 distances between several multivariate ($p > 1$) or univariate ($p = 1$) L^2-normed probability densities, estimated from samples, where a L^2-normed probability density is the original probability density function divided by its L^2-norm.

Usage

```
matdistl2dnorm(x, method = "gaussiand", varwL = NULL)
```

Arguments

x   object of class "folder" containing the data. Its elements have only numeric
                variables (observations of the probability densities). If there are non numeric
                variables, there is an error.

method   string. It can be:

                • "gaussiand" if the densities are considered to be Gaussian.

- "kern" if they are estimated using the Gaussian kernel method.

varwL   list of matrices. The smoothing bandwidths for the estimation of each probability density. If they are omitted, the smoothing bandwidths are computed using the normal reference rule matrix bandwidth (see details of the l2d function).

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise distances between the L^2-normed probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

distl2dnorm.

matdistl2d for the distance matrix between probability densities.

matdistl2dnormpar when the probability densities are Gaussian, given the parameters (means and variances).

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
mean.X <- mean(X)
var.X <- var.folder(X)

# Parametrically estimated Gaussian densities:
matdistl2dnorm(X)

## Not run:
# Estimated densities using the Gaussian kernel method ()normal reference rule bandwidth):
matdistl2dnorm(X, method = "kern")

# Estimated densities using the Gaussian kernel method (bandwidth provided):
matdistl2dnorm(X, method = "kern", varwL = var.X)

## End(Not run)

# Univariate :
X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
summary(X1)
mean.X1 <- mean(X1)
var.X1 <- var.folder(X1)

# Parametrically estimated Gaussian densities:
matdistl2dnorm(X1)
```

```
# Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
  matdistl2dnorm(X1, method = "kern")

# Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
  matdistl2dnorm(X1, method = "kern", varwL = var.X1)
```

---

matdistl2dnormpar   *Matrix of L^2 distances between L^2-normed Gaussian densities given their parameters*

Description

Computes the matrix of the L^2 distances between several multivariate ($p > 1$) or univariate ($p = 1$) L^2-normed Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate), where a L^2-normed Gaussian density is the original probability density function divided by its L^2-norm.

Usage

```
matdistl2dnormpar(meanL, varL)
```

Arguments

meanL   list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.

varL   list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the Gaussian densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise distances between the L^2-normed probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

distl2dnormpar.

matdistl2dpar for the distance matrix between Gaussian densities, given their parameters.

matdistl2dnorm for the distance matrix between normed probability densities which are estimated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))

# Gaussian densities, given parameters
matdistl2dnormpar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)

# Gaussian densities, given parameters
matdistl2dnormpar(mean.X1, var.X1)
```

---

matdistl2dpar	*Matrix of L^2 distances between Gaussian densities given their parameters*

Description

Computes the matrix of the L^2 distances between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate).

Usage

```
matdistl2dpar(meanL, varL)
```

Arguments

meanL	list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.
varL	list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the Gaussian densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise distances between the probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

distl2dpar.

matdistl2d for the distance matrix between probability densities which are estimated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))

# Gaussian densities, given parameters
matdistl2dpar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)

# Gaussian densities, given parameters
matdistl2dpar(mean.X1, var.X1)
```

---

mathellinger   *Matrix of Hellinger distances between Gaussian densities*

Description

Computes the matrix of the Hellinger distances between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities given samples and using hellinger.

Usage

```
mathellinger(x)
```

Arguments

x   object of class "folder" containing the data. Its elements have only numeric
                   variables (observations of the probability densities). If there are non numeric
                   variables, there is an error.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise Hellinger distances between the probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

hellinger.

mathellingerpar when the probability densities are Gaussian, given the parameters (means and variances).

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
mathellinger(X)

# Univariate :
X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
summary(X1)
mathellinger(X1)
```

---

mathellingerpar	*Matrix of Hellinger distances between Gaussian densities given their parameters*

Description

Computes the matrix of the Hellinger distances between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their means and variances, using hellingerpar.

Usage

```
mathellingerpar(meanL, varL)
```

Arguments

meanL	list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.
varL	list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the Gaussian densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise distances between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

hellingerpar.

mathellinger for the distance matrix between probability densities which are estimated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))
mathellingerpar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)
mathellingerpar(mean.X1, var.X1)
```

---

matipl2d   *Matrix of L^2 inner products of probability densities*

Description

Computes the matrix of the L^2 inner products between several multivariate ($p > 1$) or univariate ($p = 1$) probability densities, estimated from samples, using l2d.

Usage

```
matipl2d(x, method = "gaussiand", varwL = NULL)
```

Arguments

x   object of class "folder" containing the data. Its elements have only numeric variables (observations of the probability densities). If there are non numeric variables, there is an error.

method   string. It can be:

- "gaussiand" if the densities are considered to be Gaussian.
- "kern" if they are estimated using the Gaussian kernel method.

varwL   list of matrices. The smoothing bandwidths for the estimation of each probability density. If they are omitted, the smoothing bandwidths are computed using the normal reference rule matrix bandwidth (see details of the `l2d` function).

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise inner products between the probability densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

`l2d`.

`matipl2dpar` when the probability densities are Gaussian, given the parameters (means and variances).

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
mean.X <- mean(X)
var.X <- var.folder(X)

# Parametrically estimated Gaussian densities:
matipl2d(X)

# Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
matipl2d(X, method  = "kern")

# Estimated densities using the Gaussian kernel method (bandwidth provided):
matipl2d(X, method  = "kern", varwL = var.X)

# Univariate :
X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
summary(X1)
mean.X1 <- mean(X1)
var.X1 <- var.folder(X1)

# Parametrically estimated Gaussian densities:
matipl2d(X1)

# Estimated densities using the Gaussian kernel method (normal reference rule bandwidth):
matipl2d(X1, method = "kern")

# Estimated densities using the Gaussian kernel method (bandwidth provided):
matipl2d(X1, method = "kern", varwL = var.X1)
```

***

matipl2dpar   *Matrix of Lˆ2 inner products of Gaussian densities*

***

Description

Computes the matrix of the L^2 inner products between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate).

Usage

```
matipl2dpar(meanL, varL)
```

Arguments

meanL	list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.
varL	list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the Gaussian densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise inner products between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

l2dpar.

matipl2d for the distance matrix between probability densities which are estimated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))

# Gaussian densities, given parameters
matipl2dpar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
```

```
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)

# Gaussian densities, given parameters
matipl2dpar(mean.X1, var.X1)
```

---

matjeffreys *Matrix of the Jeffreys measures (symmetrised Kullback-Leibler diver-*
gences) between Gaussian densities

Description

Computes the matrix of Jeffreys measures between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given samples.

Usage

```
matjeffreys(x)
```

Arguments

x   object of class "folder" containing the data. Its elements have only numeric variables (observations of the probability densities). If there are non numeric variables, there is an error.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of pairwise Jeffreys measures between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[matjeffreyspar](#) if the parameters of the Gaussian densities are known.

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
matjeffreys(X)

# Univariate :
```

```
X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
summary(X1)
matjeffreys(X1)
```

---

matjeffreyspar	*Matrix of Jeffreys measures (symmetrised Kullback-Leibler diver-gences) between Gaussian densities*

Description

Computes the matrix of Jeffreys measures between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate), using `jeffreyspar`.

Usage

```
matjeffreyspar(meanL, varL)
```

Arguments

meanL	list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.
varL	list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the probability densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of pairwise Jeffreys measures between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

`jeffreyspar`.

`matjeffreys` for the matrix of Jeffreys divergences between probability densities which are esti-mated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))
```

```
matjeffreyspar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)
matjeffreyspar(mean.X1, var.X1)
```

---

matwasserstein   *Matrix of 2-Wassterstein distance between Gaussian densities*

Description

Computes the matrix of the 2-Wassterstein distances between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given samples.

Usage

```
matwasserstein(x)
```

Arguments

x   object of class "folder" containing the data. Its elements have only numeric variables (observations of the probability densities). If there are non numeric variables, there is an error.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise 2-Wassterstein distance between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[matwassersteinpar](#) if the parameters of the Gaussian densities are known.

Examples

```
data(roses)

# Multivariate:
X <- as.folder(roses[,c("Sha","Den","Sym","rose")], groups = "rose")
summary(X)
matwasserstein(X)
```

```
# Univariate :
X1 <- as.folder(roses[,c("Sha","rose")], groups = "rose")
summary(X1)
matwasserstein(X1)
```

---

matwassersteinpar   *Matrix of 2-Wasserstein distances between Gaussian densities*

Description

Computes the matrix of the 2-Wasserstein distances between several multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities, given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate), using `wassersteinpar`.

Usage

```
matwassersteinpar(meanL, varL)
```

Arguments

meanL   list of the means ($p = 1$) or vector means ($p > 1$) of the Gaussian densities.

varL   list of the variances ($p = 1$) or covariance matrices ($p > 1$) of the probability densities.

Value

Positive symmetric matrix whose order is equal to the number of densities, consisting of the pairwise 2-Wasserstein distances between the Gaussian densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

`wasserstein`.

`matwasserstein` for the matrix of 2-Wasserstein distances between probability densities which are estimated from the data.

Examples

```
data(roses)

# Multivariate:
X <- roses[,c("Sha","Den","Sym","rose")]
summary(X)
mean.X <- as.list(by(X[, 1:3], X$rose, colMeans))
var.X <- as.list(by(X[, 1:3], X$rose, var))
```

```
matwassersteinpar(mean.X, var.X)

# Univariate :
X1 <- roses[,c("Sha","rose")]
summary(X1)
mean.X1 <- by(X1$Sha, X1$rose, mean)
var.X1 <- by(X1$Sha, X1$rose, var)
matwassersteinpar(mean.X1, var.X1)
```

---

mdsdd   *Multidimensional scaling of discrete probability distributions*

Description

Applies the multidimensional scaling (MDS) method to discrete probability distributions in order to describe T groups of individuals on which are observed q categorical variables. It returns an object of class mdsdd. It applies [cmdscale](#) to the distance matrix between the T distributions.

Usage

```
mdsdd(xf, group.name = "group", distance = c("l1", "l2", "chisqsym", "hellinger",
   "jeffreys", "jensen", "lp"), nb.factors = 3, nb.values = 10, association = c("cramer",
    "tschuprow", "pearson", "phi"), sub.title = "", plot.eigen = TRUE,
    plot.score = FALSE, nscore = 1:3, filename = NULL, add = TRUE, p)
```

Arguments

xf   object of class [folder](#), list of arrays (or tables) or data frame.

- If it is a folder, its elements are data frames with q columns (considered as factors). The tth element ($t = 1, \ldots, T$) matches with the tth group.
- If it is a data frame, the columns with name given by the group.name argument is a factor giving the groups. The other columns are all considered as factors.
- If it is a list of arrays (or tables), the tth element ($t = 1, \ldots, T$) is the table of the joint frequency distribution of q variables within the tth group. The frequency distribution is expressed with relative or absolute frequencies. These arrays have the same shape.
  Each array (or table) xf[[i]] has:
  - the same dimension(s). If $q = 1$ (univariate), dim(xf[[i]]) is an integer. If $q > 1$ (multivariate), dim(xf[[i]]) is an integer vector of length q.
  - the same dimension names dimnames(xf[[i]]) (is non NULL). These dimnames are the names of the variables.
  The elements of the arrays are non-negative numbers (if they are not, there is an error).

group.name   string. Name of the grouping variable. Default: groupname = "group".

distance   The distance or divergence used to compute the distance matrix between the
                        discrete distributions (see Details). It can be:

- "l1" (default) the L^p distance with $p = 1$
- "l2" the L^p distance with $p = 2$
- "chisqsym" the symmetric Chi-squared distance
- "hellinger" the Hellinger metric (Matusita distance)
- "jeffreys" Jeffreys distance (symmetrised Kullback-Leibler divergence)
- "jensen" the Jensen-Shannon distance
- "lp" the L^p distance with p given by the argument p of the function.

nb.factors   numeric. Number of returned principal coordinates (default nb.factors = 3).
                        This number must be less than $T - 1$.

                        Warning: The plot.mdsdd and interpret.mdsdd functions cannot take into
                        account more than nb.factors principal factors.

nb.values   numeric. Number of returned eigenvalues (default nb.values = 10).

association   The association measure between two discrete distributions to be used (see De-
                        tails). It can be:

- "cramer" (default) Cramer's V (see cramer.folder).
- "tschuprow" Tschuprow's T (tschuprow.folder).
- "pearson" Pearson's contingency coefficient (pearson.folder).
- "phi" phi (phi.folder).

sub.title   string. Subtitle for the graphs (default NULL).

plot.eigen   logical. If TRUE (default), the barplot of the eigenvalues is plotted.

plot.score   logical. If TRUE, the graphs of new coordinates are plotted. A new graphic device
                        is opened for each pair of coordinates defined by nscore argument.

nscore   numeric vector. If plot.score = TRUE, the numbers of the principal coordinates
                        which are plotted. By default, nscore = 1:3. Its components cannot be greater
                        than nb.factors.

filename   string. Name of the file in which the results are saved. By default (filename =
                        NULL) they are not saved.

add   logical indicating if an additive constant should be computed and added to the
                        non diagonal dissimilarities such that the modified dissimilarities are Euclidean
                        (default TRUE; see add argument of cmdscale).

p   integer. Optional. When distance = "lp" (L^p distance with $p > 2$), p is the
                        parameter of the distance.

Details

If a folder is given as argument, the T discrete probability distributions f_t corresponding to the T
groups of individuals are estimated from observations. Then the distances/dissimilarities between
the estimated distributions are computed, using the distance or divergence defined by the distance
argument:

If the distance is "l1", "l2" or "lp", the distances are computed by the function matddlppar. Oth-
erwise, it can be computed by matddchisqsympar ("chisqsym"), matddhellingerpar ("hellinger"),
matddjeffreyspar ("jeffreys") or matddjensenpar ("jensen").

The association measures are computed accordingly to the value of the parameter association The computation uses the corresponding function of the package DescTools (see `Assocs`). Notice that an association measure between a constant variable with and other variable is set to zero. The association measure between each variable with itself is not computed and the diagonal of the returned association matrices is set to NA.

Value

Returns an object of class mdsdd, that is a list including:

inertia   data frame of the eigenvalues and the percentages of their sum.

scores   data frame of the coordinates along the nb.factors first principal coordinates.

jointp   list of arrays. The joint probability distribution for each group.

margins   list of two data frames giving respectively:

- The probability distribution of each variable for each group. Each column of the data frame corresponds to one level of one categorical variable and contains the probabilities of this level in each group.
- The joint probability distribution of each pair of variables for each group. Each column of the data frame corresponds to one pair of levels of two categorical variables (one level per variable) and contains the probabilities of this pair of levels in each group.

associations   list of T matrices. Each matrix corresponds to a group and gives the pairwise association measures between the q categorical variables.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

References

Cox, T.F., Cox, M.A.A. (2001). Multidimensional Scaling, second ed. Chapman & Hall/CRC.

Saporta, G. (2006). Probabilit\'es, Analyse des donn\'ees et Statistique. Editions Technip, Paris.

See Also

print.mdsdd, plot.mdsdd, interpret.mdsdd

Examples

```
# Example 1 with a folder (10 groups) of 3 factors
# obtained by converting numeric variables
data(roses)
xr = roses[,c("Sha", "Den", "Sym", "rose")]
xf = as.folder(xr, groups = "rose")
xf = cut(xf, breaks = list(c(0, 5, 7, 10), c(0, 4, 6, 10), c(0, 6, 8, 10)), cutcol = 1:3)
af = mdsdd(xf)
print(af)
print(af$jointp)
print(af$margins[[1]]) # equivalent to print(af$margins$margin1)
```

```
print(af$margins[[2]])
print(af$associations)

# Example 2 with a data frame obtained by converting numeric variables
data(roses)
xr = roses[,c("Sha", "Den", "Sym", "rose")]
xr = cut(xr, breaks = list(c(0, 5, 7, 10), c(0, 4, 6, 10), c(0, 6, 8, 10)), cutcol = 1:3)
ar = mdsdd(xr, group.name = "rose")
print(ar)
print(ar$jointp)
print(ar$margins[[1]]) # equivalent to print(ar$margins$margin1)
print(ar$margins[[2]])
print(ar$associations)

# Example 3 with a list of 7 arrays
data(dspg)
xl = dspg
mdsdd(xl)
```

---

mean.folder   *Means of a folder of data sets*

Description

Computes the means by column of the elements of an object of class folder.

Usage

```
## S3 method for class 'folder'
mean(x, ..., na.rm = FALSE)
```

Arguments

x	an object of class [folder](#) that is a list of data frames with the same column names.
...	further arguments passed to or from other methods.
na.rm	logical. Should missing values (including NaN) be omitted from the calculations? (see [mean](#) or [colMeans](#))

Details

It uses [colMeans](#) to compute the mean by numeric column of each element of the folder. If some columns of the data frames are not numeric, there is a warning, and the means are computed on the numeric columns only.

Value

A list whose elements are the mean by column of the elements of the folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#) to create an object of class folder. [var.folder](#), [cor.folder](#), [skewness.folder](#), [kurtosis.folder](#) for other statistics for folder objects.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
iris.means <- mean(iris.fold)
print(iris.means)

# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
roses.means <- mean(roses.fold)
print(roses.means)
```

---

mtgcomponents   *Components of upper scale of a vertex*

Description

For a vertex in an object of class [foldermtg](#), computes its decomposition into vertices of an upper scale.

Usage

```
mtgcomponents(x, vertex, scale)
```

Arguments

x	an object of class [foldermtg](#).
vertex	character. The identifier of a vertex. These identifiers are the rownames of the data frame x$topology.
scale	integer. The scale of the components of vertex which will be returned.

Details

If vertex is a vertex of scale i, then scale (the scale of the returned components of vertex) must be higher than i. For example, if vertex is a vertex of scale 2, then scale > 2, for instance scale = 3. The returned components are then vertices of scale 3 which have a decomposition relationship with vertex.

Value

A character vector, containing the idendifiers of the components of vertex.

If there is no component, then the returned vector is empty.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

read.mtg: reads a MTG file and builds an object of class foldermtg.

mtgorder, mtgrank.

Examples

```
mtgfile <- system.file("extdata/plant1.mtg", package = "dad")
xmtg <- read.mtg(mtgfile)

# Vertex of class "P" (plant, of scale 1), components of class 2 (axes: "A")
mtgcomponents(xmtg, vertex = "v01", scale = 2)

# Vertex of class "P" (plant, of scale 1), components of class 3 ("O", "M" and "I")
mtgcomponents(xmtg, vertex = "v01", scale = 3)

# Vertex of class "A" (stem, of scale 2), components of class 3 ("O", "M" and "I")
mtgcomponents(xmtg, vertex = "v12", scale = 3)
```

---

mtgorder   *Branching order of vertices*

Description

Computes the branching order of vertices contained in an object of class foldermtg. The order of a vertex is the number of the column of topology, which contains this vertex.

Usage

```
mtgorder(x, classes = "all", display = FALSE)
```

Arguments

x	an object of class `foldermtg`.
classes	character vector. The classes of entities for which the branching order is computed. If omitted, the branching orders are computed for all entities.
display	logical. If TRUE, the data frames of x corresponding to classes are displayed. Default: FALSE.

Details

Returns x after appending the branching orders of the vertices of the classes given in the argument classes. The branching orders are appended to the data frames containing the vertices (one data frame per class) and the values of their corresponding features.

Value

Returns an object of class `foldermtg`, that is a list of data frames.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

`read.mtg`: reads a MTG file and builds an object of class foldermtg.

`mtgorder`.

Examples

```
mtgfile <- system.file("extdata/plant1.mtg", package = "dad")
xmtg <- read.mtg(mtgfile)

# The branching orders
ymtg <- mtgorder(xmtg)
print(ymtg)

# Add the branching orders to the 'foldermtg'
zmtg <- mtgorder(xmtg, display = TRUE)
print(zmtg)
```

---

mtgplant1   *Class* foldermtg

Description

These data produced by the SAGAH team (Sciences Agronomiques Appliquées à l'Horticulture, now Research Institute on Horticulture and Seeds), provide the topological structure of a rosebush.

Usage

```
data("mtgplant1")
```

Format

This object of class [foldermtg](#) is a list of 10 data frames:

mtgplant1$classes: data frame with 6 rows and 5 columns named SYMBOL (factor: the classes of the vertices), SCALE (integer: the scale at which they appear), DECOMPOSITION (factor), INDEXATION (factor) and DEFINITION (factor).

> The vertex classes are:
>
> - P: the whole plant (scale 1)
> - A: the axes (scale 2)
> - O, M, I: the ..., metamers (phytomers) and inflorescences (scale 3)

mtgplant1$description: data frame with 8 rows and 4 columns (factors) named LEFT, RIGHT, RELTYPE and MAX.

mtgplant1$features: data frame with 13 rows and 2 columns (factors) named NAME and TYPE.

mtgplant1$topology: data frame with 88 rows and 4 columns:

> - order1, order2 and order3 (factors): the codes of the vertices, as they are found in the MTG table of the MTG file. The column on which a code appears gives the branching order of the corresponding vertex.
> - vertex (character): the same codes of vertices, on a single column.

mtgplant1$coordinates: data frame with 86 rows and 6 columns (numeric) named XX, YY and 22: cartesian coordinates of the vertices, and AA, BB and CC: an other coordinates system.

mtgplant1$P, mtgplant1$A, mtgplant1$M, mtgplant1$I: data frames of the features on the vertices (all numeric).

Details

This object of class foldermtg can be built by reading the data in a MTG file (see examples).

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

read.mtg: to read an MTG file and build an object of class MTG.

mtgplant2: an other example of such data.

Examples

```
data(mtgplant1)
print(mtgplant1)

# To read these data from a MTG file:
mtgfile1 <- system.file("extdata/plant1.mtg", package = "dad")
mtgplant1 <- read.mtg(mtgfile1)
print(mtgplant1)
```

---

mtgplant2   *Class* foldermtg

Description

These data provides the topology of a bushy plant.

Usage

```
data("mtgplant2")
```

Format

This object of class foldermtg is a list of 9 data frames:

mtgplant2$classes: data frame with 6 rows and 5 columns named SYMBOL (factor: the classes of the vertices), SCALE (integer: the scale at which they appear), DECOMPOSITION (factor), INDEXATION (factor) and DEFINITION (factor).

   The vertex classes are:

   - P: the whole plant (scale 1)
   - A: the axes (scale 2)
   - F, I: the flower and internodes (scale 3)

mtgplant2$description: data frame with 4 rows and 4 columns (factors) named LEFT, RIGHT, RELTYPE and MAX.

mtgplant2$features: data frame with 9 rows and 2 columns (factors) named NAME and TYPE.

mtgplant2$topology: data frame with 14 rows and 3 columns:

   - order1 and order2 (factors): the codes of the vertices, as they are found in the MTG table of the MTG file. The column on which a code appears gives the branching order of the corresponding vertex.
   - vertex (character): the same codes of vertices, on a single column.

mtgplant2$coordinates: data frame with 0 rows and 0 columns (there are no spatial coordinates in these MTG data).

mtgplant2$P, mtgplant2$A, mtgplant2$F **and** mtgplant2$I: data frames of the features on the vertices (all numeric).

Details

This object of class foldermtg can be built by reading the data in a MTG file (see examples).

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

read.mtg: to read an MTG file and build an object of class MTG.

mtgplant1: an other example of such data.

Examples

```
data(mtgplant2)
print(mtgplant2)

# To read these data from a MTG file:
mtgfile2 <- system.file("extdata/plant2.mtg", package = "dad")
mtgplant2 <- read.mtg(mtgfile2)
print(mtgplant2)
```

---

mtgrank   *Ranks of vertices in a decomposition*

Description

Computes the rank of the vertices contained in an object of class foldermtg. The vertex sequences resulting from a decomposition of other vertices, the rank of the vertices making up the sequences are computed from the beginning of the sequence or from its end. These ranks can be absolute or relative.

For example: ranks of the phytomeres and inflorescences in each stem.

Usage

```
mtgrank(x, classe, parent.class = NULL, sibling.classes = NULL,
  relative = FALSE, from = c("origin", "end"), rank.name = "Rank",
  display = FALSE)
```

Arguments

x	an object of class foldermtg.
classe	character. The class of the vertices for which the ranks are computed.
parent.class	character. The class of the parent entities of those for which the ranks are computed. If omitted, the entities of scale maxscal - 1, where maxscal is the highest scale in x data.

sibling.classes

character vector. The classes of vertices appearing at the same scale as classe, which are used in the computing of the ranks.

If omitted, only the vertices of class classe are used to compute the ranks.

relative   logical. If TRUE, the relative ranks are computed, i.e. ranks from 0 to 1. Default: FALSE.

from   character. It can be "origin" (default) or "end".

If from = "origin", the ranks are computed from the origin to the end, i.e. from 1 to its maximum (from 0 to 1 if relative = TRUE). If from = "end", they are computed from the end to the origin, i.e. from the maximum to 1 (from 1 to 0 if relative = TRUE).

rank.name   character. Name of the rank column that is appended to x[[classe]]. The default is "Rank".

display   logical. If TRUE, the data frames of x corresponding to classes are displayed. Default: FALSE.

Details

If the branching orders of the entities given by classe, parent.class and, if relevant, sibling.classes are not contained in x, mtgrank() uses [mtgorder](#) to compute them. The ranks are appended to the data frames containing the vertices (one data frame per class) and the values of their corresponding features.

Value

Returns an object of class [foldermtg](#), that is a list of data frames.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). [MTG user guide](#)

See Also

[read.mtg](#): reads a MTG file and builds an object of class foldermtg.

[mtgorder](#).

Examples

```
mtgfile <- system.file("extdata/plant1.mtg", package = "dad")
xmtg <- read.mtg(mtgfile)

ymtg <- mtgrank(xmtg, "M")
print(ymtg)

mtgrank(xmtg, "M", display = TRUE)
```

```
mtgrank(xmtg, "M", parent.class = "A", display = TRUE)
mtgrank(xmtg, "M", parent.class = "A", sibling.classes = c("O", "I"), display = TRUE)
mtgrank(xmtg, "M", relative = TRUE, display = TRUE)
mtgrank(xmtg, "M", from = "origin", display = TRUE)
mtgrank(xmtg, "M", from = "end", display = TRUE)
```

---

plot.dstatis   *Plotting scores of STATIS method (interstructure) analysis*

Description

Applies to an object of class "dstatis" (see details of the dstatis.inter function). Plots the scores.

Usage

```
## S3 method for class 'dstatis'
plot(x, nscore = c(1, 2), sub.title = NULL, color = NULL, fontsize.points = 1.5, ...)
```

Arguments

x	object of class "dstatis" (returned by dstatis.inter).
nscore	a length 2 numeric vector. The numbers of the score vectors to be plotted.
	Warning: Its components cannot be greater than the nb.factors argument in the call of the dstatis.inter function.
sub.title	string. Subtitle to be added to each graph.
color	When provided, the colour of the symbols of each group. Can be a vector with length equal to the number of groups.
fontsize.points	
	Numeric. Expansion of the characters (or symbols) of the groups on the graph. This works as a multiple of par("cex") (see points).
...	optional arguments to plot methods.

Details

Plots the principal scores returned by the dstatis.inter function. A new graphics window is opened for each pair of principal axes defined by the nscore argument.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Lavit, C., Escoufier, Y., Sabatier, R., Traissac, P. (1994). The ACT (STATIS method). Computational Statistics & Data Analysis, 18 (1994), 97-119.

See Also

[dstatis.inter](); [print.dstatis](); [interpret.dstatis]().

Examples

```
data(roses)
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])

# Dual STATIS on the covariance matrices
result <- dstatis.inter(rosesf, data.scaled = FALSE, group.name = "rose")
plot(result)
```

---

plot.fhclustd   *Plotting a hierarchical clustering*

Description

Applies to an object of class fhclustd (see details of the [fhclustd]() function). Plots the dendogram.

Usage

```
## S3 method for class 'fhclustd'
plot(x, labels = NULL, hang = 0.1, check = TRUE, axes = TRUE,
                    frame.plot = FALSE, ann = TRUE,
                    main = "HCA of probability density functions",
                    sub = NULL, xlab = NULL, ylab = "Height", ...)
```

Arguments

x	object of class fhclustd (returned by [fhclustd]()).
labels, hang, check, axes, frame.plot, ann, main, sub, xlab, ylab	
	Arguments concerning the graphical representation of the dendogram. See [plot.hclust]().
...	Further graphical arguments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[fhclustd](); [print.fhclustd]().

Examples

```
data(castles.dated)
xf <- as.folder(castles.dated$stones)
## Not run:
result <- fhclustd(xf)
plot(result)
plot(result, hang = -1)

## End(Not run)
```

---

plot.fmdsd   *Plotting scores of multidimensional scaling of density functions*

Description

Applies to an object of class "fmdsd" (see the details section of the [fmdsd](#) function). Plots the scores.

Usage

```
## S3 method for class 'fmdsd'
plot(x, nscore = c(1, 2), main="MDS of probability density functions",
    sub.title = NULL, color = NULL, fontsize.points = 1.5, ...)
```

Arguments

x	object of class "fmdsd".
nscore	a length 2 numeric vector. The numbers of the score vectors to be plotted.
	Warning: Its components cannot be greater than the nb.factors argument in the call of the [fmdsd](#) function.
main	this argument to title has an useful default here.
sub.title	string. Subtitle to be added to each graph.
color	When provided, the colour of the symbols of each group. Can be a vector with length equal to the number of groups.
fontsize.points	
	Numeric. Expansion of the characters (or symbols) of the groups on the graph. This works as a multiple of par("cex") (see [points](#)).
...	optional arguments to plot methods.

Details

Plots the principal scores returned by the function [fmdsd](#). A new graphics window is opened for each pair of principal score vectors defined by the nscore argument.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

[fmdsd](); [print.fmdsd](); [interpret.fmdsd]().

Examples

```
data(roses)
x <- roses[,c("Sha","Den","Sym","rose")]
rosesfold <- as.folder(x)
result <- fmdsd(rosesfold)
plot(result)
```

---

plot.foldert	*Plotting data of a foldert*

Description

Applies to an object of class [foldert]() (called foldert below) that is a list. Plots the longitudinal evolution of a numeric variable for every individuals.

Usage

```
## S3 method for class 'foldert'
plot(x, which, na.inter = TRUE, type = "l", ylim = NULL, ylab = which,
                 main = "", ...)
```

Arguments

x	object of class [foldert]() that is a list of data frames with the same column names, each of them corresponding to a time of observation.
which	character. Name of a column of the data frames of x. It gives the name of the variable to be plotted.
	For each element x[[k]] of x, x[[k]] must be numeric. Otherwise, there is an error
na.inter	logical. If TRUE (default), for each individual, the missing values are deleted before plotting its evolution. If FALSE, the line corresponding to each individual is interrupted if there is a missing value, as for [matplot]().
type	character string (length 1 vector) or vector of 1-character strings (default "l") indicating the type of plot for each of the individuals followed among time, that is for each line of the data frames in the foldert. For further information about this argument, see [matplot]().

ylim	ranges of y axis. xlim is as in [matplot](). See details.
ylab	a label for the y axis. Default: the name of the plotted variable (which argument).
main	an overall title for the plot: see [title]().
...	optional arguments to plot methods.

Details

Internally, plot.foldert builds a matrix mdata containing the data of the variable given by which argument. The element mdata[ind, t] of this matrix is the value of the variable which for the individual ind: x[[t]][ind, which].

If the ylim argument is omitted, the range of y axis is given by range(mdata, na.rm = TRUE)*c(0, 1.2).

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

[foldert](): object of class foldert. [as.foldert.data.frame](): build an object of class foldert from a data frame. [as.foldert.array](): build an object of class foldert from a $3d$-array.

Examples

```
data(floribundity)
ftflor <- foldert(floribundity, cols.select = "union", rows.select = "union")
plot(ftflor, which = "nflowers", ylab = "Number of flowers per plant",
     main = "Floribundity of rosebushes, 2010, Angers (France)")
```

---

plot.fpcad   *Plotting scores of principal component analysis of density functions*

Description

Applies to an object of class "fpcad" (see details of the [fpcad]() function). Plots the scores.

Usage

```
## S3 method for class 'fpcad'
plot(x, nscore = c(1, 2), main = "PCA of probability density functions",
     sub.title = NULL, color = NULL, fontsize.points = 1.5, ...)
```

Arguments

x	object of class "fpcad" (returned by fpcad).
nscore	a length 2 numeric vector. The numbers of the score vectors to be plotted.
	Warning: Its components cannot be greater than the nb.factors argument in the call of the fpcad function.
main	this argument to title has an useful default here.
sub.title	string. Subtitle to be added to each graph.
color	When provided, the colour of the symbols of each group. Can be a vector with length equal to the number of groups.
fontsize.points	
	Numeric. Expansion of the characters (or symbols) of the groups on the graph. This works as a multiple of par("cex") (see points).
...	optional arguments to plot methods.

Details

Plots the principal scores returned by the fpcad function. A new graphics window is opened for each pair of principal axes defined by the nscore argument.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcad; print.fpcad; interpret.fpcad.

Examples

```
data(roses)
rosefold <- as.folder(roses[,c("Sha","Den","Sym","rose")])
result <- fpcad(rosefold)
plot(result)
```

---

plot.fpcat   *Plotting scores of principal component analysis of density functions among time*

Description

Applies to an object of class `"fpcat"` (see details of the [fpcat](#) function). Plots the scores.

Usage

```
## S3 method for class 'fpcat'
plot(x, nscore=c(1, 2), main = "PCA of probability density functions",
    sub.title = NULL, ...)
```

Arguments

x	object of class `"fpcat"` (returned by [fpcat](#)).
nscore	numeric or length 2 numeric vector. If it is a length 2 numeric vector (default), it contains the numbers of the score vectors to be plotted. If it is a single value, it is the number of the score which is plotted among time.
	Warning: The components of nscore cannot be greater than the nb.factors argument in the call of the [fpcat](#) function.
main	this argument to title has an useful default here.
sub.title	string. Subtitle to be added to each graph.
...	optional arguments to `plot` methods.

Details

Plots:

- if nscore is a length 2 vector (default): the principal scores returned by the [fpcat](#) function with arrows from the point corresponding to each time to the next one.
- if nscore is a single value, the principal scores among time with arrows from each time to the next one.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcat; print.fpcat

Examples

```
times <- as.Date(c("2017-03-01", "2017-04-01", "2017-05-01", "2017-06-01"))
x1 <- data.frame(z1=rnorm(6,1,5), z2=rnorm(6,3,3))
x2 <- data.frame(z1=rnorm(6,4,6), z2=rnorm(6,5,2))
x3 <- data.frame(z1=rnorm(6,7,2), z2=rnorm(6,8,4))
x4 <- data.frame(z1=rnorm(6,9,3), z2=rnorm(6,10,2))
ft <- foldert(x1, x2, x3, x4, times = times, rows.select="intersect")
print(ft)
result <- fpcat(ft)
plot(result)
plot(result, nscore = c(1, 2))
plot(result, nscore = 1)
plot(result)
```

---

plot.hclustdd   *Plotting a hierarchical clustering of discrete distributions*

Description

Applies to an object of class hclustdd (see details of the hclustdd function). Plots the dendogram.

Usage

```
## S3 method for class 'hclustdd'
plot(x, labels = NULL, hang = 0.1, check = TRUE, axes = TRUE,
                    frame.plot = FALSE, ann = TRUE,
                    main = "HCA of probability density functions",
                    sub = NULL, xlab = NULL, ylab = "Height", ...)
```

Arguments

x   object of class hclustdd (returned by hclustdd).

labels, hang, check, axes, frame.plot, ann, main, sub, xlab, ylab
                Arguments concerning the graphical representation of the dendogram. See plot.hclust.

...   Further graphical arguments.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

hclustdd; print.hclustdd.

Examples

```
data(dspg)
xl = dspg
result <- hclustdd(xl)
plot(result)
plot(result, hang = -1)
```

---

plot.mdsdd   *Plotting scores of multidimensional scaling analysis of discrete distri-*
                           butions

Description

Applies to an object of class "mdsdd" (see the details section of the [mdsdd](#) function). Plots the scores.

Usage

```
## S3 method for class 'mdsdd'
plot(x, nscore = c(1, 2), main="MDS of probability density functions",
     sub.title = NULL, color = NULL, fontsize.points = 1.5, ...)
```

Arguments

x	object of class "mdsdd".
nscore	a length 2 numeric vector. The numbers of the score vectors to be plotted.
	Warning: Its components cannot be greater than the nb.factors argument in the call of the [fmdsd](#) function.
main	this argument to title has an useful default here.
sub.title	string. Subtitle to be added to each graph.
color	When provided, the colour of the symbols of each group. Can be a vector with length equal to the number of groups.
fontsize.points	
	Numeric. Expansion of the characters (or symbols) of the groups on the graph. This works as a multiple of par("cex") (see [points](#)).
...	optional arguments to plot methods.

Details

Plots the principal scores returned by the function [mdsdd](#). A new graphics window is opened for each pair of principal score vectors defined by the nscore argument.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

See Also

mdsdd; print.mdsdd; interpret.mdsdd.

Examples

```
# INSEE (France): Diploma x Socio professional group, seven years.
data(dspg)
xlista = dspg
a <- mdsdd(xlista)
plot(a)
```

---

plotframes   *Plotting of two sets of variables*

Description

Plots a set of numeric variables vs. another set and prints the pairwise correlations. It uses the ggplot2 package.

Usage

```
plotframes(x, y, xlab = NULL, ylab = NULL, font.size = 12, layout = NULL)
```

Arguments

x	data frame (can also be a tibble). Variables on x coordinates.
y	data frame (or tibble). Variables on y coordinates.
xlab	a label for the x axis, by default the column names of y.
ylab	a label for the y axis (by default there is no label).
font.size	integer. Size of the characters in the strips.
layout	numeric vector of length 2 or 3 giving the number of columns, rows, and optionally pages of the lattice. If omitted, the graphs will be displayed on 3 lines and 3 columns, with a number of pages set to the required number.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

Examples

```
require(MASS)
mx <- c(0,0)
vx <- matrix(c(1,0,0,1),ncol = 2)
my <- c(0,1)
vy <- matrix(c(4,1,1,9),ncol = 2)
x <- as.data.frame(mvrnorm(n = 10, mu = mx, Sigma = vx))
y <- as.data.frame(mvrnorm(n = 10, mu = my, Sigma = vy))
```

```
colnames(x) <- c("x1", "x2")
colnames(y) <- c("y1", "y2")
plotframes(x, y)
```

---

print.discdd.misclass   *Printing results of discriminant analysis of discrete probability distri-*
                         butions

Description

Applies to an object of class ″discdd.misclass″. Prints the numerical results of [discdd.misclass](#).

Usage

```
## S3 method for class 'discdd.misclass'
print(x, dist.print=FALSE, prox.print=FALSE, digits=2, ...)
```

Arguments

x	object of class ″discdd.misclass″, returned by [discdd.misclass](#).
dist.print	logical. Its default value is FALSE. If TRUE, prints the matrix of distances between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
prox.print	logical. Its default value is FALSE. If TRUE, prints the matrix of proximity indices (in percent))between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
digits	numeric. Number of significant digits for the display of numerical results.
...	optional arguments to print methods.

Details

By default, are printed the whole misallocation ratio, the confusion matrix (allocations versus origins) with the misallocation ratios per class, and the data frame whose rows are the groups, and whose columns are the origin classes and allocation classes, and a logical variable indicating misclassification.

If dist.print = TRUE or prox.print = TRUE, the distances or proximity indices (in percent) between groups and classes, are displayed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[discdd.misclass](#); [print](#).

Examples

```
data("castles.dated")
stones <- castles.dated$stones
periods <- castles.dated$periods
stones$height <- cut(stones$height, breaks = c(19, 27, 40, 71), include.lowest = TRUE)
stones$width <- cut(stones$width, breaks = c(24, 45, 62, 144), include.lowest = TRUE)
stones$edging <- cut(stones$edging, breaks = c(0, 3, 4, 8), include.lowest = TRUE)
stones$boss <- cut(stones$boss, breaks = c(0, 6, 9, 20), include.lowest = TRUE )

castlefh <- folderh(periods, "castle", stones)

res <- discdd.misclass(castlefh, "period")

print(res)
```

---

print.discdd.predict   *Printing results of discriminant analysis of discrete probability distributions*

Description

print function, applied to an object of class "discdd.predict", prints numerical results of [discdd.predict](#) .

Usage

```
## S3 method for class 'discdd.predict'
print(x, dist.print=TRUE, prox.print=FALSE, digits=2, ...)
```

Arguments

x	object of class "discdd.predict", returned by [discdd.predict.](#)
dist.print	logical. If TRUE (the default), prints the matrix of distances between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
prox.print	logical. Its default value is FALSE. If TRUE, prints the matrix of proximity indices between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
digits	numerical. Number of significant digits for the display of numerical results.
...	optional arguments to print methods.

Details

By default, are printed:

- if available (if misclass.ratio argument of [discdd.predict](#) was TRUE), the whole misallocation ratio, the confusion matrix (allocations versus origins) and the misallocation ratio per class are printed.

- the data frame the rows of which are the groups, and the columns of which are of the origin (NA if not available) and allocation classes.

If dist.print = TRUE or prox.print = TRUE, the distances or proximity indices between groups and classes, are displayed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

discdd.predict; print.

Examples

```
data(castles.dated)
data(castles.nondated)
stones <- rbind(castles.dated$stones, castles.nondated$stones)
periods <- rbind(castles.dated$periods, castles.nondated$periods)
stones$height <- cut(stones$height, breaks = c(19, 27, 40, 71), include.lowest = TRUE)
stones$width <- cut(stones$width, breaks = c(24, 45, 62, 144), include.lowest = TRUE)
stones$edging <- cut(stones$edging, breaks = c(0, 3, 4, 8), include.lowest = TRUE)
stones$boss <- cut(stones$boss, breaks = c(0, 6, 9, 20), include.lowest = TRUE )

castlesfh <- folderh(periods, "castle", stones)

result <- discdd.predict(castlesfh, "period")
print(result)
print(result, prox.print=TRUE)
```

---

print.dstatis   *Printing results of STATIS method (interstructure) analysis*

Description

Applies to an object of class "dstatis". Prints the numeric results returned by the dstatis.inter function.

Usage

```
## S3 method for class 'dstatis'
print(x, mean.print = FALSE, var.print = FALSE,
  cor.print = FALSE, skewness.print = FALSE, kurtosis.print = FALSE,
  digits = 2, ...)
```

Arguments

x	object of class "dstatis", returned by the dstatis.inter function.
mean.print	logical. If TRUE, prints for each group the means and standard deviations of the variables and the norm of the density.
var.print	logical. If TRUE, prints for each group the variances and covariances of the variables.
cor.print	logical. If TRUE, prints for each group the correlations between the variables.
skewness.print	logical. If TRUE, prints for each group the skewness coefficients of the variables.
kurtosis.print	logical. If TRUE, prints for each group the kurtosis coefficients of the variables.
digits	numeric. Number of significant digits for the display of numeric results.
...	optional arguments to print methods.

Details

By default, are printed the inertia explained by the nb.values (see dstatis.inter) first principal components, the contributions, the qualities of representation of the densities along the nb.factors (see dstatis.inter) first principal components, and the principal scores.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Lavit, C., Escoufier, Y., Sabatier, R., Traissac, P. (1994). The ACT (STATIS method). Computational Statistics & Data Analysis, 18 (1994), 97-119.

See Also

dstatis.inter; plot.dstatis; interpret.dstatis; print.dstatis.

Examples

```
data(roses)
rosesf <- as.folder(roses[,c("Sha","Den","Sym","rose")])

# Dual STATIS on the covariance matrices
result <- dstatis.inter(rosesf, data.scaled = FALSE, group.name = "rose")
print(result)
```

---

print.fdiscd.misclass   *Printing results of discriminant analysis of probability density functions*

Description

Applies to an object of class `"fdiscd.misclass"`. Prints the numerical results of `fdiscd.misclass`.

Usage

```
## S3 method for class 'fdiscd.misclass'
print(x, dist.print=FALSE, prox.print=FALSE, digits=2, ...)
```

Arguments

x	object of class `"fdiscd.misclass"`, returned by `fdiscd.misclass`.
dist.print	logical. Its default value is FALSE. If TRUE, prints the matrix of distances between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
prox.print	logical. Its default value is FALSE. If TRUE, prints the matrix of proximity indices (in percent))between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
digits	numeric. Number of significant digits for the display of numerical results.
...	optional arguments to `print` methods.

Details

By default, are printed the whole misallocation ratio, the confusion matrix (allocations versus origins) with the misallocation ratios per class, and the data frame whose rows are the groups, and whose columns are the origin classes and allocation classes, and a logical variable indicating misclassification.

If `dist.print` = TRUE or `prox.print` = TRUE, the distances or proximity indices (in percent) between groups and classes, are displayed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (2004). Discriminant analysis with independently repeated multivariate measurements: an L^2 approach. Computational Statistics & Data Analysis, 47, 823-843.

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens, Centre de Recherches Archéologiques Médiévales de Saverne, 5, 5-38.

See Also

[fdiscd.misclass](); [print]().

Examples

```
data(castles.dated)
castlesfh <- folderh(castles.dated$periods, "castle", castles.dated$stones)
result <- fdiscd.misclass(castlesfh, "period")
print(result)
print(result, dist.print=TRUE)
print(result, prox.print=TRUE)
```

---

print.fdiscd.predict   *Printing results of discriminant analysis of probability density functions*

Description

`print` function, applied to an object of class `"fdiscd.predict"`, prints numerical results of [fdiscd.predict]().

Usage

```
## S3 method for class 'fdiscd.predict'
print(x, dist.print=TRUE, prox.print=FALSE, digits=2, ...)
```

Arguments

x	object of class `"fdiscd.predict"`, returned by [fdiscd.predict]().
dist.print	logical. If TRUE (the default), prints the matrix of distances between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
prox.print	logical. Its default value is FALSE. If TRUE, prints the matrix of proximity indices between, on one side, the groups (densities) and, on the other side, the classes (of groups or densities).
digits	numerical. Number of significant digits for the display of numerical results.
...	optional arguments to `print` methods.

Details

By default, are printed:

- if available (if `misclass.ratio` argument of [fdiscd.predict]() was TRUE), the whole misallocation ratio, the confusion matrix (allocations versus origins) and the misallocation ratio per class are printed.
- the data frame the rows of which are the groups, and the columns of which are of the origin (NA if not available) and allocation classes.

If `dist.print = TRUE` or `prox.print = TRUE`, the distances or proximity indices between groups and classes, are displayed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R. (2004). Discriminant analysis with independently repeated multivariate measurements: an L^2 approach. Computational Statistics & Data Analysis, 47, 823-843.

Rudrauf, J.M., Boumaza, R. (2001). Contribution à l'étude de l'architecture médiévale: les caractéristiques des pierres à bossage des châteaux forts alsaciens, Centre de Recherches Archéologiques médiévales de Saverne, 5, 5-38.

See Also

fdiscd.predict; print.

Examples

```
data(castles.dated)
data(castles.nondated)
castles.stones <- rbind(castles.dated$stones, castles.nondated$stones)
castles.periods <- rbind(castles.dated$periods, castles.nondated$periods)
castlesfh <- folderh(castles.periods, "castle", castles.stones)
result <- fdiscd.predict(castlesfh, "period")
print(result)
print(result, prox.print=TRUE)
```

---

print.fhclustd	*Printing results of a hierarchical clustering of probability density functions*

Description

print function, applied to an object of class "fhclustd", prints numerical results of fhclustd .

Usage

```
## S3 method for class 'fhclustd'
print(x, dist.print=FALSE, digits=2, ...)
```

Arguments

x	object of class "fhclustd", returned by fhclustd.
dist.print	logical. If TRUE (default: FALSE), prints the matrix of distances between the groups (densities).
digits	numerical. Number of significant digits for the display of numerical results.
...	optional arguments to print methods.

Details

If `dist.print = TRUE`, the distances between groups are displayed.

By default, the result of the clustering is printed. The display is the same as that of the `print.hclust` function.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

fhclustd; print.

Examples

```
data(castles.dated)
xf <- as.folder(castles.dated$stones)
## Not run:
result <- fhclustd(xf)
print(result)
print(result, dist.print = TRUE)

## End(Not run)
```

| print.fmdsd | *Printing results of a multidimensional scaling analysis of probability densities* |

Description

Applies to an object of class `"fmdsd"`. Prints the numeric results returned by the fmdsd function.

Usage

```
## S3 method for class 'fmdsd'
print(x, mean.print = FALSE, var.print = FALSE,
  cor.print = FALSE, skewness.print = FALSE, kurtosis.print = FALSE,
  digits = 2, ...)
```

Arguments

x	object of class `"fmdsd"`, returned by the fmdsd function.
mean.print	logical. If `TRUE`, prints for each group the means and standard deviations of the variables and the norm of the density.
var.print	logical. If `TRUE`, prints for each group the variances and covariances of the variables.
cor.print	logical. If `TRUE`, prints for each group the correlations between the variables.

skewness.print	logical. If TRUE, prints for each group the skewness coefficients of the variables.
kurtosis.print	logical. If TRUE, prints for each group the kurtosis coefficients of the variables.
digits	numeric. Number of significant digits for the display of numeric results.
...	optional arguments to print methods.

Details

By default, are printed the inertia explained by the nb.values (see fmdsd) first coordinates and the nb.factors (see fmdsd) coordinates of the densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fmdsd; plot.fmdsd; interpret.fmdsd; print.

Examples

```
data(roses)
x <- roses[,c("Sha","Den","Sym","rose")]
rosesfold <- as.folder(x)
result <- fmdsd(rosesfold)
print(result)
print(result, mean.print = TRUE)
```

---

print.foldermtg   *Printing an object of class* foldermtg

Description

print function, applied to an object of class "foldermtg", prints an MTG (Multiscale Tree Graph) folder, as returned by foldermtg function.

Usage

```
## S3 method for class 'foldermtg'
print(x, classes = TRUE, description = FALSE, features = TRUE,
  topology = FALSE, coordinates = FALSE, ...)
```

Arguments

x	an object of class [foldermtg]().
classes	logical. If TRUE (default), prints the data frame describing the classes (CLASSES: table in the MTG file).
description	logical. If TRUE (default: FALSE), prints the description data frame (DESCRIPTION: table in the MTG file).
features	logical. If TRUE (default), prints the data frame of the features and their types (FEATURES: table in the MTG file).
topology	logical. If TRUE (default: FALSE), prints the data frame of the plant topology.
coordinates	logical. If TRUE (default: FALSE), prints the spatial coordinates of the entities of the plant.
...	optional arguments to print methods.

Details

If classes, description or features are TRUE, the corresponding data frames are displayed.

If topology = TRUE, the plant structure is displayed; and if coordinates = TRUE, the spatial coordinates are displayed.

By default, the data frames containing the features on the vertices per class are printed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

[read.mtg](): reads a MTG file and creates an object of class "foldermtg".

Examples

```
mtgfile1 <- system.file("extdata/plant1.mtg", package = "dad")
xmtg1 <- read.mtg(mtgfile1)
print(xmtg1)
print(xmtg1, topology = TRUE)
print(xmtg1, coordinates = TRUE)

mtgfile2 <- system.file("extdata/plant2.mtg", package = "dad")
xmtg2 <- read.mtg(mtgfile2)
print(xmtg2)
print(xmtg2, topology = TRUE)
print(xmtg2, coordinates = TRUE)
```

---

print.foldert | *Printing an object of class* foldert

Description

print function, applied to an object of class "foldert", prints a foldert, as returned by foldert or as.foldert function.

Usage

```
## S3 method for class 'foldert'
print(x, ...)
```

Arguments

x   an object of class foldert.

...   optional arguments to print methods.

Details

The foldert is printed. In any data frame x[[t]] of this foldert, if a row is entirely NA (which means that the corresponding individual was not observed at time t), this row are not printed.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

foldert: object of class foldert. as.foldert.data.frame: build an object of class foldert from a data frame. as.foldert.array: build an object of class foldert from a $3d$-array.

Examples

```
data(floribundity)

ft <- foldert(floribundity, cols.select = "union", rows.select = "union")
print(ft)
```

---

print.fpcad | *Printing results of a functional PCA of probability densities*

Description

Applies to an object of class "fpcad". Prints the numeric results returned by the fpcad function.

Usage

```
## S3 method for class 'fpcad'
print(x, mean.print = FALSE, var.print = FALSE,
  cor.print = FALSE, skewness.print = FALSE, kurtosis.print = FALSE,
  digits = 2, ...)
```

Arguments

x	object of class "fpcad", returned by the fpcad function.
mean.print	logical. If TRUE, prints for each group the means and standard deviations of the variables and the norm of the density.
var.print	logical. If TRUE, prints for each group the variances and covariances of the variables.
cor.print	logical. If TRUE, prints for each group the correlations between the variables.
skewness.print	logical. If TRUE, prints for each group the skewness coefficients of the variables.
kurtosis.print	logical. If TRUE, prints for each group the kurtosis coefficients of the variables.
digits	numeric. Number of significant digits for the display of numeric results.
...	optional arguments to print methods.

Details

By default, are printed the inertia explained by the nb.values (see fpcad) first principal components, the contributions, the qualities of representation of the densities along the nb.factors (see fpcad) first principal components, and the principal scores.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcad; plot.fpcad; interpret.fpcad; print.

Examples

```
data(roses)
rosefold <- as.folder(roses[,c("Sha","Den","Sym","rose")])
result <- fpcad(rosefold)
print(result)
print(result, mean.print = TRUE)
```

---

print.fpcat	*Printing results of a functional PCA of probability densities among time*

Description

Applies to an object of class "fpcat". Prints the numeric results returned by the fpcat function.

Usage

```
## S3 method for class 'fpcat'
print(x, mean.print = FALSE, var.print = FALSE,
  cor.print = FALSE, skewness.print = FALSE, kurtosis.print = FALSE,
  digits = 2, ...)
```

Arguments

x	object of class "fpcat", returned by the fpcat function.
mean.print	logical. If TRUE, prints for each observation time the means and standard deviations of the variables and the norm of the density.
var.print	logical. If TRUE, prints for each observation time the variances and covariances of the variables.
cor.print	logical. If TRUE, prints for each observation time the correlations between the variables.
skewness.print	logical. If TRUE, prints for each observation time the skewness coefficients of the variables.
kurtosis.print	logical. If TRUE, prints for each observation time the kurtosis coefficients of the variables.
digits	numeric. Number of significant digits for the display of numeric results.
...	optional arguments to print methods.

Details

By default, are printed the vector of observation times (numeric, ordered factor or object of class "Date"), the inertia explained by the nb.values (see fpcat) first principal components, the contributions, the qualities of representation of the densities along the nb.factors (see fpcat) first principal components, and the principal scores.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Boumaza, R., Yousfi, S., Demotes-Mainard, S. (2015). Interpreting the principal component analysis of multivariate density functions. Communications in Statistics - Theory and Methods, 44 (16), 3321-3339.

See Also

fpcat; plot.fpcat; print.

Examples

```
times <- as.Date(c("2017-03-01", "2017-04-01", "2017-05-01", "2017-06-01"))
x1 <- data.frame(z1=rnorm(6,1,5), z2=rnorm(6,3,3))
x2 <- data.frame(z1=rnorm(6,4,6), z2=rnorm(6,5,2))
x3 <- data.frame(z1=rnorm(6,7,2), z2=rnorm(6,8,4))
x4 <- data.frame(z1=rnorm(6,9,3), z2=rnorm(6,10,2))
ft <- foldert(x1, x2, x3, x4, times = times, rows.select="intersect")
print(ft)
result <- fpcat(ft)

print(result)
print(result, mean.print = TRUE, var.print = TRUE)
```

---

print.hclustdd   *Printing results of a hierarchical clustering of discrete distributions*

Description

`print` function, applied to an object of class "hclustdd", prints numerical results of hclustdd .

Usage

```
## S3 method for class 'hclustdd'
print(x, dist.print=FALSE, digits=2, ...)
```

Arguments

x	object of class "hclustdd", returned by hclustdd.
dist.print	logical. If TRUE (default: FALSE), prints the matrix of distances between the groups (densities).
digits	numerical. Number of significant digits for the display of numerical results.
...	optional arguments to print methods.

Details

If dist.print = TRUE, the distances between groups are displayed.

By default, the result of the clustering is printed. The display is the same as that of the print.hclust function.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

hclustdd; plot.hclustdd.

Examples

```
data(dspg)
xl = dspg
result <- hclustdd(xl)
print(result)
print(result, dist.print = TRUE)
```

---

print.mdsdd	*Printing results of a multidimensional scaling analysis of discrete distributions*

Description

Applies to an object of class "mdsdd". Prints the numeric results returned by the mdsdd function.

Usage

```
## S3 method for class 'mdsdd'
print(x, joint = FALSE, margin1 = FALSE, margin2 = FALSE,
        association = FALSE, ...)
```

Arguments

x	object of class "mdsdd", returned by the mdsdd function.
joint	logical. If TRUE, prints for each group the table of estimated joint distribution.
margin1	logical. If TRUE, prints for each group the data frame of estimated marginal distributions.
margin2	logical. If TRUE, prints for each group the data frame of the estimated marginal distributions per combination of two variables.
association	logical. If TRUE, prints for each group the matrix of the pairwise association measures of the variables.
...	optional arguments to print methods.

Details

By default, are printed the inertia explained by the nb.values (see mdsdd) first coordinates and the nb.factors (see mdsdd) coordinates of the densities.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Sabine Demotes-Mainard

See Also

mdsdd; plot.mdsdd; interpret.mdsdd

Examples

```
# INSEE (France): Diploma x Socio professional group, seven years.
data(dspg)
xlista = dspg
a <- mdsdd(xlista)
print(a, joint = TRUE, margin1 = TRUE, margin2 = TRUE)
```

---

read.mtg   *Read a MTG (Multiscale Tree Graph) file*

Description

Reads an MTG (Multiscale Tree Graph) file and returns an object of class foldermtg, that is a list of data frames (see Details).

Usage

```
read.mtg(file, ...)
```

Arguments

file	character. Path of the MTG file.
...	optional arguments to print methods.

Details

Recalling that a MTG file is a text file that can be opened with a spreadsheet (Excel, LibreOffice-Calc...). Its 4 tables are:

- **CLASSES:** In this table the first column, named SYMBOL, contains the symbolic character denoting each botanical entity (or vertex class, plant component...) used in the MTG (for example, P for plant, A for axis...). The second column, named SCALE, represents the scale at which each entity appears in the MTG (for example 1 for P, 2 for axis...).

- **DESCRIPTION:** This table displays the relations between the vertices: + (branching relationship) or < (successor relationship).

- **FEATURES:** This table contains the features that can be attached to the vertices and their types: INT (integer), REAL (real numbers), STRING (character)...

- **MTG:** This table describes the plant topology, that is the vertices (one vertex per row) and their relations, the spatial coordinates of each vertex and the values taken by each vertex on the above listed features.

  Each vertex is labelled by its class, designating its botanical entity, and its index, designating its position among its immediate neighbours having the same scale. Each vertex label is preceded by + or <, seen above, or by the symbol / (decomposition relationship) that means that the corresponding vertex is the first vertex of the decomposition of the vertex which precedes /.

  Notice that the column number of a vertex matches with its branching order. The vertices of scale k resulting from the decomposition of a vertex of scale k-1, named parent vertex, have the same order as that of the parent vertex.

See the example below.

Value

read.mtg returns an object, say x, of class fodermtg, that is a list of at least 6 data frames:

classes	the table CLASSES: in the MTG file.
description	the table DESCRIPTION: in the MTG file.
features	the table FEATURES: in the MTG file.
topology	data frame containing the first columns of the "MTG:" table of the MTG file. If the maximum branching order of the elements of the MTG is p, then x$topology has p columns.
	If the i-th vertex appears on the j-th column, it means that its branching order is j, that is it belongs to a vertex of the j-th order.
coordinates	data frame of the spatial coordinates of the entities. It has six columns: XX, YY, ZZ (cartesian coordinates), AA, BB, CC (angle coordinates). If there are no coordinates in the MTG file, this data frame has 0 row.

The sixth and following elements are nclass data frames, nclass being the number of classes in the MTG file. Each data frame matches with a vertex class, such as "P" (plant), "A" (axes), "M" (metamers or phytomers), and contains the features on the corresponding vertices.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

print.foldermtg

mtgorder

Examples

```
mtgfile1 <- system.file("extdata/plant1.mtg", package = "dad")
x1 <- read.mtg(mtgfile1)
print(x1)

mtgfile2 <- system.file("extdata/plant2.mtg", package = "dad")
x2 <- read.mtg(mtgfile2)
print(x2)
```

---

rmcol.folder   *Remove columns in all elements of a folder*

Description

Remove some columns in all data frames of a folder.

Usage

```
rmcol.folder(object, name)
```

Arguments

object   object of class [folder](#) that is a list of data frames with the same column names.

name   character vector. The names of the columns to be removed in each data frame
                of the folder.

Value

A folder with the same number of elements as `object`. Its kth element is a data frame, and its
columns are the columns of `object[[k]]`, except those given by `name`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#): object of class `folder`.

[getcol.folder](#): select columns in all elements of a folder.

[getrow.folder](#): select rows in all elements of a folder.

[rmrow.folder](#): remove rows in all elements of a folder.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
rmcol.folder(iris.fold, c("Petal.Length", "Petal.Width"))
```

rmcol.foldert   *Remove cols in all elements of a foldert*

Description

Remove some columns in all data frames of a foldert.

Usage

```
rmcol.foldert(object, name)
```

Arguments

object	object of class [foldert](#) that is a list of data frames with the same column names, each of them corresponding to a time of observation.
name	character vector. The names of the columns to be removed in each data frame of the foldert.

Value

A foldert with the same number of elements as object. Its kth element is a data frame, and its columns are the columns of object[[k]], except those given by name.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](#): object of class foldert.

[getcol.foldert](#): select columns in all elements of a foldert.

[getrow.foldert](#): get rows in all elements of a foldert.

[rmrow.foldert](#): remove rows in all elements of a foldert.

Examples

```
data(floribundity)

ft0 <- foldert(floribundity, cols.select = "union", rows.select = "union")
ft0
rmcol.foldert(ft0, c("area"))
```

---

rmrow.folder *Remove rows in all elements of a folder*

Description

Remove some rows in all data frames of a folder.

Usage

```
rmrow.folder(object, name)
```

Arguments

object	object of class [folder](#) that is a list of data frames with the same column names.
name	character vector. The names of the rows to be removed in each data frame of the folder.

Value

A folder with the same number of elements as `object`. Its kth element is a data frame, and its rows are the rows of `object[[k]]`, except those given by `name`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#): object of class folder.

[getrow.folder](#): select rows in all elements of a folder.

[getcol.folder](#): select columns in all elements of a folder.

[rmcol.folder](#): remove columns in all elements of a folder.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
rmrow.folder(iris.fold, as.character(seq(1, 150, by = 2)))
```

---

rmrow.foldert *Remove rows in all elements of a foldert*

Description

Remove some rows in all data frames of a foldert.

Usage

```
rmrow.foldert(object, name)
```

Arguments

object   object of class [foldert](#) that is a list of data frames with the same column names,
            each of them corresponding to a time of observation.

name   character vector. The names of the rows to be removed in each data frame of the
            foldert.

Value

A foldert with the same number of elements as object. Its kth element is a data frame, and its rows
are the rows of object[[k]], except those given by name.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](#): object of class foldert.

[getrow.foldert](#): select rows in all elements of a foldert.

[getcol.foldert](#): select columns in all elements of a foldert.

[rmcol.foldert](#): remove columns in all elements of a foldert.

Examples

```
data(floribundity)

ft0 <- foldert(floribundity, cols.select = "union", rows.select = "union")
ft0
rmrow.foldert(ft0, c("rose", c("16", "51")))
```

---

roseflowers   *Rose flowers*

Description

The data are extracted from measures on roses from an agronomic experiment in a greenhouse and outdoors.

Usage

```
data(roseflowers)
```

Format

roseflowers is a list of two data frames:

roseflowers$variety: this first data frame has 5 rows and 3 columns (factors) named place, rose and variety.

roseflowers$flower: this second data frame has 11 cases and 5 columns named numflower (the order number of the flower), rose, diameter and height (the diameter and height of the flower), and nleaves (the number of the leaves of the axis).

Examples

```
data(roseflowers)
summary(roseflowers$variety)
summary(roseflowers$flower)
```

---

roseleaves   *Rose leaves*

Description

The data are extracted from measures on roses from an agronomic experiment in a greenhouse and outdoors.

Usage

```
data("roseleaves")
```

Format

roseleaves is a list of four data frames:

roseflowers$rose: data frame with 7 rows and 3 columns (factors) named rose, place and variety.

roseflowers$stem: data frame with 12 rows and 5 columns named rose, stem, date, order (the ramification order of the stem) and nleaves (the number of leaves of the stem).

roseflowers$leaf: data frame with 35 rows and 5 columns named stem, leaf, rank (the rank of the leaf on the stem), nleaflets and lrachis (the number of leaflets of the leaf and the length of its rachis).

roseflowers$leaflet: data frame with 221 rows and 4 columns named leaf, leaflet, lleaflet and wleaflet (the length and width of the leaflet).

Each row (rose) in roseleaves$rose pertains to several rows (stems) in roseleaves$stem.

Each row (stem) in roseleaves$rose pertains to several rows (leaves) in roseleaves$leaf.

Each row (leaf) in roseleaves$rose pertains to several rows (leaflets) in roseleaves$leaflet.

Examples

```
data(roseleaves)
summary(roseleaves$rose)
summary(roseleaves$stem)
summary(roseleaves$leaf)
summary(roseleaves$leaflet)
```

---

rosephytomer   *Rose leaf and internode dynamics*

Description

These data are extracted from measures on rosebushes during a study on leaf and internode expansion dynamics. For four rosebushes, on each metamer, the length of the terminal leaflet and the length of the internode were measured on several days, from the 24 april 2010 to the 19 july 2010.

The metamers which have no leaflets are omitted.

Usage

```
data("rosephytomer")
```

Format

A data frame with 643 rows (4 plants, 7, 8 or 9 metamers per plant, 37 days of observation) and 6 columns:

date a POSIXct

nplant a factor with levels 113 114 118 121. Numbers of the plants.

rank numeric. Rank of the metamer on the stem.

lleaflet, linternode numeric. Length of the terminal leaflet, length of the internode.

phytomer factor. Identifiers of the metamers.

Source

Demotes-Mainard, S., Bertheloot, J., Boumaza, R., Huché-Thélier, L., Guéritaine, G., Guérin, V. and Andrieu, B. (2013). Rose bush leaf and internode expansion dynamics: analysis and development of a model capturing interplant variability. Frontiers in Plant Science 4: 418. Doi: 10.3389/fpls.2013.00418

Examples

```
data(rosephytomer)
as.foldert(rosephytomer, method = 1, ind = "phytomer", timecol = "date", same.rows = TRUE)
```

---

roses *Roses data*

Description

Sensory data characterising the visual aspect of 10 rosebushes

Usage

```
data(roses)
```

Format

roses is a data frame of sensory data with 420 rows (10 products, 14 assessors, 3 sessions) and 17 columns. The first 16 columns are numeric and correspond to 16 visual characteristics of rosebushes. The last column is a factor giving the name of the corresponding rosebush.

Sha: top sided shape

Den: foliage thickness

Sym: plant symmetry

Vgr: stem vigour

Qrm: quantity of stems

Htr: branching level

Qfl: quantity of flowers

Efl: staggering of flowering

Mvfl: flower enhancement

Difl: flower size

Qfr: quantity of faded flowers/fruits

Qbt: quantity of floral buds

Defl: density of flower petals

Vcfl: intensity of flower colour

Tfe: leaf size

Vfe: darkness of leaf colour

rose: factor with 10 levels: A, B, C, D, E, F, G, H, I and J

Source

Boumaza, R., Huché-Thélier, L., Demotes-Mainard, S., Le Coz, E., Leduc, N., Pelleschi-Travier, S., Qannari, E.M., Sakr, S., Santagostini, P., Symoneaux, R., Guérin, V. (2010). Sensory profile and preference analysis in ornamental horticulture: The case of rosebush. Food Quality and Preference, 21, 987-997.

Examples

```
data(roses)
summary(roses)
```

---

skewness.folder   *Skewness coefficients of a folder of data sets*

Description

Computes the skewness coefficient by column of the elements of an object of class folder.

Usage

```
skewness.folder(x, na.rm = FALSE, type = 3)
```

Arguments

x	an object of class [folder](#) that is a list of data frames with the same column names.
na.rm	logical. Should missing values be omitted from the calculations? (see [skewness](#))
type	an integer between 1 and 3 (see [skewness](#)).

Details

It uses [skewness](#) to compute the mean by numeric column of each element of the folder. If some columns of the data frames are not numeric, there is a warning, and the means are computed on the numeric columns only.

Value

A list whose elements are the skewness coefficients by column of the elements of the folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder to create an object is of class folder. mean.folder, var.folder, cor.folder, kurtosis.folder for other statistics for folder objects.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
iris.skewness <- skewness.folder(iris.fold)
print(iris.skewness)

# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
roses.skewness <- skewness.folder(roses.fold)
print(roses.skewness)
```

---

sqrtmatrix	*Square root of a symmetric, positive semi-definite matrix*

Description

Calculation of the square root of a positive semi-definite matrix (see Details for the definition of such a matrix).

Usage

```
sqrtmatrix(mat)
```

Arguments

mat   numeric matrix.

Details

The matrix mat must be symmetric and positive semi-definite. Otherwise, there is an error.

The square root of the matrix mat is the positive semi-definite matrix M such as t(M) %*% M = mat. Do not confuse with sqrt(mat), which returns the square root of the elements of mat.

The computation is based on the diagonalisation of mat. The eigenvalues smaller than 10^-16 are identified as null values.

Value

Matrix: the square root of the matrix `mat`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

Examples

```
M2 <- matrix(c(5, 4, 4, 5), nrow = 2)
M <- sqrtmatrix(M2)
M
```

---

summary.folder   *Summarize a folder*

Description

Summarize an object of class `folder`.

Usage

```
## S3 method for class 'folder'
summary(object, ...)
```

Arguments

object	object of class [folder](#) that is a list of data frames with the same column names.
...	further arguments passed to or from other methods.

Value

A list, each element of it contains the summary of the corresponding element of `object`. This list has an attribute `attr(, "same.rows")`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folder](#): object of class folder. [as.folder.data.frame](#): build an object of class folder from a data frame.

Examples

```
data(iris)

iris.fold <- as.folder(iris, "Species")
summary(iris.fold)
```

summary.folderh   *Summarize a folderh*

Description

Summarize an object of class `folderh`.

Usage

```
## S3 method for class 'folderh'
summary(object, ...)
```

Arguments

object	object of class [folderh](#) that is a list of data frames.
...	further arguments passed to or from other methods.

Value

A list, each element of it containing the summary of the corresponding element of `object`. This list has an attribute `attr(, "keys")` (see [folderh](#)).

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[folderh](#): object of class folderh.

Examples

```
# First example
mtgfile <- system.file("extdata/plant1.mtg", package = "dad")
x <- read.mtg(mtgfile)
fh1 <- as.folderh(x, classes = c("P", "A", "M"))
summary(fh1)

# Second example
data(roseleaves)
roses <- roseleaves$rose
stems <- roseleaves$stem
leaves <- roseleaves$leaf
leaflets <- roseleaves$leaflet
fh2 <- folderh(roses, "rose", stems, "stem", leaves, "leaf", leaflets)
summary(fh2)
```

summary.foldermtg   *Summary of an object of class* foldermtg

---

Description

Summary method for S3 class foldermtg.

Usage

```
## S3 method for class 'foldermtg'
summary(object, ...)
```

Arguments

object	an object of class foldermtg.
...	optional arguments to summary methods.

Value

The summary of the data frames containing the vertices of each class and the values of the features on these vertices.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Pradal, C., Godin, C. and Cokelaer, T. (2023). MTG user guide

See Also

read.mtg: reads a MTG file and creates an object of class "foldermtg".

Examples

```
mtgfile1 <- system.file("extdata/plant1.mtg", package = "dad")
x1 <- read.mtg(mtgfile1)
summary(x1)

mtgfile2 <- system.file("extdata/plant2.mtg", package = "dad")
x2 <- read.mtg(mtgfile2)
summary(x2)
```

summary.foldert   *Summarize a foldert*

Description

Summarize an object of class `foldert`.

Usage

```
## S3 method for class 'foldert'
summary(object, ...)
```

Arguments

object	object of class [foldert](#) that is a list of data frames organised according to time.
...	further arguments passed to or from other methods.

Value

A list, each element of it contains the summary of the corresponding element of `object`. This list has two attributes `attr(, "times")` and `attr(, "same.rows")`.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

[foldert](#): object of class foldert. [as.foldert.data.frame](#): build an object of class foldert from a data frame. [as.foldert.array](#): build an object of class foldert from a $3d$-array.

Examples

```
# 1st example
data(floribundity)
ftflor <- foldert(floribundity, cols.select = "union", rows.select = "union")
summary(ftflor)
```

---

var.folder   *Variance matrices of a folder of data sets*

Description

Computes the variance matrices of the elements of an object of class folder.

Usage

```
var.folder(x, na.rm = FALSE, use = "everything")
```

Arguments

x	an object of class folder that is a list of data frames with the same column names.
na.rm	logical. Should missing values be removed? (see [var](var))
use	an optional character string giving a method for computing covariances in the presence of missing values. This must be (an abbreviation of) one of the strings "everything", "all.obs", "complete.obs", "na.or.complete", or "pairwise.complete.obs" (see [var](var)).

Details

It uses [var](var) to compute the variance matrix of the numeric columns of each element of the folder. If some columns of the data frames are not numeric, there is a warning, and the variances are computed on the numeric columns only.

Value

A list whose elements are the variance matrices of the elements of the folder.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

See Also

folder to create an object is of class folder. mean.folder, cor.folder, skewness.folder, kurtosis.folder for other statistics for folder objects.

Examples

```
# First example: iris (Fisher)
data(iris)
iris.fold <- as.folder(iris, "Species")
iris.vars <- var.folder(iris.fold)
print(iris.vars)
```

```
# Second example: roses
data(roses)
roses.fold <- as.folder(roses, "rose")
roses.vars <- var.folder(roses.fold)
print(roses.vars)
```

---

varietyleaves *Rose variety leaves*

Description

The data are extracted from measures on roses from an agronomic experiment in a greenhouse and outdoors.

Usage

```
data("varietyleaves")
```

Format

varietyleaves is an object of class "folderh", that is a list of two data frames:

varietyleaves$variety: data frame with 31 rows and 2 columns (factors) named rose and variety.

varietyleaves$leaves: data frame with 581 rows and 5 columns named rose, nleaflet (number of leaflets), lrachis (length of the rachis), lleaflet (length of the principal leaflet) and wleaflet (width of the principal leaflet).

Examples

```
data(varietyleaves)
summary(varietyleaves)
```

---

wasserstein *2-Wasserstein distance between Gaussian densities*

Description

The 2-Wasserstein distance between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities (see Details).

Usage

```
wasserstein(x1, x2, check = FALSE)
```

Arguments

x1	a matrix or data frame of n_1 rows (observations) and p columns (variables) (can also be a tibble) or a vector of length n_1.
x2	matrix or data frame (or tibble) of n_2 rows and p columns or vector of length n_2.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

The Wasserstein distance between the two Gaussian densities is computed by using the wassersteinpar function and the density parameters estimated from samples.

Value

Returns the 2-$Wasserstein$ distance between the two probability densities.

Be careful! If check = FALSE and one smoothing bandwidth matrix is degenerate, the result returned can not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Peterson, A., Mueller, H.G. (2016). Functional Data Analysis for Density Functions by Transformation to a Hilbert Space. The annals of Statistics, 44 (1), 183-218. DOI: 10.1214/15-AOS1363

Dowson, D.C., Ladau, B.V. (1982). The Fréchet Distance between Multivariate Normal Distributions. Journal of Multivariate Analysis, 12, 450-455.

See Also

wassersteinpar: 2-Wasserstein distance between Gaussian densities, given their parameters.

Examples

```
require(MASS)
m1 <- c(0,0)
v1 <- matrix(c(1,0,0,1),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(4,1,1,9),ncol = 2)
x1 <- mvrnorm(n = 3,mu = m1,Sigma = v1)
x2 <- mvrnorm(n = 5, mu = m2, Sigma = v2)
wasserstein(x1, x2)
```

| wassersteinpar | *2-Wasserstein distance between Gaussian densities given their parameters* |

Description

The 2-Wasserstein distance between two multivariate ($p > 1$) or univariate ($p = 1$) Gaussian densities given their parameters (mean vectors and covariance matrices if the densities are multivariate, or means and variances if univariate) (see Details).

Usage

```
wassersteinpar(mean1, var1, mean2, var2, check = FALSE)
```

Arguments

mean1	p-length numeric vector: the mean of the first Gaussian density.
var1	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the first Gaussian density.
mean2	p-length numeric vector: the mean of the second Gaussian density.
var2	p x p symmetric numeric matrix ($p > 1$) or numeric ($p = 1$): the covariance matrix ($p > 1$) or the variance ($p = 1$) of the second Gaussian density.
check	logical. When TRUE (the default is FALSE) the function checks if the covariance matrices are not degenerate (multivariate case) or if the variances are not zero (univariate case).

Details

The mean vectors ($m1$ and $m2$) and variance matrices ($v1$ and $v2$) given as arguments (mean1, mean2, var1 and var2) are used to compute the 2-Wasserstein distance between the two Gaussian densities, equal to:

$$(||m1 - m2||_2^2 + trace((v1 + v2) - 2 * (v2^{1/2}v1v2^{1/2})^{1/2}))^{1/2}$$

If $p = 1$:

$$((m1 - m2)^2 + v1 + v2 - 2 * (v1 * v2)^{1/2})^{1/2}$$

Value

The 2-Wasserstein distance between two Gaussian densities.

Be careful! If check = FALSE and one covariance matrix is degenerated (multivariate case) or one variance is zero (univariate case), the result returned must not be considered.

Author(s)

Rachid Boumaza, Pierre Santagostini, Smail Yousfi, Gilles Hunault, Sabine Demotes-Mainard

References

Peterson, A., Mueller, H.G (2016). Functional Data Analysis for Density Functions by Transformation to a Hilbert Space. The annals of Statistics, 44 (1), 183-218. DOI: 10.1214/15-AOS1363

Dowson, D.C., Ladau, B.V. (1982). The Fréchet Distance between Multivariate Normal Distributions. Journal of Multivariate Analysis, 12, 450-455.

See Also

wasserstein: 2-Wasserstein distance between Gaussian densities estimated from samples.

Examples

```
m1 <- c(1,1)
v1 <- matrix(c(4,1,1,9),ncol = 2)
m2 <- c(0,1)
v2 <- matrix(c(1,0,0,1),ncol = 2)
wassersteinpar(m1,v1,m2,v2)
```

Index