Package ‘HistDAWass’

June 22, 2021

Type Package
Title Histogram-Valued Data Analysis
Version 1.0.6
Date 2021-06-22
Maintainer Antonio Irpino <antonio.irpino@unicampania.it>
Description In the framework of Symbolic Data Analysis, a relatively new approach to the statistical analysis of multi-valued data, we consider histogram-valued data, i.e., data described by univariate histograms. The methods and the basic statistics for histogram-valued data are mainly based on the L2 Wasserstein metric between distributions, i.e., the Euclidean metric between quantile functions. The package contains unsupervised classification techniques, least square regression and tools for histogram-valued data and for histogram time series. An introducing paper is Irpino A. Verde R. (2015) <doi:10.1007/s11634-014-0176-4>.
License GPL (>= 2)
Imports graphics, class, FactoMineR, ggplot2, ggridges, grid, histogram, grDevices, stats, utils, Rcpp
Depends R(>= 3.1), methods
LazyData true
Encoding UTF-8
RoxygenNote 7.1.1
NeedsCompilation yes
LinkingTo Rcpp,RcppArmadillo
Author Antonio Irpino [aut, cre] (<https://orcid.org/0000-0001-9293-7180>)
Repository CRAN
Date/Publication 2021-06-22 17:20:02 UTC
### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>HistDAWass-package</td>
<td>4</td>
</tr>
<tr>
<td>-<em>methods</em></td>
<td>5</td>
</tr>
<tr>
<td>+</td>
<td>6</td>
</tr>
<tr>
<td>Age_Pyramids_2014</td>
<td>6</td>
</tr>
<tr>
<td>Agronomique</td>
<td>7</td>
</tr>
<tr>
<td>BLOOD</td>
<td>7</td>
</tr>
<tr>
<td>BloodBRITO</td>
<td>8</td>
</tr>
<tr>
<td>Center.cell.MatH</td>
<td>8</td>
</tr>
<tr>
<td>checkEmptyBins</td>
<td>9</td>
</tr>
<tr>
<td>China_Month</td>
<td>10</td>
</tr>
<tr>
<td>China_Sea</td>
<td>10</td>
</tr>
<tr>
<td>compP</td>
<td>11</td>
</tr>
<tr>
<td>compQ</td>
<td>11</td>
</tr>
<tr>
<td>crwtransform</td>
<td>12</td>
</tr>
<tr>
<td>data2hist</td>
<td>13</td>
</tr>
<tr>
<td>distributionH-class</td>
<td>14</td>
</tr>
<tr>
<td>dotpW</td>
<td>16</td>
</tr>
<tr>
<td>DouglasPeucker</td>
<td>17</td>
</tr>
<tr>
<td>get.cell.MatH</td>
<td>18</td>
</tr>
<tr>
<td>get.distr</td>
<td>18</td>
</tr>
<tr>
<td>get.histo</td>
<td>19</td>
</tr>
<tr>
<td>get.m</td>
<td>20</td>
</tr>
<tr>
<td>get.MatH.main.info</td>
<td>20</td>
</tr>
<tr>
<td>get.MatH.ncols</td>
<td>21</td>
</tr>
<tr>
<td>get.MatH.nrows</td>
<td>21</td>
</tr>
<tr>
<td>get.MatH.rownames</td>
<td>22</td>
</tr>
<tr>
<td>get.MatH.stats</td>
<td>22</td>
</tr>
<tr>
<td>get.MatH.varnames</td>
<td>24</td>
</tr>
<tr>
<td>get.s</td>
<td>24</td>
</tr>
<tr>
<td>HTS-class</td>
<td>25</td>
</tr>
<tr>
<td>HTS.exponential.smoothing</td>
<td>25</td>
</tr>
<tr>
<td>HTS.moving.averages</td>
<td>26</td>
</tr>
<tr>
<td>HTS.predict.knn</td>
<td>27</td>
</tr>
<tr>
<td>is.registeredMH</td>
<td>28</td>
</tr>
<tr>
<td>kurtH</td>
<td>29</td>
</tr>
<tr>
<td>MatH-class</td>
<td>30</td>
</tr>
<tr>
<td>meanH</td>
<td>31</td>
</tr>
<tr>
<td>minus</td>
<td>32</td>
</tr>
<tr>
<td>OzoneFull</td>
<td>33</td>
</tr>
<tr>
<td>OzoneH</td>
<td>33</td>
</tr>
<tr>
<td>plot-distributionH</td>
<td>34</td>
</tr>
<tr>
<td>plot-HTS</td>
<td>34</td>
</tr>
<tr>
<td>plot-MatH</td>
<td>35</td>
</tr>
<tr>
<td>plot-TdistributionH</td>
<td>36</td>
</tr>
<tr>
<td>plotPredVsObs</td>
<td>37</td>
</tr>
<tr>
<td>plot_errors</td>
<td>38</td>
</tr>
</tbody>
</table>
R topics documented:

- register
- registerMH
- RetHTS
- rQQ
- set.cell.MatH
- ShortestDistance
- show
- show-MatH
- skewH
- stations_coordinates
- stdH
- subsetHTS
- TdistributionH-class
- TMatH-class
- WassSqDistH
- WH.1d.PCA
- WH.bind
- WH.bind.col
- WH.bind_row
- WH.correlation
- WH.correlation2
- WH.mat.prod
- WH.mat.sum
- WH.MultiplePCA
- WH.plot_multiple_indivs
- WH.plot_multiple_Spanish.funs
- WH.regression.GOF
- WH.regression.two.components
- WH.regression.two.components.predict
- WH.SSQ
- WH.SSQ2
- WH.var.covar
- WH.var.covar2
- WH.vec.mean
- WH.vec.sum
- WH_2d_Adaptive_Kohonen_maps
- WH_2d_Kohonen_maps
- WH_adaptive.kmeans
- WH_adaptive_fcmens
- WH_fcmens
- WH_hclust
- WH_hclust
- WH_kmeans
- WH_MAT_DIST

Index
Description

We consider histogram-valued data, i.e., data described by univariate histograms. The methods and the basic statistics for histogram-valued data are mainly based on the L2 Wasserstein metric between distributions, i.e., a Euclidean metric between quantile functions. The package contains unsupervised classification techniques, least square regression and tools for histogram-valued data and for histogram time series.

Details

Package: HistDAWass
Type: Package
Version: 0.1.1
Date: 2014-09-17
License: GPL (>=2)
Depends: methods

~ An overview of how to use the package, including the most important functions ~

Author(s)

Antonio Irpino <antonio.irpino@unicampania.it>

References


Examples

# Generating a list of distributions
a <- vector("list", 4)
a[[1]] <- distributionH(
  x = c(80, 100, 120, 135, 150, 165, 180, 200, 240),
  p = c(0, 0.025, 0.1, 0.275, 0.525, 0.725, 0.887, 0.975, 1)
)
a[[2]] <- distributionH(
  x = c(80, 100, 120, 135, 150, 165, 180, 195, 210, 240),
  p = c(0, 0.013, 0.101, 0.255, 0.508, 0.718, 0.895, 0.961, 0.987, 1)
)
a[[3]] <- distributionH(
  x = c(95, 110, 125, 140, 155, 170, 185, 200, 215, 230, 245),
  p = c(0, 0.012, 0.041, 0.154, 0.36, 0.595, 0.781, 0.929, 0.972, 0.992, 1)
)
a[[4]] <- distributionH(
  x = c(105, 120, 135, 150, 165, 180, 195, 210, 225, 240, 260),
  p = c(0, 0.009, 0.035, 0.081, 0.186, 0.385, 0.633, 0.832, 0.932, 0.977, 1)
)

# Generating a list of names of observations
namerows <- list("u1", "u2")
# Generating a list of names of variables
namevars <- list("Var_1", "Var_2")
# creating the MatH
Mat_of_distributions <- MatH(
  x = a, nrows = 2, ncols = 2,
  rownames = namerows, varnames = namevars, by.row = FALSE
)

**Methods**

**Method**

<table>
<thead>
<tr>
<th>Description</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>the product of a number and a distribution according to the L2 Wasssertein</td>
<td><code>e1 * e2</code></td>
</tr>
<tr>
<td>the product of a number and a distribution according to the L2 Wasssertein</td>
<td><code>e1 * e2</code></td>
</tr>
<tr>
<td>the product of a number and a distribution according to the L2 Wasssertein</td>
<td><code>e1 * e2</code></td>
</tr>
</tbody>
</table>

**Arguments**

<table>
<thead>
<tr>
<th>e1</th>
<th>a distributionH object or a number</th>
</tr>
</thead>
<tbody>
<tr>
<td>e2</td>
<td>a distributionH object or a number</td>
</tr>
</tbody>
</table>
Description

the sum of two distribution according to the L2 Wasserstein
the sum of a number and a distribution according to the L2 Wasserstein
the sum of a distribution and a number according to the L2 Wasserstein

Usage

```r
## S4 method for signature 'distributionH,distributionH'
e1 + e2
```

```r
## S4 method for signature 'numeric,distributionH'
e1 + e2
```

```r
## S4 method for signature 'distributionH,numeric'
e1 + e2
```

Arguments

e1 a `distributionH` object or a number
e2 a `distributionH` object or a number

Value

a `distributionH` object

---

**Age_Pyramids_2014**

**Age pyramids of all the countries of the World in 2014**

Description

The dataset contains a `MatH` (matrix of histogram-valued data) object, with three histogram-valued variables, the 5-years age (relative frequencies) distribution of all the population, of the male and of the female population of 228 countries of the World. The first row is the World data. Thus it contains 229 rows (228 countries plus the World) and 3 variables: "Both.Sexes.Population", "Male.Population", "Female.Population"

Format

a `MatH` object, a matrix of distributions.
**Agronomique**

**Author(s)**
Antonio Irpino, 2014-10-05

**Source**
United States Census Bureau [https://www.census.gov/data.html](https://www.census.gov/data.html)

**Description**
A dataset with the distributions of marginal costs of farms in 22 France regions. It contains four histogram variables: "Y_TSC" (Total costs of a farm), "X_Wheat" (Costs for Wheat), "X_Pig" (Costs for Pigs) "X_Cmilk" (Costs for Cow Milk)

**Format**
a MatH object, a matrix of distributions.

**Author(s)**
Antonio Irpino, 2014-10-05

**Source**
Rosanna Verde, Antonio Irpino, Second University of Naples; Dominique Desbois, UMR Economie publique, INRA-AgroParisTech, How to cope with modelling and privacy concerns? A regression model and a visualization tool for aggregated data, Conference of European Statistics Stakeholders, Rome, November, 24-25, 2014

**Blood**

**Blood dataset for Histogram data analysis**

**Description**
The dataset contains a MatH (matrix of histogram-valued data) object This data set list 14 groups of patients described by 3 variables.

**Format**
a MatH istance, 1 row per group.

**Author(s)**
Antonio Irpino, 2014-10-05
Source

BloodBRITO                 Blood dataset from Brito P. for Histogram data analysis

Description
The dataset contains a MatH (matrix of histogram-valued data) object. This data set lists 10 patients described by 2 variables.

Format
a MatH instance, 1 row per patient.

Author(s)
Antonio Irpino, 2014-10-05

Source

Center.cell.MatH            Method Center.cell.MatH Centers all the cells of a matrix of distributions

Description
The function centers a MatH object (i.e. a matrix of distributions), such that each distribution is shifted and has a mean equal to zero.

Usage
Center.cell.MatH(object)

## S4 method for signature 'MatH'
Center.cell.MatH(object)

Arguments
object         a MatH object, a matrix of distributions.
checkEmptyBins

Value

A MatH object, having each distribution with a zero mean.

Examples

CEN_BLOOD <- Center.cell.MatH(BLOOD)
get.MatH.stats(BLOOD, stat = "mean")

checkEmptyBins Method checkEmptyBins

Description

The method checking for empty bins in a distribution, i.e. if two cdf consecutive values are equal. In that case a probability value of $1e^{-7}$ is assigned to the empty bin and the cdf is recomputed. This methods is useful for numerical reasons.

Usage

checkEmptyBins(object)

## S4 method for signature 'distributionH'
checkEmptyBins(object)

Arguments

object a distributionH object

Value

A distributionH object without empty bins

Author(s)

Antonio Irpino

Examples

## ---- A mydist distribution with an empty bin i.e. two consecutive values of p are equal----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.5, 0.5, 1))
## ---- Checks for empty byns and returns the newdist object without empty bins ----
newdist <- checkEmptyBins(mydist)
China_Month

*A monthly climatic dataset of China*

**Description**

A dataset with the distributions of some climatic variables collected for each month in 60 stations of China. The collected variables are 168 i.e. 14 climatic variables observed for 12 months. The 14 variables are the following: mean station pressure (mb), mean temperature, mean maximum temperature, mean minimum temperature, total precipitation (mm), sunshine duration (h), mean cloud amount (percentage of sky cover), mean relative humidity (mean wind speed (m/s), dominant wind frequency (extreme minimum temperature. Use the command get.MatH.main.info(China_Month) for rapid info.

**Format**

A `MatH` object, a matrix of distributions.

**Author(s)**

Antonio Irpino, 2014-10-05

**Source**

raw data are available here: [https://cdiac.ess-dive.lbl.gov/ftp/tr055/](https://cdiac.ess-dive.lbl.gov/ftp/tr055/)

China_Seas

*A seasonal climatic dataset of China*

**Description**

A dataset with the distributions of some climatic variables collected for each season in 60 stations of China. The collected variables are 56 i.e. 14 climatic variables observed for 4 seasons. The 14 variables are the following: mean station pressure (mb), mean temperature, mean maximum temperature, mean minimum temperature, total precipitation (mm), sunshine duration (h), mean cloud amount (percentage of sky cover), mean relative humidity (mean wind speed (m/s), dominant wind frequency (extreme minimum temperature. Use the command get.MatH.main.info(China_Seas) for rapid info.

**Format**

A `MatH` object, a matrix of distributions.

**Author(s)**

Antonio Irpino, 2014-10-05
Source


compP  Method compP

Description

Compute the cdf probability at a given value for a histogram

Usage

compP(object, q)

## S4 method for signature 'distributionH,numeric'
compP(object, q)

Arguments

object is an object of distributionH class
q is a numeric value

Value

Returns a value between 0 and 1.

Examples

## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the cfd value for q=5 (not observed) ----
p <- compP(mydist, 5)

compQ  Method compQ

Description

Compute the quantile value of a histogram for a given probability.
Usage

compQ(object, p)

## S4 method for signature 'distributionH,numeric'
compQ(object, p)

Arguments

object an object of distributionH class
p a number between 0 and 1

Value

\[ y = F^{-1}(p) = Q(p) \]

A number that is the quantile of the passed histogram object at level p.

Author(s)

Antonio Irpino

Examples

```r
## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the quantile of mydist for different values of p ----
y <- compQ(mydist, 0.5) # the median
y <- compQ(mydist, 0) # the minimum
y <- compQ(mydist, 1) # the maximum
y <- compQ(mydist, 0.25) # the first quartile
y <- compQ(mydist, 0.9) # the ninth decile
```
Arguments

object a distributionH object

Value

A list containing

$Centers  The midpoints of the bins of the histogram
$Radii  The half-lengths of the bins of the histogram
$Weights  The relative frequencies or the probabilities associated with each bin (the sum is equal to 1)

Author(s)

Antonio Irpino

References


Examples

```r
## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the cfd value for q=5 (not observed) ----
crwtransform(mydist)
```

Description

From real data to distributionH.

Usage

data2hist(
  data,
  algo = "histogram",
  type = "combined",
  qua = 10,
  breaks = numeric(0),
  epsilon = 0.01
)
distributionH-class

Arguments

data a set of numeric values.
algo (optional) a string. Default is "histogram", i.e. the function "histogram" defined in the `histogram` package.
If "base" the `hist` function is used.
"FixedQuantiles" computes the histogram using as breaks a fixed number of quantiles.
"ManualBreaks" computes a histogram where breaks are provided as a vector of values.
An epsilon parameter is required. The data are scaled in order to have a standard deviation equal to one.
type (optional) a string. Default is "combined" and generates a histogram having regularly spaced breaks (i.e., equi-width bins) and irregularly spaced ones. The choice is done accordingly with the penalization method described in `histogram`.
"regular" returns equi-width binned histograms, "irregular" returns a histogram without equi-width histograms.
qua a positive integer to provide if algo="FixedQuantiles" is chosen. Default=10.
breaks a vector of values to provide if algo="ManualBreaks" is chosen.
epsilon a number between 0 and 1 to provide if algo="PolyLine" is chosen. Default=0.01.

Value

A `distributionH` object, i.e. a distribution.

See Also

`histogram` function

Examples

```r
data <- rnorm(n = 1000, mean = 2, sd = 3)
mydist <- data2hist(data)
plot(mydist)
```

distributionH-class Class `distributionH`

Description

Class "distributionH" defines an histogram object. The class describes a histogram by means of its cumulative distribution function. The methods are developed accordingly to the L2 Wasserstein distance between distributions.

A histogram object can be created also with the function `distributionH(...)`, the constructor function for creating an object containing the description of a histogram.
Usage

## S4 method for signature 'distributionH'
initialize(
  .Object,
  x = numeric(0),
  p = numeric(0),
  m = numeric(0),
  s = numeric(0)
)

distributionH(x = numeric(0), p = numeric(0))

Arguments

/Object/ the type ("distributionH")
/x/ a numeric vector. It is the domain of the distribution (i.e. the extremes of bins).
/p/ a numeric vector (of the same length of x). It is the cumulative distribution function CDF.
/m/ (optional) a numeric value. It is the mean of the histogram.
/s/ (optional) a numeric positive value. It is the standard deviation of a histogram.

Details

Class distributionH defines a histogram object

Value

A distributionH object

Objects from the Class

Objects can be created by calls of the form new("distributionH",x,p,m,s).

Author(s)

Antonio Irpino

References


See Also

meanH computes the mean. stdH computes the standard deviation.
Examples

--- initialize a distributionH object mydist
from a simple histogram

<table>
<thead>
<tr>
<th>Bins</th>
<th>Prob</th>
<th>cdf</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,2)</td>
<td>0.4</td>
<td>0.4</td>
</tr>
<tr>
<td>[2,3]</td>
<td>0.6</td>
<td>1.0</td>
</tr>
<tr>
<td>Tot.</td>
<td>1.0</td>
<td>-</td>
</tr>
</tbody>
</table>

mydist <- new("distributionH", c(1, 2, 3), c(0, 0.4, 1))

str(mydist)

mydist <- distributionH(x = c(1, 2, 3), p = c(0, 0.4, 1))

dotpW

**Method dotpW**

**Description**

The dot product of two distributions inducing the L2 Wasserstein metric
The dot product of a number (considered as an impulse distribution function) and a distribution
The dot product of a distribution and a number (considered as an impulse distribution function).

**Usage**

dotpW(e1, e2)

**Arguments**

- e1 a distributionH object or a number
- e2 a distributionH object or a number
DouglasPeucker

Value

A numeric value

Author(s)

Antonio Irpino

References


Examples

```r
## let's define two distributionH objects
mydist1 <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
mydist2 <- distributionH(x = c(5, 7, 15), p = c(0, 0.7, 1))

## the dot product between the distributions
dotpW(mydist1, mydist2) #---> 39.51429

## the dot product between a distribution and a numeric
dotpW(mydist1, 3) #---> 13.2
dotpW(3, mydist1) #---> 13.2
```

# DOTPW method -----

DouglasPeucker

Ramer-Douglas-Peucker algorithm for curve fitting with a PolyLine

Description

Ramer-Douglas-Peucker algorithm for curve fitting with a PolyLine

Usage

DouglasPeucker(points, epsilon)

Arguments

points a 2D matrix with the coordinates of 2D points
epsilon an number between 0 and 1. Recommended 0.01.

Value

A matrix with the points of segments of a Poly Line.
See Also

`data2hist` function

---

**get.cell.MatH**

Method `get.cell.MatH` Returns the histogram in a cell of a matrix of distributions

**Description**

Returns the histogram data in the r-th row and the c-th column.

**Usage**

```r
get.cell.MatH(object, r, c)
```

## S4 method for signature 'MatH,numeric,numeric'

get.cell.MatH(object, r, c)

**Arguments**

- **object**: a `MatH` object, a matrix of distributions.
- **r**: an integer, the row index.
- **c**: an integer, the column index

**Value**

A `distributionH` object.

**Examples**

```r
get.cell.MatH(BLOOD, r = 1, c = 1)
```

---

**get.distr**

Method `get.distr`: show the distribution

**Description**

This function return the cumulative distribution function of a `distributionH` object.

**Usage**

```r
get.distr(object)
```

## S4 method for signature 'distributionH'

get.distr(object)
get.histo

Arguments

- object: a distributionH object.

Value

A data frame: the first column contains the domain the second the CDF values.

Examples

```r
D <- distributionH(x = c(1, 2, 3, 4), p = c(0, 0.2, 0.6, 1))
get.distr(D) # a data.frame describing the CDF of D
```

---

**get.histo**  
*Method get.histo: show the distribution with bins*

Description

This function return a data.frame describing the histogram of a distributionH object.

Usage

```r
get.histo(object)
```

# S4 method for signature 'distributionH'
```r
get.histo(object)
```

Arguments

- object: a distributionH object.

Value

A matrix: the two columns contains the bounds of the histogram the third contains the probability (or the relative frequency) of the bin.

Examples

```r
D <- distributionH(x = c(1, 2, 3, 4), p = c(0, 0.2, 0.6, 1))
get.histo(D) # returns the histogram representation of D by a data.frame
```
get.m

Method get.m: the mean of a distribution

Description
This function returns the mean of a distribution object.

Usage
get.m(object)

## S4 method for signature 'distributionH'
get.m(object)

Arguments
object a distribution object

Value
A numeric value

Examples
D <- distributionH(x = c(1, 2, 3, 4), p = c(0, 0.2, 0.6, 1))
get.m(D) # returns the mean of D

get.MatH.main.info

Method get.MatH.main.info

Description
It returns the number of rows, of columns the labels of rows and columns of a MatH object.

Usage
get.MatH.main.info(object)

## S4 method for signature 'MatH'
get.MatH.main.info(object)

Arguments
object a MatH object
get.MatH.ncols

Value
A list of char, the labels of the columns, or the names of the variables.

Slots
nrows - the number of rows
ncols - the number of columns
rownames - a vector of char, the names of rows
varnames - a vector of char, the names of columns

description
It returns the number of columns of a MatH object

Usage
get.MatH.ncols(object)

## S4 method for signature 'MatH'
get.MatH.ncols(object)

Arguments
object a MatH object

Value
An integer, the number of columns.

description
It returns the number of rows of a MatH object

Usage
## S4 method for signature 'MatH'
get.MatH.nrows(object)

get.MatH.nrows

Method get.MatH.nrows
Arguments

object a MatH object

Value

An integer, the number of rows.

get.MatH.rownames

Method get.MatH.rownames

Description

It returns the labels of the rows of a MatH object

Usage

get.MatH.rownames(object)

## S4 method for signature 'MatH'
get.MatH.rownames(object)

Arguments

object a MatH object

Value

A vector of char, the label of the rows.

get.MatH.stats

Method get.MatH.stats

Description

It returns statistics for each distribution contained in a MatH object.

Usage

get.MatH.stats(object, ...)

## S4 method for signature 'MatH'
get.MatH.stats(object, stat = "mean", prob = 0.5)
get.MatH.stats

Arguments

  object  a MatH object
  ...  a set of other parameters
  stat  (optional) a string containing the required statistic. Default='mean'
   - stat='mean' - for computing the mean of each histogram
   - stat='median' - for computing the median of each histogram
   - stat='min' - for computing the minimum of each histogram
   - stat='max' - for computing the maximum of each histogram
   - stat='std' - for computing the standard deviation of each histogram
   - stat='skewness' - for computing the skewness of each histogram
   - stat='kurtosis' - for computing the kurtosis of each histogram
   - stat='quantile' - for computing the quantile at level prob of each histogram

  prob  (optional) a number between 0 and 1 for computing the value once chosen the 'quantile' option for stat.

Value

A list

Slots

  stat  - the chosen statistic
  prob  - level of probability if stat='quantile'
  MAT  - a matrix of values

Examples

get.MatH.stats(BLOOD) # the means of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "median") # the medians of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "quantile", prob = 0.5) # the same as median
get.MatH.stats(BLOOD, stat = "min") # minima of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "quantile", prob = 0) # the same as min
get.MatH.stats(BLOOD, stat = "max") # maxima of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "quantile", prob = 1) # the same as max
get.MatH.stats(BLOOD, stat = "std") # standard deviations of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "skewness") # skewness indices of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "kurtosis") # kurtosis indices of the distributions in BLOOD dataset
get.MatH.stats(BLOOD, stat = "quantile", prob = 0.05)
  # the fifth percentiles of distributions in BLOOD dataset
**get.MatH.varnames**  
*Method get.MatH.varnames*

**Description**

It returns the labels of the columns, or the names of the variables, of a MatH object.

**Usage**

```r
get.MatH.varnames(object)
```

```
## S4 method for signature 'MatH'
get.MatH.varnames(object)
```

**Arguments**

- `object`  
  a MatH object

**Value**

A vector of char, the labels of the columns, or the names of the variables.

---

**get.s**  
*Method get.s: the standard deviation of a distribution*

**Description**

This function returns the standard deviation of a distributionH object.

**Usage**

```r
get.s(object)
```

```
## S4 method for signature 'distributionH'
get.s(object)
```

**Arguments**

- `object`  
  a distributionH object.

**Value**

A numeric positive value, the standard deviation.

**Examples**

```r
D <- distributionH(x = c(1, 2, 3, 4), p = c(0, 0.2, 0.6, 1))
get.s(D) # returns the standard deviation of D
```
HTS-class

Class HTS
defines a histogram time series, i.e. a set of histograms observed along time.

Usage
## S4 method for signature 'HTS'
initialize(.Object, epochs = 1, ListOfTimedElements = c(new("TdistributionH")))

Arguments
/Object/ the object type ("HTS") a histogram time series
epochs the number of histograms (one for each timepoint or period)
ListOfTimedElements a vector of TdistributionH objects

HTS.exponential.smoothing

Smoothing with exponential smoothing of a histogram time series

Description
(Beta version of) Extends the exponential smoothing of a time series to a histogram time series, using L2 Wasserstein distance.

Usage
HTS.exponential.smoothing(HTS, alpha = 0.9)

Arguments
HTS A HTS object (a histogram time series).
alpha a number between 0 and 1 for exponential smoothing

Value
a list with the results of the smoothing procedure.

Slots
smoothing.alpha the alpha parameter
AveragedHTS The smoothed HTS
HTS.moving.averages

Examples

```
mov.expo.smooth <- HTS.exponential.smoothing(HTS = RetHTS, alpha = 0.8)
# a show method for HTS must be implemented you can see it using
# str(mov.expo.smooth$AveragedHTS)
```

HTS.moving.averages | Smoothing with moving averages of a histogram time series

Description

(Beta version of) Extends the moving average smoothing of a time series to a histogram time series, using L2 Wasserstein distance.

Usage

```
HTS.moving.averages(HTS, k = 3, weights = rep(1, k))
```

Arguments

- **HTS**: A HTS object (a histogram time series).
- **k**: an integer value, the number of elements for moving averages
- **weights**: a vector of positive weights for a weighted moving average

Value

a list with the results of the smoothing procedure.

Slots

- **k**: the number of elements for the average
- **weights**: the vector of weights for smoothing
- **AveragedHTS**: The smoothed HTS

Examples

```
mov.av.smoothed <- HTS.moving.averages(HTS = RetHTS, k = 5)
# a show method for HTS must be implemented you can see it using
# str(mov.av.smoothed$AveragedHTS)
```
HTS.predict.knn  K-NN predictions of a histogram time series

Description

(Beta version of) Extends the K-NN algorithm for predicting a time series to a histogram time series, using L2 Wasserstein distance.

Usage

HTS.predict.knn(HTS, position = length(HTS@data), k = 3)

Arguments

- **HTS**: A HTS object (a histogram time series).
- **position**: an integer, the data histogram to predict
- **k**: the number of neighbours (default=3)

Details

Histogram time series (HTS) describe situations where a distribution of values is available for each instant of time. These situations usually arise when contemporaneous or temporal aggregation is required. In these cases, histograms provide a summary of the data that is more informative than those provided by other aggregates such as the mean. Some fields where HTS are useful include economy, official statistics and environmental science. The function adapts the k-Nearest Neighbours (k-NN) algorithm to forecast HTS and, more generally, to deal with histogram data. The proposed k-NN relies on the L2 Wasserstein distance that is used to measure dissimilarities between sequences of histograms and to compute the forecasts.

Value

a distributionHT object predicted from data.

References


Examples

prediction <- HTS.predict.knn(HTS = RetHTS, position = 108, k = 3)
is.registeredMH  

Method is.registeredMH

Description

Checks if a MatH contains histograms described by the same number of bins and the same cdf.

Usage

is.registeredMH(object)

## S4 method for signature 'MatH'

is.registeredMH(object)

Arguments

object  

A MatH object

Value

a logical value TRUE if the distributions share the same cdf, FALSE otherwise.

Author(s)

Antonio Irpino

References


Examples

```r
## ---- initialize three distributionH objects mydist1 and mydist2
mydist1 <- new("distributionH", c(1, 2, 3), c(0, 0.4, 1))
mydist2 <- new("distributionH", c(7, 8, 10, 15), c(0, 0.2, 0.7, 1))
mydist3 <- new("distributionH", c(9, 11, 20), c(0, 0.8, 1))
## create a MatH object
MyMAT <- new("MatH", nrows = 1, ncols = 3, ListOfDist = c(mydist1, mydist2, mydist3), 1, 3)
is.registeredMH(MyMAT)
## [1] FALSE #the distributions do not share the same cdf
## Hint: check with str(MyMAT)

## register the two distributions
MATregistered <- registerMH(MyMAT)
```
Method kurtH: computes the kurthosis of a distribution

Description
Kurtosis of a histogram (using the fourth standardized moment)

Usage

kurtH(object)

## S4 method for signature 'distributionH'
kurtH(object)

Arguments

object

a distributionH object

Value
A value for the kurtosis index, 3 is the kurtosis of a Gaussian distribution

Author(s)
Antonio Irpino

Examples

## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the kurthosis of mydist ----
kurtH(mydist) #---> 1.473242
Class MatH.

Description

Class MatH defines a matrix of distribution objects

This function creates a matrix of histogram data, i.e. a MatH object

Usage

```r
## S4 method for signature 'MatH'
initialize(
  .Object,
  nrows = 1,
  ncols = 1,
  ListOfDist = NULL,
  names.rows = NULL,
  names.cols = NULL,
  by.row = FALSE
)

MatH(
  x = NULL,
  nrows = 1,
  ncols = 1,
  rownames = NULL,
  varnames = NULL,
  by.row = FALSE
)
```

Arguments

- `.Object`: the object type "MatH"
- `nrows`: (optional, default=1) an integer, the number of rows.
- `ncols`: (optional, default=1) an integer, the number of columns (aka variables).
- `ListOfDist`: a vector or a list of distribution objects
- `names.rows`: a vector or list of strings with the names of the rows
- `names.cols`: a vector or list of strings with the names of the columns (variables)
- `by.row`: (optional, default=FALSE) a logical value, TRUE the matrix is row wise filled, FALSE the matrix is filled column wise.
- `x`: (optional, default= an empty distribution object) a list of distribution objects
- `rownames`: (optional, default=NULL) a list of strings containing the names of the rows.
- `varnames`: (optional, default=NULL) a list of strings containing the names of the columns (aka variables).
Value

A `MatH` object

Author(s)

Antonio Irpino

References


Examples

```r
## ---- create a list of six distributionH objects
ListOfDist <- list()
ListOfDist[[1]] <- distributionH(c(1, 2, 3), c(0, 0.4, 1))
ListOfDist[[2]] <- distributionH(c(7, 8, 10, 15), c(0, 0.2, 0.7, 1))
ListOfDist[[3]] <- distributionH(c(9, 11, 20), c(0, 0.5, 1))
ListOfDist[[4]] <- distributionH(c(2, 5, 8), c(0, 0.3, 1))
ListOfDist[[5]] <- distributionH(c(8, 10, 15), c(0, 0.75, 1))
ListOfDist[[6]] <- distributionH(c(20, 22, 24), c(0, 0.12, 1))

## create a MatH object filling it by columns
MyMAT <- new("MatH",
  nrows = 3, ncols = 2,
  ListOfDist = ListOfDist,
  names.rows = c("I1", "I2", "I3"),
  names.cols = c("Var1", "Var2"),
  by.row = FALSE)
showClass("MatH")

# building an empty 10 by 4 matrix of histograms
MAT <- MatH(nrows = 10, ncols = 4)
```

---

**meanH**

Method meanH: computes the mean of a distribution

Description

Mean of a histogram (First moment of the distribution)

Usage

```r
meanH(object)
```

```r
## S4 method for signature 'distributionH'
meanH(object)
```
### Arguments

- object: a `distributionH` object

### Value

-the mean of the distribution

### Author(s)

Antonio Irpino

### Examples

```r
## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the mean of mydist ----
meanH(mydist) #---> 4.4
```

---

### Description

-the difference of two distribution according to the L2 Wasssertein
-the difference of a number and a distribution according to the L2 Wasssertein
-the difference of a distribution and a number according to the L2 Wasssertein

### Usage

```r
## S4 method for signature 'distributionH,distributionH'
e1 - e2
## S4 method for signature 'numeric,distributionH'
e1 - e2
## S4 method for signature 'distributionH,numeric'
e1 - e2
```

### Arguments

- e1: a `distributionH` object or a number
- e2: a `distributionH` object or a number

### Note

-it may not works properly if the difference is not a distribution
**OzoneFull**

*Full Ozone dataset for Histogram data analysis*

**Description**

The dataset contains MatH (matrix of histogram-valued data) object. This data set lists 78 stations located in the USA recording four variables, without missing data.

**Format**

A MatH instance, 1 row per station.

**Author(s)**

Antonio Irpino, 2014-10-05

**Source**


---

**OzoneH**

*Complete Ozone dataset for Histogram data analysis*

**Description**

The dataset contains MatH (matrix of histogram-valued data) object. This data set lists 84 stations located in the USA recording four variables. Some stations contain missing data.

**Format**

A MatH instance, 1 row per station.

**Author(s)**

Antonio Irpino, 2014-10-05

**Source**

Description

A plot function for a distributionH object. The function returns a representation of the histogram.

Usage

## S4 method for signature 'distributionH'
plot(x, type = "HISTO", col = "green", border = "black")

Arguments

x
  a distributionH object

type
  (optional) a string describing the type of plot, default="HISTO". Other allowed types are
  "CDF"=Cumulative distribution function,
  "QF"= quantile function,
  "DENS"=a density approximation,
  "HBOXPLOT"=horizontal boxplot,
  "VBOXPLOT"= vertical boxplot,

col
  (optional) a string the color of the plot, default="green".

Examples

## ---- initialize a distributionH
mydist <- distributionH(x = c(7, 8, 10, 15), p = c(0, 0.2, 0.7, 1))
# show the histogram
plot(mydist) # plots mydist
plot(mydist, type = "HISTO", col = "red", border = "blue") # plots mydist
plot(mydist, type = "DENS", col = "red", border = "blue") # plots a density approximation for mydist
plot(mydist, type = "HBOXPLOT") # plots a horizontal boxplot for mydist
plot(mydist, type = "VBOXPLOT") # plots a vertical boxplot for mydist
plot(mydist, type = "CDF") # plots the cumulative distribution function of mydist
plot(mydist, type = "QF") # plots the quantile function of mydist

Description

An overloading plot function for a HTS object. The method returns a graphical representation of a histogram time series.
Usage

## S4 method for signature 'HTS'
plot(x, y = "missing", type = "VIOLIN", border = "black", maxno.perplot = 30)

Arguments

- **x**: a `distributionH` object
- **y**: not used in this implementation
- **type**: (optional) a string describing the type of plot, default="BOXPLOT". Other allowed types are
  - "VIOLIN"=a violin-plot representation,
- **border**: (optional) a string the color of the border of the plot, default="black".
- **maxno.perplot**: An integer (DEFAULT=30). Maximum number of timestamps per row. It allows a plot organized by rows, each row of the plot contains a max number of time stamps indicated by maxno.perplot.

Examples

```
plot(subsetHTS(RetHTS, from = 1, to = 10)) # plots RetHTS dataset
## Not run:
plot(RetHTS, type = "BOXPLOT", border = "blue", maxno.perplot = 20)
plot(RetHTS, type = "VIOLIN", border = "blue", maxno.perplot = 20)
plot(RetHTS, type = "VIOLIN", border = "blue", maxno.perplot = 10)
## End(Not run)
```

plot-MatH  Method plot for a matrix of histograms

Description

An overloading plot function for a MatH object. The method returns a graphical representation of the matrix of histograms.

Usage

## S4 method for signature 'MatH'
plot(x, y = "missing", type = "HISTO", border = "black", angL = 330)

Arguments

- **x**: a `distributionH` object
- **y**: not used in this implementation
- **type**: (optional) a string describing the type of plot, default="HISTO". Other allowed types are
  - "DENS"=a density approximation,
  - "BOXPLOT"=a boxplot
plot-TdistributionH

border (optional) a string the color of the border of the plot, default="black".
angL (optional) angle of labels of rows (DEFAULT=330).

Examples

plot(BLOOD) # plots BLOOD dataset
## Not run:
plot(BLOOD, type = "HISTO", border = "blue") # plots a matrix of histograms
plot(BLOOD, type = "DENS", border = "blue") # plots a matrix of densities
plot(BLOOD, type = "BOXPLOT") # plots a boxplots
## End(Not run)

Description

A plot function for a TdistributionH object. The function returns a representation of the histogram.

Usage

## S4 method for signature 'TdistributionH'
plot(x, type = "HISTO", col = "green", border = "black")

Arguments

x a TdistributionH object
type (optional) a string describing the type of plot, default="HISTO". Other allowed types are
"CDF"=Cumulative distribution function,
"QF"=quantile function,
"DENS"=a density approximation,
"HBOXPLOT"=horizontal boxplot,
"VBOXPLOT"=vertical boxplot,
col (optional) a string the color of the plot, default="green".
border (optional) a string the color of the border of the plot, default="black".
plotPredVsObs

A function for comparing observed vs predicted histograms

Description

This function allows the representation of observed vs predicted histograms. It can be used as a tool for interpreting predictive methods (for example, the regression of histogram data)

Usage

plotPredVsObs(PRED, OBS, type = "HISTO", ncolu = 2)

Arguments

PRED  a MatH object with one column, the predicted data
OBS   a MatH object with one column, the observed data
type  a string. "HISTO" (default), if ones want to compare histograms "CDF", if ones want to compare cumulative distribution functions; "DENS" if ones want to compare approximated densities (using KDE);
ncolu number of columns in which is arranged the plot, default is 2. If you have a lot of data consider to choose higher values.

Value

A plot with compared histogram-valued data.

Examples

```r
## do a regression
pars <- WH.regression.two.components(BLOOD, Yvar = 1, Xvars = c(2:3))
## predict data
PRED <- WH.regression.two.components.predict(data = BLOOD[, 2:3], parameters = pars)
## define observed data
## Not run:
OBS <- BLOOD[, 1]
plotPredVsObs(PRED, OBS, "HISTO")
plotPredVsObs(PRED, OBS, "CDF")
plotPredVsObs(PRED, OBS, "DENS")
## End(Not run)
```
plot_errors

A function for plotting functions of errors

Description

This function allows the representation of the difference between observed histograms and the respective predicted ones. It can be used as a tool for interpreting predictive methods (for example, the regression of histogram data).

Usage

plot_errors(PRED, OBS, type = "HISTO_QUA", np = 200)

Arguments

PRED    a Math object with one column, the predicted data
OBS     a Math object with one column, the observed data
type    a string. "HISTO_QUA" (default), if ones want to compare histograms quantile differences
         "HISTO_DEN", if ones want to show the histogram densities differences;
         "DENS_KDE" if ones want to show the differences between approximated densities (using KDE);
np       number of points considered for density or quantile computation (default=200).

Value

A plot with functions of differences between observed and predicted histograms, and a Root Mean Squared value computing by using the L2 Wasserstein distance.

Examples

## do a regression
pars <- WH.regression.two.components(BLOOD, Yvar = 1, Xvars = c(2:3))
## predict data
PRED <- WH.regression.two.components.predict(data = BLOOD[, 2:3], parameters = pars)
## define observed data
OBS <- BLOOD[, 1]
plot_errors(PRED, OBS, "HISTO_QUA")
plot_errors(PRED, OBS, "HISTO_DEN")
plot_errors(PRED, OBS, "DENS_KDE")
Description

Given two distributionH objects, it returns two equivalent distributions such that they share the same cdf values. This function is useful for computing basic statistics.

Usage

register(object1, object2)

## S4 method for signature 'distributionH,distributionH'
register(object1, object2)

Arguments

object1 A distributionH object

object2 A distributionH object

Value

The two distributionH objects in input sharing the same cdf (the p slot)

Author(s)

Antonio Irpino

References


Examples

## ---- initialize two distributionH objects mydist1 and mydist2
mydist1 <- distributionH(c(1, 2, 3), c(0, 0.4, 1))
mydist2 <- distributionH(c(7, 8, 10, 15), c(0, 0.2, 0.7, 1))

## register the two distributions
regDist <- register(mydist1, mydist2)

## OUTPUT:
## regDist$[[1]]
## A object of class "distributionH"
registerMH

Method registerMH

Description

registerMH method registers a set of distributions of a MatH object. All the distributions are recomputed to obtain distributions sharing the same p slot. This method is useful for using fast computation of all methods based on L2 Wasserstein metric. The distributions will have the same number of elements in the x slot without modifying their density function.

Usage

registerMH(object)

## S4 method for signature 'MatH'

registerMH(object)

Arguments

object A MatH object (a matrix of distributions)

Value

A MatH object, a matrix of distributions sharing the same p slot (i.e. the same cdf).

Author(s)

Antonio Irpino

References

Examples

# initialize three distributionH objects mydist1 and mydist2
mydist1 <- new("distributionH", c(1, 2, 3), c(0, 0.4, 1))
mydist2 <- new("distributionH", c(7, 8, 10, 15), c(0, 0.2, 0.7, 1))
mydist3 <- new("distributionH", c(9, 11, 20), c(0, 0.8, 1))

# create a MatH object
MyMAT <- new("MatH", nrows = 1, ncols = 3, ListOfDist = c(mydist1, mydist2, mydist3), 1, 3)

# register the two distributions
MATregistered <- registerMH(MyMAT)

# OUTPUT the structure of MATregistered
str(MATregistered)

RetHTS

A histogram-valued dataset of returns

Description

A histogram-valued dataset of returns of dollar vs yen change rates

Format

a MatH object, a matrix of distributions.

Author(s)

Antonio Irpino, 2014-10-05
**Description**

Quantile-Quantile correlation between two distributions

**Usage**

\[ \texttt{rQQ(e1, e2)} \]

## S4 method for signature 'distributionH,distributionH'

\[ \texttt{rQQ(e1, e2)} \]

**Arguments**

- **e1**: A `distributionH` object
- **e2**: A `distributionH` object

**Value**

Pearson correlation index between quantiles

**Author(s)**

Antonio Irpino

**References**


**Examples**

```r
## ---- initialize two distributionH object mydist1 and mydist2
mydist1 <- distributionH(x = c(1, 2, 3), p = c(0, 0.4, 1))
mydist2 <- distributionH(x = c(7, 8, 10, 15), p = c(0, 0.2, 0.7, 1))
## computes the rQQ
rQQ(mydist1, mydist2)
## OUTPUT 0.916894
```
**set.cell.MatH**

*Method set.cell.MatH assign a histogram to a cell of a matrix of histograms*

**Description**

Assign a histogram data to the r-th row and the c-th column of a matrix of histograms.

**Usage**

```r
set.cell.MatH(object, mat, r, c)
```

## S4 method for signature 'distributionH,MatH,numeric,numeric'

```r
set.cell.MatH(object, mat, r, c)
```

**Arguments**

- `object`: a distributionH object, a matrix of distributions.
- `mat`: a MatH object, a matrix of distributions.
- `r`: an integer, the row index.
- `c`: an integer, the column index

**Value**

A MatH object.

**Examples**

```r
mydist <- distributionH(x = c(0, 1, 2, 3, 4), p = c(0, 0.1, 0.6, 0.9, 1))
MAT <- set.cell.MatH(mydist, BLOOD, r = 1, c = 1)
```

---

**ShortestDistance**

*Shortes distance from a point o a 2d segment*

**Description**

Shortes distance from a point o a 2d segment

**Usage**

```r
ShortestDistance(p, line)
```

**Arguments**

- `p`: coordinates of a point
- `line`: a 2x2 matrix with the coordinates of two points defining a line
Value

A numeric value, the Euclidean distance of point p to the line.

See Also

data2hist function and DouglasPeucker function

show  Method show for distributionH

Description

An overriding show function for a distributionH object. The function returns a representation of the histogram, if the number of bins is high the central part of the histogram is truncated.

Usage

```r
## S4 method for signature 'distributionH'
show(object)
```

Arguments

object a distributionH object

Examples

```r
## ---- initialize a distributionH
mydist <- distributionH(x = c(7, 8, 10, 15), p = c(0, 0.2, 0.7, 1))
# show the histogram
mydist
```

show-Math  Method show for MatH

Description

An overriding show method for a MatH object. The method returns a representation of the matrix using the mean and the standard deviation for each histogram.

Usage

```r
## S4 method for signature 'MatH'
show(object)
```

Arguments

object a MatH object
Examples

```r
show(BLOOD)
print(BLOOD)
BLOOD
```

---

### skewH

**Method skewH**: computes the skewness of a distribution

---

### Description

Skewness of a histogram (using the third standardized moment)

### Usage

```r
skewH(object)
```

---

### Arguments

- **object**: a `distributionH` object

---

### Value

A value for the skewness index

---

### Author(s)

Antonio Irpino

---

### Examples

```r
## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the skewness of mydist ----
skewH(mydist) #---> -1.186017
```
stations_coordinates  
*Stations coordinates of China_Month and China_Seas datasets*

**Description**

A dataset containing the geographical coordinates of stations described in China_Month and China_Seas datasets

**Format**

a data.frame

**Author(s)**

Antonio Irpino, 2014-10-05

**Source**


---

**stdH**

*Method stdH: computes the standard deviation of a distribution*

**Description**

Standard deviation of a histogram (i.e., the square root of the centered second moment)

**Usage**

```
stdH(object)
```

```r
## S4 method for signature 'distributionH'
stdH(object)
```

**Arguments**

- `object`  
a distributionH object

**Value**

A value for the standard deviation

**Author(s)**

Antonio Irpino
Examples

```r
## ---- A mydist distribution ----
mydist <- distributionH(x = c(1, 2, 3, 10), p = c(0, 0.1, 0.5, 1))
## ---- Compute the standard deviation of mydist ----
stdH(mydist) #---> 2.563851
```

### Method subsetHTS: extract a subset of a histogram time series

Description

This function returns the mean of a `distributionH` object.

Usage

```r
subsetHTS(object, from, to)
```

Arguments

- `object`: A `HTS` object. A histogram 1d time series
- `from`: an integer, the initial timepoint
- `to`: an integer, a final timepoint

Value

A `HTS` object. A histogram 1d time series

Examples

```r
SUB_RetHTS <- subsetHTS(RetHTS, from = 1, to = 20) # the first 20 elements
```
Class \textit{TdistributionH} defines a histogram with a time (point or period).

Usage

```r
## S4 method for signature 'TdistributionH'
initialize(
  .Object,
  tstamp = numeric(0),
  period = list(start = -Inf, end = -Inf),
  x = numeric(0),
  p = numeric(0),
  m = numeric(0),
  s = numeric(0)
)
```

Arguments

- \texttt{.Object}: the type of object ("TdistributionH") a "distributionH" object with a time reference.
- \texttt{tstamp}: a numeric value related to a timestamp.
- \texttt{period}: a list of two values, the starting time and the ending time (alternative to \texttt{tstamp} if the distribution is observed along a period and not on a timestamp).
- \texttt{x}: a vector of increasing values, the domain of the distribution (the same of \textit{distributionH} object).
- \texttt{p}: a vector of increasing values from 0 to 1, the CDF of the distribution (the same of \textit{distributionH} object).
- \texttt{m}: a number, the mean of the distribution (the same of \textit{distributionH} object).
- \texttt{s}: a positive number, the standard deviation of the distribution (the same of \textit{distributionH} object).

Class \textit{TMatH} defines a matrix of histograms, a \textit{TMatH} object, with a time (a timepoint or a time window).
Usage

```r
## S4 method for signature 'TMatH'
initialize(
  .Object,
  tstamp = numeric(0),
  period = list(start = -Inf, end = -Inf),
  mat = new("MatH")
)
```

Arguments

- `.Object`: the type of object ("TMatH")
- `tstamp`: a vector of time stamps, numeric.
- `period`: a list of pairs with a vector of starting time and a vector of ending time. This parameter is used alternatively to `tstamp` if the distributions are related to time periods instead of timestamps
- `mat`: a `MatH` object

Description

Computes the squared L2 Wasserstein distance between two `distributionH` objects.

Usage

```r
WassSqDistH(object1, object2, ...)
```

Arguments

- `object1`: is an object of `distributionH` class
- `object2`: is an object of `distributionH` class
- `...`: optional parameters
- `details` (optional, default=FALSE) is a logical value, if TRUE returns the decomposition of the distance

Value

If `details=FALSE`, the function returns the squared L2 Wasserstein distance.
If `details=TRUE`, the function returns list containing the squared distance, its decomposition in three parts (position, size and shape) and the correlation coefficient between the quantile functions.
References


Examples

```r
## ---- create two distributionH objects ----
mydist1 <- distributionH(x = c(1, 2, 3), p = c(0, 0.4, 1))
mydist2 <- distributionH(x = c(7, 8, 10, 15), p = c(0, 0.2, 0.7, 1))
# -- compute the squared L2 Waaserstein distance
WassSqDistH(mydist1, mydist2)
# -- compute the squared L2 Waaserstein distance with details
WassSqDistH(mydist1, mydist2, details = TRUE)
```

---

**WH.1d.PCA**

Principal components analysis of histogram variable based on Wasserstein distance

Description

The function implements a Principal components analysis of histogram variable based on Wasserstein distance. It performs a centered (not standardized) PCA on a set of quantiles of a variable. Being a distribution a multivalued description, the analysis performs a dimensional reduction and a visualization of distributions. It is a 1d (one dimension) because it is considered just one histogram variable.

Usage

```
WH.1d.PCA(
  data,     # A MatH object (a matrix of distributionH).
  var,      # An integer, the variable number.
  quantiles = 10,     # An integer, it is the number of quantiles used in the analysis.
  plots = TRUE,         # a logical value. Default=TRUE plots are drawn.
  listaxes = c(1:4),  # An integer, the variable number.
  axisequal = FALSE,   # An integer, it is the number of quantiles used in the analysis.
  qcut = 1,            # An integer, it is the number of quantiles used in the analysis.
  outl = 0
)
```
listaxes  A vector of integers listing the axis for the 2d factorial representations.
axisequal  A logical value. Default TRUE, the plot have the same scale for the x and the y axes.
qcut  a number between 0.5 and 1, it is used for the plot of densities, and avoids very peaked densities. Default=1, all the densities are considered.
outl  a number between 0 (default) and 0.5. For each distribution, is the amount of mass removed from the tails of the distribution. For example, if 0.1, from each distribution is cut away a left tail and a right one each containing the 0.1 of mass.

Details
In the framework of symbolic data analysis (SDA), distribution-valued data are defined as multivalued data, where each unit is described by a distribution (e.g., a histogram, a density, or a quantile function) of a quantitative variable. SDA provides different methods for analyzing multivalued data. Among them, the most relevant techniques proposed for a dimensional reduction of multivalued quantitative variables is principal component analysis (PCA). This paper gives a contribution in this context of analysis. Starting from new association measures for distributional variables based on a peculiar metric for distributions, the squared Wasserstein distance, a PCA approach is proposed for distribution-valued data, represented by quantile-variables.

Value
a list with the results of the PCA in the MFA format of package FactoMineR for function MFA

References

Examples
results <- WH.1d.PCA(data = BLOOD, var = 1, listaxes = c(1:2))

WH.bind  Method WH.bind
It attaches two MatH objects with the same columns by row, or the same rows by column.
Usage

```r
WH.bind(object1, object2, byrow)
```

## S4 method for signature 'MatH,MatH'

```r
WH.bind(object1, object2, byrow = TRUE)
```

Arguments

- `object1`: a `MatH` object
- `object2`: a `MatH` object
- `byrow`: a logical value (default=TRUE) attaches the objects by row

Value

a `MatH` object.

See Also

- `WH.bind.row` for binding by row. `WH.bind.col` for binding by column

Examples

```r
# binding by row
M1 <- BLOOD[1:10, 1]
M2 <- BLOOD[1:10, 3]
MAT <- WH.bind(M1, M2, byrow = TRUE)

# binding by col
M1 <- BLOOD[1:10, 1]
M2 <- BLOOD[1:10, 3]
MAT <- WH.bind(M1, M2, byrow = FALSE)
```
**WH.bind.row**

**Arguments**

- object1: a MatH object
- object2: a MatH object

**Value**

a MatH object.

**Examples**

```r
M1 <- BLOOD[1:10, 1]
M2 <- BLOOD[1:10, 3]
MAT <- WH.bind.col(M1, M2)
```

---

**WH.bind.row**

**Method WH.bind.row**

**Description**

It attaches two MatH objects with the same columns by row.

**Usage**

```r
WH.bind.row(object1, object2)
```

```r
## S4 method for signature 'MatH,MatH'
WH.bind.row(object1, object2)
```

**Arguments**

- object1: a MatH object
- object2: a MatH object

**Value**

a MatH object.

**Examples**

```r
M1 <- BLOOD[1:3, ]
M2 <- BLOOD[5:8, ]
MAT <- WH.bind.row(M1, M2)
```
WH.correlation  

**Method WH.correlation**

**Description**

Compute the correlation matrix of a `MatH` object, i.e. a matrix of values consistent with a set of distributions equipped with a L2 wasserstein metric.

**Usage**

```r
WH.correlation(object, ...)  
## S4 method for signature 'MatH'
WH.correlation(object, w = numeric(0))
```

**Arguments**

- `object`  
  a `MatH` object

- `...`  
  some optional parameters

- `w`  
  it is possible to add a vector of weights (positive numbers) having the same size of the rows of the `MatH` object, default = equal weight for each row

**Value**

a squared matrix with the (weighted) correlations indices

**References**


**Examples**

```r
WH.correlation(BLOOD)
# generate a set of random weights
RN <- runif(get.MatH.nrows(BLOOD))
WH.correlation(BLOOD, w = RN)
```
**WH.correlation2**

**Method WH.correlation2**

**Description**

Compute the correlation matrix using two MatH objects having the same number of rows. It returns a rectangular a matrix of numbers, consistent with a set of distributions equipped with a L2 wasserstein metric.

**Usage**

```
WH.correlation2(object1, object2, ...)
```

## S4 method for signature 'MatH,MatH'

```
WH.correlation2(object1, object2, w = numeric(0))
```

**Arguments**

- `object1`: a MatH object
- `object2`: a MatH object
- `...`: some optional parameters
- `w`: it is possible to add a vector of weights (positive numbers) having the same size of the rows of the MatH object, default = equal weight for each row

**Value**

a rectangular matrix with the weighted sum of squares

**Examples**

```
M1 <- BLOOD[, 1]
M2 <- BLOOD[, 2:3]
WH.correlation2(M1, M2)
# generate a set of random weights
RN <- runif(get.MatH.nrows(BLOOD))
WH.correlation2(M1, M2, w = RN)
```
Method WH.mat.prod

Description

It is the matrix product of two MatH objects, i.e. two matrices of distributions, by using the dot product of two histograms that is consistent with a set of distributions equipped with a L2 wasserstein metric.

Usage

WH.mat.prod(object1, object2, ...)

## S4 method for signature 'MatH,MatH'
WH.mat.prod(object1, object2, transpose1 = FALSE, transpose2 = FALSE)

Arguments

- object1: a MatH object
- object2: a MatH object
- ...: other optional parameters
- transpose1: a logical value, default=FALSE. If TRUE transposes object1
- transpose2: a logical value, default=FALSE. If TRUE transposes object2

Value

A matrix of numbers

Examples

M1 <- BLOOD[1:5, ]
M2 <- BLOOD[6:10, ]
MAT <- WH.mat.prod(M1, M2, transpose1 = TRUE, transpose2 = FALSE)

Method WH.mat.sum

Description

It sums two MatH objects, i.e. two matrices of distributions, by summing the quantile functions of histograms. This sum is consistent with a set of distributions equipped with a L2 wasserstein metric.
Usage

WH.mat.sum(object1, object2)

## S4 method for signature 'MatH,MatH'
WH.mat.sum(object1, object2)

Arguments

- object1: a MatH object
- object2: a MatH object

Value

- a MatH object,

Examples

# binding by row
M1 <- BLOOD[1:5, ]
M2 <- BLOOD[6:10, ]
MAT <- WH.mat.sum(M1, M2)

Description

(Beta version) The function implements a Principal components analysis of a set of histogram variables based on Wasserstein distance. It performs a centered (not standardized) PCA on a set of quantiles of a variable. Being a distribution a multivalued description, the analysis performs a dimensional reduction and a visualization of distributions. It is a 1d (one dimension) because it is considered just one histogram variable.

Usage

WH.MultiplePCA(data, list.of.vars, quantiles = 10, outl = 0)

Arguments

- data: A MatH object (a matrix of distributionH).
- list.of.vars: A list of integers, the active variables.
- quantiles: An integer, it is the number of quantiles used in the analysis. Default=10.
- outl: a number between 0 (default) and 0.5. For each distribution, is the amount of mass removed from the tails of the distribution. For example, if 0.1, from each distribution is cut away a left tail and a right one each containing the 0.1 of mass.
Details

It is an extension of WH.1d.PCA to the multiple case.

Value

a list with the results of the PCA in the MFA format of package FactoMineR for function MFA

WH.plot_multiple_indivs

Plot histograms of individuals after a Multiple factor analysis of Histogram Variables

Description

(Beta version) The function plots histogram data of the individuals for a particular variable on a factorial palne after a Multiple factor analysis.

Usage

WH.plot_multiple_indivs(
  data,
  res,
  axes = c(1, 2),
  indiv = 0,
  var = 1,
  strx = 0.1,
  stry = 0.1,
  HISTO = TRUE,
  coor = 0,
  stat = "mean"
)

Arguments

data a MatH object
res Results from WH.MultiplePCA.
axes A list of integers, the new factorial axes c(1,2) are the default.
indiv A list of objects (rows) of data to plot. Default=0 all the objects of data.
var An integer indicating an original histogram variable to plot.
strx a resizing factor for the domain of histograms (default=0.1 means that each distribution has a support that is one tenth of the spread of the x axis)
stry a resizing factor for the density of histograms (default=0.1 means that each distribution has a density that is one tenth of the spread of the y axis)
HISTO a logical value. Default=TRUE plots histograms, FALSE plot smooth densities.
WH.plot_multiple_Spanish.funs

**Description**

The function plots the circle of correlation of the quantiles of the histogram variables after a Multiple factor analysis.

**Usage**

```r
WH.plot_multiple_Spanish.funs(
  res,
  axes = c(1, 2),
  var = 1,
  LABS = TRUE,
  coor = (optional) if 0 (Default) takes the coordinates in res, if a matrix is passed the coordinates are those passed
  stat = (optional) if 'mean' (Default) a plot of individuals labeled by the means is produced. Otherwise if 'std', 'skewness' or 'kurtosis', data are labeled with this statistic.

**Value**

A plot of class ggplot

**Examples**

```r
# Do a MultiplePCA on the BLOOD dataset
## Not run:
# results=WH.MultiplePCA(BLOOD,list.of.vars = c(1:3))
# Plot histograms of variable 1 of BLOOD dataset on the first factorial plane showing histograms
WH.plot_multiple_indivs(BLOOD, results,
  axes = c(1, 2), var = 1, strx = 0.1,
  stry = 0.1, HISTO = TRUE
)
# Plot histograms of variable 1 of BLOOD dataset on the first factorial plane showing densities
WH.plot_multiple_indivs(BLOOD, results,
  axes = c(1, 2), var = 1, strx = 0.1,
  stry = 0.1, HISTO = FALSE
)
## End(Not run)
```
WH.regression.GOF

Goodness of Fit indices for Multiple regression of histogram variables based on a two component model and L2 Wasserstein distance

Description

It computes three goodness of fit indices using the results and the predictions of a regression done with WH.regression.two.components function.

Usage

WH.regression.GOF( observed, predicted)
Arguments

observed A one column MatH object, the observed histogram variable
predicted A one column MatH object, the predicted histogram variable.

Value

a list with the GOF indices

References


Examples

# do regression
model.parameters <- WH.regression.two.components(data = BLOOD, Yvar = 1, Xvars = c(2:3))
# do prediction
Predicted.BLOOD <- WH.regression.two.components.predict(data = BLOOD[, 2:3], parameters = model.parameters)
# compute GOF indices
GOF.indices <- WH.regression.GOF(observed = BLOOD[, 1], predicted = Predicted.BLOOD)

WH.regression.two.components

Multiple regression analysis for histogram variables based on a two component model and L2 Wasserstein distance

Description

The function implements Multiple regression analysis for histogram variables based on a two component model and L2 Wasserstein distance. Taking as input dependent histogram variable and a set of explanatory histogram variables the methods return a least squares estimation of a two component regression model based on the decomposition of L2 Wasserstein metric for distributional data.

Usage

WH.regression.two.components(data, Yvar, Xvars, simplify = FALSE, qua = 20)
Arguments

data A MatH object (a matrix of distributionH).
Yvar An integer, the dependent variable number in data.
Xvars A set of integers the explanatory variables in data.
simplify a logical argument (default=FALSE). If TRUE only few equally spaced quantiles are considered (for speeding up the algorithm).
qua If simplify=TRUE is the number of quantiles to consider.

Details

A two component regression model is implemented. The observed variables are histogram variables according to the definition given in the framework of Symbolic Data Analysis and the parameters of the model are estimated using the classic Least Squares method. An appropriate metric is introduced in order to measure the error between the observed and the predicted distributions. In particular, the Wasserstein distance is proposed. Such a metric permits to predict the response variable as direct linear combination of other independent histogram variables.

Value

a named vector with the model estimated parameters

References


Examples

model.parameters <- WH.regression.two.components(data = BLOOD, Yvar = 1, Xvars = c(2:3))
Arguments

data A MatH object (a matrix of distributions) explanatory part.
parameters A named vector with the parameter from a WH.regression.two.components model

Value

a MatH object, the predicted histograms

References


Examples

# do regression
model.parameters <- WH.regression.two.components(data = BLOOD, Yvar = 1, Xvars = c(2:3))
# do prediction
Predicted.BLOOD <- WH.regression.two.components.predict(data = BLOOD[, 2:3], parameters = model.parameters)

WH.SSQ

Method WH.SSQ

Description

Compute the sum-of-squares-deviations (from the mean) matrix of a MatH object, i.e. a matrix of numbers, consistent with a set of distributions equipped with a L2 wasserstein metric.

Usage

WH.SSQ(object, ...)

## S4 method for signature 'MatH'
WH.SSQ(object, w = numeric(0))

Arguments

object a MatH object

... some optional parameters

w it is possible to add a vector of weights (positive numbers) having the same size of the rows of the MatH object, default = equal weight for each row
Value

a squared matrix with the weighted sum of squares

Examples

```
WH.SSQ(BLOOD)
# generate a set of random weights
RN <- runif(get.MatH.nrows(BLOOD))
WH.SSQ(BLOOD, w = RN)
```

WH.SSQ2

Method WH.SSQ2

Description

Compute the sum-of-squares-deviations (from the mean) matrix using two MatH objects having the same number of rows. It returns a rectangular a matrix of numbers, consistent with a set of distributions equipped with a L2 wasserstein metric.

Usage

```
WH.SSQ2(object1, object2, ...)
```

## S4 method for signature 'MatH,MatH'
```
WH.SSQ2(object1, object2, w = numeric(0))
```

Arguments

- `object1`: a MatH object
- `object2`: a MatH object
- `...`: some optional parameters
- `w`: it is possible to add a vector of weights (positive numbers) having the same size of the rows of the MatH object, default = equal weight for each row

Value

a rectangular matrix with the weighted sum of squares

Examples

```
M1 <- BLOOD[, 1]
M2 <- BLOOD[, 2:3]
WH.SSQ2(M1, M2)
# generate a set of random weights
RN <- runif(get.MatH.nrows(BLOOD))
WH.SSQ2(M1, M2, w = RN)
```
**WH.var.covar**

**Method WH.var.covar**

**Description**

Compute the variance-covariance matrix of a `MatH` object, i.e. a matrix of values consistent with a set of distributions equipped with a L2 wasserstein metric.

**Usage**

```r
WH.var.covar(object, ...)  
## S4 method for signature 'MatH'  
WH.var.covar(object, w = numeric(0))
```

**Arguments**

- `object` a `MatH` object
- `...` some optional parameters
- `w` it is possible to add a vector of weights (positive numbers) having the same size of the rows of the `MatH` object, default = equal weight for each row

**Value**

a squared matrix with the (weighted) variance-covariance values

**References**


**Examples**

```r
WH.var.covar(BLOOD)  
# generate a set of random weights  
RN <- runif(get.MatH.nrows(BLOOD))  
WH.var.covar(BLOOD, w = RN)
```
Description

Compute the covariance matrix using two MatH objects having the same number of rows. It returns a rectangular a matrix of numbers, consistent with a set of distributions equipped with a L2 wasserstein metric.

Usage

\texttt{WH.var.covar2(object1, object2, \ldots)}

## S4 method for signature 'MatH,MatH'
\texttt{WH.var.covar2(object1, object2, w = numeric(0))}

Arguments

\begin{itemize}
  \item \texttt{object1} a MatH object
  \item \texttt{object2} a MatH object
  \item \ldots some optional parameters
  \item \texttt{w} it is possible to add a vector of weights (positive numbers) having the same size of the rows of the MatH object, default = equal weight for each row
\end{itemize}

Value

a rectangular matrix with the weighted sum of squares

Examples

\begin{verbatim}
M1 <- BLOOD[, 1]
M2 <- BLOOD[, 2:3]
\texttt{WH.var.covar2(M1, M2)}  
# generate a set of random weights
RN <- \texttt{runif(get.MatH.nrows(BLOOD))}
\texttt{WH.var.covar2(M1, M2, w = RN)}
\end{verbatim}
### WH.vec.mean

**Method WH.vec.mean**

**Description**

Compute a histogram that is the weighted mean of the set of histograms contained in a MatH object, i.e. a matrix of histograms, consistent with a set of distributions equipped with a L2 Wasserstein metric.

**Usage**

\[
\text{WH.vec.mean}(\text{object}, ...) \\
## S4 method for signature 'MatH' \\
\text{WH.vec.mean}(\text{object}, w = \text{numeric}(0))
\]

**Arguments**

- **object**: a MatH object
- **...**: optional arguments
- **w**: it is possible to add a vector of weights (positive numbers) having the same size of the MatH object, default = equal weights for all

**Value**

a distribution object, i.e. a histogram

**Examples**

```r
hmean <- WH.vec.mean(BLOOD) \\
# generate a set of random weights \\
RN <- runif(get.MatH.nrows(BLOOD) * get.MatH.ncols(BLOOD)) \\
hmean <- WH.vec.mean(BLOOD, w = RN)
```

### WH.vec.sum

**Method WH.vec.sum**

**Description**

Compute a histogram that is the weighted sum of the set of histograms contained in a MatH object, i.e. a matrix of histograms, consistent with a set of distributions equipped with a L2 Wasserstein metric.
Usage

\texttt{WH.vec.sum(object, \ldots)}

\texttt{## S4 method for signature 'MatH'  
WH.vec.sum(object, w = numeric(0))}

Arguments

- \texttt{object}: a MatH object
- \ldots: optional arguments
- \texttt{w}: it is possible to add a vector of weights (positive numbers) having the same size of the MatH object, default = equal weights for all cells

Value

a distributionH object, i.e. a histogram

Examples

\begin{verbatim}
hsum <- WH.vec.sum(BLOOD)  
# generate a set of random weights
RN <- runif(get.MatH.nrows(BLOOD) * get.MatH.ncols(BLOOD))
hsum <- WH.vec.sum(BLOOD, w = RN)
### SUM of distributions ----
\end{verbatim}

\textbf{WH_2d_Adaptive_Kohonen_maps}

\textit{Batch Kohonen self-organizing 2d maps using adaptive distances for histogram-valued data}

Description

The function implements a Batch Kohonen self-organizing 2d maps algorithm for histogram-valued data.

Usage

\begin{verbatim}
WH_2d_Adaptive_Kohonen_maps(
  x,
  net = list(xdim = 4, ydim = 3, topo = c("rectangular")),
  kern.param = 2,
  TMAX = -9999,
  Tmin = -9999,
  niter = 30,
  repetitions,
  simplify = FALSE,
  qua = 10,
\end{verbatim}
standardize = FALSE,
schema = 6,
init.weights = "EQUAL",
weight.sys = "PROD",
theta = 2,
Wfix = FALSE,
verbose = FALSE,
atleast = 2
)

Arguments

x  A MatH object (a matrix of distributionH).

net a list describing the topology of the net list(xdim=number of rows,ydim=numbers of columns,topo=c('rectangular' or 'hexagonal')), see somgrid syntax in packageclass default net=list(xdim=4,ydim=3,topo=c('rectangular'))

kern.param (default =2) the kernel parameter for the RBF kernel used in the algorithm

TMAX a parameter useful for the iterations (default=2)

Tmin a parameter useful for the iterations (default=0.2)

niter maximum number of iterations (default=30)

repetitions number of repetition of the algorithm (default=5), because each launch may generate a local optimum

simplify a logical parameter for speeding up computations (default=FALSE). If true data are recoded in order to have fast computations

qua if simplify=TRUE number of equally spaced quantiles for recodify the histograms (default=10)

standardize A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wassertein based standard deviation. Use if one wants to have variables with std equal to one.

schema a number from 1 to 4

1=A weight for each variable (default)
2=A weight for the average and the dispersion component of each variable
3=Same as 1 but a different set of weights for each cluster
4=Same as 2 but a different set of weights for each cluster

init.weights a string how to initialize weights: 'EQUAL' (default), all weights are the same,

weight.sys a string. Weights may add to one ('SUM') or their product is equal to 1 ('PROD', default).

theta a number. A parameter if weight.sys='SUM', default is 2.

Wfix a logical parameter (default=FALSE). If TRUE the algorithm does not use adaptive distances.

verbose a logical parameter (default=FALSE). If TRUE details of computation are shown during the execution. #'

atleast integer. Check for degeneration of the map into a very low number of voronoi sets. (default 2) 2 means that the map will have at least 2 neurons attracting data instances in their voronoi sets.
Details

An extension of Batch Self Organised Map (BSOM) is here proposed for histogram data. These
type of data have been defined in the context of symbolic data analysis. The BSOM cost function
is then based on a distance function: the L2 Wasserstein distance. This distance has been widely
proposed in several techniques of analysis (clustering, regression) when input data are expressed by
distributions (empirical by histograms or theoretical by probability distributions). The peculiarity
of such distance is to be an Euclidean distance between quantile functions so that all the properties
proved for L2 distances are verified again. An adaptive versions of BSOM is also introduced
considering an automatic system of weights in the cost function in order to take into account the
different effect of the several variables in the Self-Organised Map grid.

Value

a list with the results of the Batch Kohonen map

Slots

solution A list. Returns the best solution among the repetitionsetitons, i.e. the one having the
minimum sum of squares criterion.
solution$MAP The map topology.
solution$IDX A vector. The clusters at which the objects are assigned.
solution$cardinality A vector. The cardinality of each final cluster.
solution$proto A MatH object with the description of centers.
solution$Crit A number. The criterion (Sum od square deviation from the centers) value at the
end of the run.
solution$Weights.comp the final weights assigned to each component of the histogram variables
solution$Weight.sys a string the type of weighting system ("SUM" or "PRODUCT")
quality A number. The percentage of Sum of square deviation explained by the model. (The
higher the better)

References

Irpinio A, Verde R, De Carvalho FAT (2012). Batch self organizing maps for interval and histogram

Examples

```r
## Not run:
results <- WH_2d_Adaptive_Kohonen_maps(
x = BLOOD,
net = list(xdim = 2, ydim = 3, topo = c("rectangular"),
repetitions = 2, simplify = TRUE,
qua = 10, standardize = TRUE
)
## End(Not run)
```
**WH_2d_Kohonen_maps**

Batch Kohonen self-organizing 2d maps for histogram-valued data

---

**Description**

The function implements a Batch Kohonen self-organizing 2d maps algorithm for histogram-valued data.

**Usage**

```r
WH_2d_Kohonen_maps(
  x,
  net = list(xdim = 4, ydim = 3, topo = c("rectangular")),
  kern.param = 2,
  TMAX = 2,
  Tmin = 0.2,
  niter = 30,
  repetitions = 5,
  simplify = FALSE,
  qua = 10,
  standardize = FALSE,
  verbose = FALSE
)
```

**Arguments**

- `x`: A MatH object (a matrix of distribution).
- `net`: a list describing the topology of the net list(xdim=number of rows, ydim=numbers of columns,topo=c('rectangular' or 'hexagonal')), see somgrid syntax in packageclass.
- `kern.param`: (default =2) the kernel parameter for the RBF kernel used in the algorithm.
- `TMAX`: a parameter useful for the iterations (default=2)
- `Tmin`: a parameter useful for the iterations (default=0.2)
- `niter`: maximum number of iterations (default=30)
- `repetitions`: number of repetition of the algorithm (default=5), because each launch may generate a local optimum
- `simplify`: a logical parameter for speeding up computations (default=FALSE). If true data are recoded in order to have fast computations
- `qua`: if simplify=TRUE number of equally spaced quantiles for recodify the histograms (default=10)
- `standardize`: A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wassertein based standard deviation. Use if one wants to have variables with std equal to one.
- `verbose`: a logical parameter (default=FALSE). If TRUE details of computation are shown during the execution.
Details

An extension of Batch Self Organised Map (BSOM) is here proposed for histogram data. These kind of data have been defined in the context of symbolic data analysis. The BSOM cost function is then based on a distance function: the L2 Wasserstein distance. This distance has been widely proposed in several techniques of analysis (clustering, regression) when input data are expressed by distributions (empirical by histograms or theoretical by probability distributions). The peculiarity of such distance is to be an Euclidean distance between quantile functions so that all the properties proved for L2 distances are verified again. An adaptative versions of BSOM is also introduced considering an automatic system of weights in the cost function in order to take into account the different effect of the several variables in the Self-Organised Map grid.

Value

a list with the results of the Batch Kohonen map

Slots

solution A list. Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares criterion.
solution$MAP The map topology.
solution$IDX A vector. The clusters at which the objects are assigned.
solution$cardinality A vector. The cardinality of each final cluster.
solution$proto A MatH object with the description of centers.
solution$Crit A number. The criterion (Sum of square deviation from the centers) value at the end of the run.
quality A number. The percentage of Sum of square deviation explained by the model. (The higher the better)

References


Examples

```r
## Not run:
results <- WH_2d_Kohonen_maps(
x = BLOOD,
   net = list(xdim = 2, ydim = 3, topo = c("rectangular")),
   repetitions = 2, simplify = TRUE,
   qua = 10, standardize = TRUE
)
## End(Not run)
```
WH_adaptive.kmeans

**WH_adaptive.kmeans**

*K-means of a dataset of histogram-valued data using adaptive Wasserstein distances*

**Description**

The function implements the k-means using adaptive distance for a set of histogram-valued data.

**Usage**

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

**Arguments**

- **x**: A MatH object (a matrix of distribution).
- **k**: An integer, the number of groups.
- **schema**: An integer, the number of histogram-valued data.
  1= A weight for each variable (default)
  2= A weight for the average and the dispersion component of each variable
  3= Same as 1 but a different set of weights for each cluster
  4= Same as 2 but a different set of weights for each cluster
- **init**: (optional, do not use) initialization for partitioning the data default is 'RPART', other strategies should be implemented.
- **rep**: An integer, maximum number of repetitions of the algorithm (default rep=5).
- **simplify**: A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
- **qua**: An integer, if simplify=TRUE is the number of quantiles used for recodify the histograms.
- **standardize**: A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wasserstein based standard deviation. Use if one wants to have variables with std equal to one.
WH_adaptive_fcmeans

weight.sys a string. Weights may add to one (‘SUM’) or their product is equal to 1 (‘PROD’, default).
theta a number. A parameter if weight.sys=‘SUM’, default is 2.
init.weights a string how to initialize weights: ‘EQUAL’ (default), all weights are the same, ‘RANDOM’, weights are initialised at random.
verbose A logic value (default is FALSE). If TRUE, details on computations are shown.

Value

a list with the results of the k-means of the set of Histogram-valued data x into k cluster.

Slots

solution A list. Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares criterion.
solution$IDX A vector. The clusters at which the objects are assigned.
solution$cardinality A vector. The cardinality of each final cluster.
solution$centers A MatH object with the description of centers.
solution$Crit A number. The criterion (Sum od square deviation from the centers) value at the end of the run.

quality A number. The percentage of Sum of square deviation explained by the model. (The higher the better)

References


Examples

results <- WH_adaptive.kmeans(x = BLOOD, k = 2, rep = 10,
                              simplify = TRUE, qua = 10, standardize = TRUE)

WH_adaptive_fcmeans Fuzzy c-means with adaptive distances for histogram-valued data

Description

Fuzzy c-means of a dataset of histogram-valued data using different adaptive distances based on the L2 Wasserstein metric.
**Usage**

```r
WH_adaptive_fcmeans(
  x,
  k = 5,
  schema,
  m = 1.6,
  rep,
  simplify = FALSE,
  qua = 10,
  standardize = FALSE,
  init.weights = "EQUAL",
  weight.sys = "PROD",
  theta = 2,
  verbose = FALSE
)
```

**Arguments**

- **x**: A MatH object (a matrix of distributionH).
- **k**: An integer, the number of groups.
- **schema**: An integer. 1=one weight per variable, 2=two weights per variables (one for each component: the mean and the variability component), 3=one weight per variable and per cluster, 4= two weights per variable and per cluster.
- **m**: A number grater than 0, a fuzziness coefficient (default \( m = 1.6 \)).
- **rep**: An integer, maximum number of repetitions of the algorithm (default \( rep = 5 \)).
- **simplify**: A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
- **qua**: An integer, if simplify=TRUE is the number of quantiles used for recodify the histograms.
- **standardize**: A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wasserstein based standard deviation. Use if one wants to have variables with std equal to one.
- **init.weights**: A string. (default='EQUAL'). EQUAL, all variables or components have the same weight; 'RANDOM', a random assignment is done.
- **weight.sys**: A string. (default='PROD') PROD, Weights product is equal to one. SUM, the weights sum up to one.
- **theta**: A number. (default=2) A parameter for the system of weights summing up to one.
- **verbose**: A logic value (default is FALSE). If TRUE some details are provided.

**Value**

The results of the fuzzy c-means of the set of Histogram-valued data \( x \) into \( k \) cluster.

- **solution**: A list. Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares deviation.
solution$membership
    A matrix. The membership degree of each unit to each cluster.
solution$IDX
    A vector. The crisp assignment to a cluster.
solution$cardinality
    A vector. The cardinality of each final cluster (after the crisp assignment).
solution$Crit
    A number. The criterion (Sum of square deviation from the prototypes) value at
    the end of the run.
quality
    A number. The percentage of Sum of square deviation explained by the model.
    (The higher the better)

Examples

```r
results <- WH_adaptive_fcmeans(
    x = BLOOD, k = 2, schema = 4, m = 1.5, rep = 3, simplify = TRUE,
    qua = 10, standardize = TRUE, init.weights = "EQUAL", weight.sys = "PROD"
)
```

---

**WH_fcmeans**

*Fuzzy c-means of a dataset of histogram-valued data*

**Description**

The function implements the fuzzy c-means for a set of histogram-valued data.

**Usage**

```r
WH_fcmeans(x, k, m = 1.6, rep, simplify = FALSE, qua = 10, standardize = FALSE)
```

**Arguments**

- `x` A MatH object (a matrix of distributionH).
- `k` An integer, the number of groups.
- `m` A number greater than 0, a fuzziness coefficient (default `m`=1.6).
- `rep` An integer, maximum number of repetitions of the algorithm (default `rep`=5).
- `simplify` A logical value (default is `FALSE`), if `TRUE` histograms are recomputed in order
to speed-up the algorithm.
- `qua` An integer, if `simplify=TRUE` is the number of quantiles used for recodify the
    histograms.
- `standardize` A logical value (default is `FALSE`). If `TRUE`, histogram-valued data are standard-
    ized, variable by variable, using the Wassertein based standard deviation. Use if
    one wants to have variables with std equal to one.

**Value**

A list with the results of the fuzzy c-means of the set of Histogram-valued data `x` into `k` cluster.
 Slots

 solution  A list. Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares deviation.

 solution$membership  A matrix. The membership degree of each unit to each cluster.

 solution$IDX  A vector. The crisp assignment to a cluster.

 solution$cardinality  A vector. The cardinality of each final cluster (after the crisp assignment).

 solution$Crit  A number. The criterion (Sum of square deviation from the prototypes) value at the end of the run.

 quality  A number. The percentage of Sum of square deviation explained by the model. (The higher the better)

 Examples

 results <- WH_fcmeans(x = BLOOD, k = 2, m = 1.5, rep = 10, simplify = TRUE, qua = 10, standardize = TRUE)

 Description

 The function implements a Hierarchical clustering for a set of histogram-valued data, based on the L2 Wasserstein distance. Extends the hclust function of the stat package.

 Usage

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

 Arguments

 x            A MatH object (a matrix of distributionH).
 simplify     A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
 qua          An integer, if simplify=TRUE is the number of quantiles used for recodify the histograms.
standardize: A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wasserstein based standard deviation. Use if one wants to have variables with std equal to one.

distance: A string default "WDIST" the L2 Wasserstein distance (other distances will be implemented).

method: A string, default="complete", is the the agglomeration method to be used. This should be (an unambiguous abbreviation of) one of "ward.D", "ward.D2", "single", "complete", "average" (= UPGMA), "mcquitty" (= WPGMA), "median" (= WPGMC) or "centroid" (= UPGMC).

Value
An object of class hclust which describes the tree produced by the clustering process.

References

See Also
hclust of stat package for further details.

Examples

results <- WH_hclust(x = BLOOD, simplify = TRUE, method = "complete")
plot(results) # it plots the dendrogram
cutree(results, k = 5) # it returns the labels for 5 clusters

---

**WH_kmeans**

K-means of a dataset of histogram-valued data

Description
The function implements the k-means for a set of histogram-valued data.

Usage

WH_kmeans(
  x,
  k,
  rep = 5,
  simplify = FALSE,
  qua = 10,
  standardize = FALSE,
  verbose = FALSE
)
WH_kmeans

Arguments

- **x**: A MatH object (a matrix of distributions).
- **k**: An integer, the number of groups.
- **rep**: An integer, maximum number of repetitions of the algorithm (default rep=5).
- **simplify**: A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
- **qua**: An integer, if simplify=TRUE is the number of quantiles used for recodify the histograms.
- **standardize**: A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wasserstein based standard deviation. Use if one wants to have variables with std equal to one.
- **verbose**: A logic value (default is FALSE). If TRUE, details on computations are shown.

Value

a list with the results of the k-means of the set of Histogram-valued data x into k cluster.

Slots

- **solution**: A list. Returns the best solution among the repetitions, i.e. the one having the minimum sum of squares criterion.
- **solution$IDX**: A vector. The clusters at which the objects are assigned.
- **solution$cardinality**: A vector. The cardinality of each final cluster.
- **solution$centers**: A MatH object with the description of centers.
- **solution$Crit**: A number. The criterion (Sum od square deviation from the centers) value at the end of the run.
- **quality**: A number. The percentage of Sum of square deviation explained by the model. (The higher the better)

References


Examples

```r
results <- WH_kmeans(
  x = BLOOD, k = 2, rep = 10, simplify = TRUE,
  qua = 10, standardize = TRUE, verbose = TRUE
)
```
The function extracts the L2 Wasserstein distance matrix from a MatH object.

Usage

\[
\text{WH\_MAT\_DIST}(x, \text{ simplify } = \text{ FALSE}, \text{ qua } = 10, \text{ standardize } = \text{ FALSE})
\]

Arguments

- \text{x} A MatH object (a matrix of distribution).
- \text{simplify} A logic value (default is FALSE), if TRUE histograms are recomputed in order to speed-up the algorithm.
- \text{qua} An integer, if \text{simplify}=TRUE is the number of quantiles used for recodify the histograms.
- \text{standardize} A logic value (default is FALSE). If TRUE, histogram-valued data are standardized, variable by variable, using the Wasserstein based standard deviation. Use if one wants to have variables with std equal to one.

Value

A matrix of squared L2 distances.

References


Examples

\[
\text{DMAT} \leftarrow \text{WH\_MAT\_DIST}(x = \text{BLOOD}, \text{ simplify } = \text{ TRUE})
\]
extract from a MatH Method

Description
This method overrides the "[" operator for a matH object.

Usage

```r
## S4 method for signature 'MatH'
x[i, j, ..., drop = TRUE]
```

Arguments

- `x`: a matH object
- `i`: a set of integer values identifying the rows
- `j`: a set of integer values identifying the columns
- `...`: not useful
- `drop`: a logical value inherited from the basic method "[" but not used (default=TRUE)

Value

A matH object

Examples

```r
D <- BLOOD # the BLOOD dataset
SUB_D <- BLOOD[c(1, 2, 5), c(1, 2)]
```
Index

* classes
  distributionH-class, 14
  MatH-class, 30

* distribution
  checkEmptyBins, 9
  compP, 11
  crwtransform, 12
  dotpW, 16
  is.registeredMH, 28
  kurtH, 29
  meanH, 31
  register, 39
  registerMH, 40
  skewH, 45
  stdH, 46
  WassSqDistH, 49

* package
  HistDAWass-package, 4

*, distributionH, distributionH-method (*-methods), 5
*, distributionH, numeric-method (*-methods), 5
*, numeric, distributionH-method (*-methods), 5
*-methods, 5
+
+, distributionH, distributionH-method (+), 6
+, distributionH, numeric-method (+), 6
+, numeric, distributionH-method (+), 6
-, distributionH, distributionH-method (minus), 32
-, distributionH, numeric-method (minus), 32
-, numeric, distributionH-method (minus), 32

[, 81
[, MatH, ANY, ANY, ANY-method (]), 81
[, MatH-method (]), 81

Age_Pyramids_2014, 6
Agronomique, 7
BLOOD, 7
BloodBRITO, 8
Center.cell.MatH, 8
Center.cell.MatH, MatH-method (Center.cell.MatH), 8
checkEmptyBins, 9
checkEmptyBins, distributionH-method (checkEmptyBins), 9
China_Month, 10
China_Seas, 10
compP, 11
compP, distributionH, numeric-method (compP), 11
compP, distributionH-method (compP), 11
compQ, 11
compQ, distributionH, numeric-method (compQ), 11
compQ, distributionH-method (compQ), 11
crwtransform, 12
crwtransform, distributionH-method (crwtransform), 12
data2hist, 13, 18, 44
distributionH, distributionH-class, 14
distributionH-class, 14
dotpW, 16
dotpW, distributionH, distributionH-method (dotpW), 16
dotpW, distributionH, numeric-method (dotpW), 16
dotpW, distributionH-method (dotpW), 16
dotpW, numeric, distributionH-method (dotpW), 16
DouglasPeucker, 17, 44
gate.cell.MatH, 18
get.cell.MatH, MatH, numeric, numeric-method
  (get.cell.MatH), 18
get.cell.MatH, MatH-method
  (get.cell.MatH), 18
get.distr, 18
get.distr, distributionH-method
  (get.distr), 18
get.histo, 19
get.histo, distributionH-method
  (get.histo), 19
get.m, 20
get.m, distributionH-method (get.m), 20
get.MatH.main.info, 20
get.MatH.main.info, MatH-method
  (get.MatH.main.info), 20
get.MatH.ncols, 21
get.MatH.ncols, MatH-method
  (get.MatH.ncols), 21
get.MatH.nrows, 21
get.MatH.nrows, MatH-method
  (get.MatH.nrows), 21
get.MatH.rownames, 22
get.MatH.rownames, MatH-method
  (get.MatH.rownames), 22
get.MatH.stats, 22
get.MatH.stats, MatH-method
  (get.MatH.stats), 22
get.MatH.varnames, 24
get.MatH.varnames, MatH-method
  (get.MatH.varnames), 24
get.s, 24
get.s, distributionH-method (get.s), 24
hclust, 78
hist, 14
HistDAWass (HistDAWass-package), 4
HistDAWass-package, 4
histogram, 14
HTS (HTS-class), 25
HTS-class, 25
HTS.exponential.smoothing, 25
HTS.moving.averages, 26
HTS.predict.knn, 27
initialize, distributionH-method
  (distributionH-class), 14
initialize, HTS-method (HTS-class), 25
initialize, MatH-method (MatH-class), 30
initialize, TdistributionH-method
  (TdistributionH-class), 48
initialize, TMatH-method (TMatH-class), 48
is.registeredMH, 28
is.registeredMH, MatH-method
  (is.registeredMH), 28
kurtH, 29
kurtH, distributionH-method (kurtH), 29
MatH (MatH-class), 30
MatH-class, 30
meanH, 15, 31
meanH, distributionH-method (meanH), 31
minus, 32
OzoneFull, 33
OzoneH, 33
plot, distributionH-method
  (plot-distributionH), 34
plot, HTS-method (plot-HTS), 34
plot, MatH-method (plot-MatH), 35
plot, TdistributionH-method
  (plot-TdistributionH), 36
plot-distributionH, 34
plot-HTS, 34
plot-MatH, 35
plot-TdistributionH, 36
plot_errors, 38
plotPredVsObs, 37
register, 39
register, distributionH, distributionH-method
  (register), 39
register, distributionH-method
  (register), 39
registerMH, 40
registerMH, MatH-method (registerMH), 40
RetHTS, 41
rQQ, 42
rQQ, distributionH, distributionH-method
  (rQQ), 42
rQQ, distributionH-method (rQQ), 42
set.cell.MatH, 43
set.cell.MatH, distributionH, MatH, numeric, numeric-method
  (set.cell.MatH), 43
ShortestDistance, 43
show, 44
show,distributionH-method (show), 44
show, Math-method (show=Math), 44
show=Math, 44
skewH, 45
skewH,distributionH-method (skewH), 45
stations_coordinates, 46
stdH, 15, 46
stdH,distributionH-method (stdH), 46
subsetHTS, 47
TdistributionH (TdistributionH-class), 48
TdistributionH-class, 48
TMatH (TMatH-class), 48
TMatH-class, 48
WassSqDistH, 49
WassSqDistH,distributionH,distributionH-method  (WassSqDistH), 49
WassSqDistH,distributionH-method (WassSqDistH), 49
WH.1d.PCA, 50
WH.bind, 51
WH.bind, Math, Math-method (WH.bind), 51
WH.bind, Math-method (WH.bind), 51
WH.bind.col, 52, 52
WH.bind.col, Math, Math-method  (WH.bind.col), 52
WH.bind.col, Math-method (WH.bind.col), 52
WH.bind.row, 52, 53
WH.bind.row, Math, Math-method  (WH.bind.row), 53
WH.bind.row, Math-method (WH.bind.row), 53
WH.correlation, 54
WH.correlation, Math-method  (WH.correlation), 54
WH.correlation2, 55
WH.correlation2, Math, Math-method  (WH.correlation2), 55
WH.correlation2, Math-method  (WH.correlation2), 55
WH.mat.prod, 56
WH.mat.prod, Math, Math-method  (WH.mat.prod), 56
WH.mat.prod, Math-method (WH.mat.prod), 56
WH.mat.sum, 56
WH.mat.sum, Math, Math-method  (WH.mat.sum), 56
WH.mat.sum, Math-method (WH.mat.sum), 56
WH MultiplePCA, 57
WH.plot_multiple_indivs, 58
WH.plot_multiple_Spanish.funs, 59
WH.regression.GOF, 60
WH.regression.two.components, 61
WH.regression.two.components.predict, 62
WH.SSQ, 63
WH.SSQ, Math-method (WH.SSQ), 63
WH.SSQ2, 64
WH.SSQ2, Math, Math-method (WH.SSQ2), 64
WH.SSQ2, Math-method (WH.SSQ2), 64
WH.var.covar, 65
WH.var.covar, Math-method  (WH.var.covar), 65
WH.var.covar2, 66
WH.var.covar2, Math, Math-method  (WH.var.covar2), 66
WH.var.covar2, Math-method  (WH.var.covar2), 66
WH.vec.mean, 67
WH.vec.mean, Math-method (WH.vec.mean), 67
WH.vec.sum, 67
WH.vec.sum, Math-method (WH.vec.sum), 67
WH_2d_Adaptive_Kohonen_maps, 68
WH_2d_Kohonen_maps, 71
WH_adaptive.kmeans, 73
WH_adaptive.fcmeans, 74
WH_fcmeans, 76
WH_hclust, 77
WH_kmeans, 78
WH_MAT_DIST, 80