Package 'ODP'

January 22, 2025

Title Exploration for Random Objects Using Distance Profiles

URL https://github.com/yqgchen/ODP

BugReports https://github.com/yqgchen/ODP/issues

Version 0.1.1

Date 2024-03-26

Maintainer Yaqing Chen <yqchen@stat.rutgers.edu>

Description Provides tools for exploratory data analysis of random objects lying in metric spaces. References: Dubey, P., Chen, Y. & Müller, H.-G. (2022) <doi:10.48550/arXiv.2202.06117>.

License BSD_3_clause + file LICENSE

Encoding UTF-8

LazyData false

Imports frechet,
ggplot2,
ggrepel,
grDevices,
pracma,
stats,
utils

Suggests testthat

RoxygenNote 7.2.3

R topics documented:

CheckDistmat *Check distance matrices*

Description

Check whether the input distance matrix or its sub square matrix is symmetric and has zero diagonals.

Usage

```
CheckDistmat(distmat)
```

Arguments

distmat	An n-by-n matrix holding the pairwise distances between observations.

expit *Expit function*

Description

Compute the value of $\exp(x)/(1+\exp(x))$ for given x.

Usage

```
expit(x)
```

Arguments

x	A vector holding the real valued input.

Value

A vector holding the result.

GetDistPrfl *Compute distance profiles with respect to a sample of random objects*

Description

Compute the distribution of $d(x, X)$ for given x. For $x = X_i$ within the training data $\{X_i\}_{i=1}^n$, the distribution of $d(x, X)$ is estimated based on $\{d(x, X_j)\}_{j \neq i}$. For a new x out of the training data, the distribution of $d(x, X)$ is estimated based on $\{d(x, X_j)\}_{j=1}^n$.

Usage

```
GetDistPrfl(
  distmat,
  data,
  distfun = NULL,
  newdata = NULL,
  type = "qf",
  optns = list(),
  ...
)
```

Arguments

distmat	An $(n + m)$-by-n matrix holding the pairwise distances between observations in training data of size n and possibly new data of size m to the training data. If there is no new data of interest, then distmat should be a symmetric matrix of dimension n. At least one of data and distmat should be input. If both are given, distmat overrides data.
data	Training data with respect to the law of which distance profiles are of interest. Either a matrix or data frame with n rows where each row holds one observation in the training data, or a list (but not a data frame) of length n where each element holds one observation.
distfun	A function with two arguments computing the distance between two observations, which is used in GetPairDist. Default: Euclidean distance.
newdata	Optional new data, at which distance profiles are to be computed with respect to the law of data. Either a matrix or data frame with m rows and the same number of columns as data where each row holds one new observation in addition to the observations in data. or a list (but not a data frame) of length m where each element is of the same format as those in data and holds one new observation. This can only be specified if data but not distmat is given.
type	Character specifying the type of results to be computed: 'den' for densities, 'cdf' for cumulative distribution functions, 'qf' (default) for quantile functions, and 'all' for all of the three aforementioned types.
optns	A list of control parameters specified by list(name=value). See 'Details'.
...	Optional arguments of distfun.

Details

Cumulative distribution function and quantile function representations of distance profiles are obtained as the right-continuous empirical distribution functions and their left-continuous inverse, respectively. If density representation of distance profiles is of interest (`type = 'den'` or `type = 'all'`), density estimation is performed using `CreateDensity`. Available control options are `userBwMu`, `nRegGrid`, `delta`, `kernel`, `infSupport`, `outputGrid`, `nqSup` and `qSup`. See `CreateDensity` for the details about the options other than `nqSup` and `qSup`.

nqSup The number of equidistant points in the support grid of the quantile functions. Default: 101.

qSup The support grid of the quantile functions; it overrides `nqSup`. Default: `seq(0,1,length.out=optns$nqSup)`.

Value

A list. All three fields `den` (for the densities), `cdf` (for the cdfs) and `qf` (for the quantile functions) will be included if `type == 'all'`, and the corresponding field alone out of the three otherwise.

`den`	A list of n or $(n+m)$ fields if `optns$outputGrid` is unspecified, each of which is a list of three fields, `bw`, `x` and `y`; see 'Value' of `CreateDensity` for further details. If `optns$outputGrid` is specified, it is a list of three fields, `bwvec`, `x` and `ymat`, where `bwvec` is a vector of length n or $(n+m)$ holding the bandwidths used for smoothing, `x` is the (common) support grid of the n or $(n+m)$ densities specified by `optns$outputGrid`, and `ymat` is a matrix with n or $(n+m)$ rows, each row holding the values of the density for one subject.
`cdf`	A list of n or $(n+m)$ fields if `optns$outputGrid` is unspecified, each of which is a list of two fields, `x` and `y`, which are two vectors holding the support grid and the corresponding values of the cdf, respectively. If `optns$outputGrid` is specified, it is a list of two fields, `x` and `ymat`, where `x` is the (common) support grid of the n or $(n+m)$ cdfs specified by `optns$outputGrid`, and `ymat` is a matrix with n or $(n+m)$ rows, each row holding the values of the cdf for one subject.
`qf`	A list of two fields, `x` and `ymat`, where `x` is a vector holding the (common) support grid of the n or $(n+m)$ quantile functions, and `ymat` is a matrix with n or $(n+m)$ rows, each row holding the values of the quantile function for one subject.
`n`	The sample size of the training data.
`distmat`	An $(n+m)$-by-n matrix holding the pairwise distances between observations in training data of size n and possibly new data of size m to the training data. This will be output only if `data` but not `distmat` is given in the input.

Examples

```
d <- 2
n <- 10
m <- 5
data <- matrix( rnorm( n * d ), ncol = d )
newdata <- matrix( rnorm( m * d ), ncol = d )
distmat <- as.matrix( stats::dist( rbind(data,newdata) ) )[, 1:n, drop=FALSE]
## with input distmat
res <- GetDistPrfl( distmat = distmat )
## Or with input data and newdata
res2 <- GetDistPrfl( data = data, newdata = newdata )
```

GetEDF *Compute empirical distribution functions*

Description

For a sample of real-valued random variables $X_1, \ldots, X_n$, compute the empirical distribution function $\hat{F}(x) = \frac{1}{n} \sum_{i=1}^{n} \chi\{X_i \leq x\}$, where $\chi\{\cdot\}$ denotes the indicator function.

Usage

```
GetEDF(x, sup, returnSup = FALSE)
```

Arguments

x	A numeric vector holding the data.
sup	A numeric vector holding the support grid of the empirical distribution function.
returnSup	Logical; indicating whether sup is to be returned. Default: FALSE.

Value

A vector holding the empirical distribution function evaluated on sup if returnSup is FALSE; A list containing two fields x and y holding the support grid and the values of the empirical distribution function, respectively, if returnSup is TRUE.

Examples

```
x <- runif(50)
edf <- GetEDF( x = x, sup = seq(0,1,0.01), returnSup = TRUE )
```

GetEQF *Compute empirical quantile functions*

Description

For a sample of real-valued random variables $X_1, \ldots, X_n$, compute the empirical quantile function $\hat{Q}(p) = \inf\{x : \hat{F}(x) \geq p\}$, where $\hat{F}(\cdot)$ denotes the corresponding empirical distribution function.

Usage

```
GetEQF(x, qSup, returnSup = FALSE)
```

Arguments

x	A numeric vector holding the data.
qSup	A numeric vector holding the support grid of the empirical quantile function.
returnSup	Logical; indicating whether sup is to be returned. Default: FALSE.

Value

A vector holding the empirical quantile function evaluated on `qSup` if `returnSup` is `FALSE`; A list containing two fields `x` and `y` holding the support grid and the values of the empirical quantile function, respectively, if `returnSup` is `TRUE`.

Examples

```
x <- runif(50)
edf <- GetEQF( x = x, qSup = seq(0,1,0.01), returnSup = TRUE )
```

`GetPairDist` *Get pairwise distances between observations (data objects)*

Description

Get pairwise distances between observations (data objects)

Usage

```
GetPairDist(data, distfun = NULL, newdata = NULL, ...)
```

Arguments

`data`	Either a matrix or data frame with n rows where each row holds one observation in the (training) data, or a list (but not a data frame) of length n where each element holds one observation.
`distfun`	A function with (at least) two arguments `x` and `y` computing the distance between two observations `x` and `y`. Default: Euclidean distance `function(x,y){sqrt(sum((x-y)^2))}`.
`newdata`	Either a matrix or data frame with m rows and the same number of columns as `data` where each row holds one new observation in addition to the observations in `data`, or a list (but not a data frame) of length m where each element is of the same format as those in `data` and holds one new observation. Pairwise distances from observations in `newdata` to those in `data` are to be computed.
`...`	Optional additional arguments of `distfun`.

Value

An $(n+m)$-by-n matrix holding the pairwise distances between observations.

Examples

```
## 2-Wasserstein distance for distributions on the real line
mu <- 1:4
qSup <- seq(0,1,len=1001)[-c(1,1001)]
data <- lapply( mu, qnorm, p = qSup )
distmat <- GetPairDist( data = data, distfun = l2metric, sup = qSup )
```

GetPrflHomTestStat *Compute the test statistic of the distance profile based two-sample homogeneity test for random objects*

Description

Compute the test statistic of the two-sample test for random objects based on comparing the distance profiles.

Usage

```
GetPrflHomTestStat(distmat, n, nRegGrid = 101)
```

Arguments

distmat	An $(n+m)$-by-$(n+m)$ matrix holding the pairwise distances between observations in the two samples of sizes n and m, respectively.
n	Size of the first sample.
nRegGrid	Number of equidistant grid points from the minimum to the maximum of all pairwise distances, on which the integrals of difference between distance profiles will be taken. Default: 101.

Value

The value of the test statistic.

Examples

```
n <- m <- 50
data <- rep( rnorm(n,0,1), 2 )
distmat <- abs( matrix( data, nr = n+m, nc = n+m ) - matrix( data, nr = n+m, nc = n+m, byrow = TRUE ) )
stat <- GetPrflHomTestStat( distmat = distmat, n = n )
```

GetTpRank *Compute transport ranks from a distance matrix or data*

Description

Compute transport ranks based on a distance matrix or data. For $x = X_i$ within the training data $\{X_i\}_{i=1}^n$, the transport rank of x is estimated by comparing the distance profile of x with the distance profiles of $\{X_j\}_{j \neq i}$. For a new x out of the training data, the transport rank of x is estimated by comparing the distance profile of x with the distance profiles of $\{X_j\}_{j=1}^n$.

Usage

```
GetTpRank(
  distmat,
  data,
  distfun = NULL,
  newdata = NULL,
  output_profile = TRUE,
  type = "qf",
  optns = list(),
  ...
)
```

Arguments

distmat	An $(n + m)$-by-n matrix holding the pairwise distances between observations in training data of size n and possibly new data of size m to the training data. If there is no new data of interest, then distmat should be a symmetric matrix of dimension n. At least one of data and distmat should be input. If both are given, distmat overrides data.
data	Training data with respect to the law of which distance profiles are of interest. Either a matrix or data frame with n rows where each row holds one observation in the training data, or a list (but not a data frame) of length n where each element holds one observation.
distfun	A function with two arguments computing the distance between two observations, which is used in GetPairDist. Default: Euclidean distance.
newdata	Optional new data, at which distance profiles are to be computed with respect to the law of data. Either a matrix or data frame with m rows and the same number of columns as data where each row holds one new observation in addition to the observations in data. or a list (but not a data frame) of length m where each element is of the same format as those in data and holds one new observation. This can only be specified if data but not distmat is given.
output_profile	Logical, whether distance profiles need to be returned. Default: TRUE.
type	Character specifying the type of representations of distance profiles to be output: 'qf' (default) for quantile functions, and 'all' for densities, cumulative distribution functions, and quantile functions.
optns	A list of options control parameters specified by list(name=value). See 'Details'.
...	Optional arguments of distfun.

Details

This function is an intergration of integrating GetDistPrfl and GetTpRankFromDistPrfl. Cumulative distribution function and quantile function representations of distance profiles are obtained as the right-continuous empirical distribution functions and their left-continuous inverse, respectively. If density representation of distance profiles is of interest (type = 'den' or type = 'all'), density estimation is performed using CreateDensity. Available control options are userBwMu, nRegGrid, delta, kernel, infSupport, outputGrid, nqSup and qSup. See CreateDensity for the details about the options other than nqSup and qSup.

nqSup The number of equidistant points in the support grid of the quantile functions. Default: 101.

qSup The support grid of the quantile functions; it overrides nqSup. Default: `seq(0,1,length.out = optns$nqSup)`.

Value

A list of the following fields:

`rank`	A vector holding the transport ranks.
`n`	The size of the training data.
`profile`	A list of the following: **den** Density representations of distance profiles, only returned if `type='all'`. **cdf** Cumulative distribution function representations of distance profiles, only returned if `type='all'`. **qf** Quantile function representations of distance profiles.
`distmat`	An $(n+m)$-by-n matrix holding the pairwise distances between observations in training data of size n and possibly new data of size m to the training data. This will be output only if `data` but not `distmat` is given in the input.

See Also

`GetDistPrfl` and `GetTpRankFromDistPrfl`.

Examples

```
d <- 2
n <- 10
m <- 5
data <- matrix( rnorm( n * d ), ncol = d )
newdata <- matrix( rnorm( m * d ), ncol = d )
distmat <- as.matrix( stats::dist( rbind(data,newdata) ) )[, 1:n, drop=FALSE]
## with input distmat
res <- GetTpRank( distmat = distmat )
## Or with input data and newdata
res2 <- GetTpRank( data = data, newdata = newdata )
```

GetTpRankFromDistPrfl *Compute transport ranks from distance profiles*

Description

Compute transport ranks based on the distance profiles represented by quantile functions. For $x = X_i$ within the training data $\{X_i\}_{i=1}^n$, the transport rank of x is estimated by comparing the distance profile of x with the distance profiles of $\{X_j\}_{j\neq i}$. For a new x out of the training data, the transport rank of x is estimated by comparing the distance profile of x with the distance profiles of $\{X_j\}_{j=1}^n$.

Usage

```
GetTpRankFromDistPrfl(qDistPrfl, n, fun = expit)
```

Arguments

qDistPrfl	A list of two fields, x and ymat, holding the quantile functions corresponding to the distance profiles for each observation in training data and possibly new data. **x** A vector holding the (common) support grid of $(n + m)$ quantile functions, in a strictly increasing order between 0 and 1. **ymat** A matrix with $(n + m)$ rows, each row holding the values of the quantile function for one data point. Here, n is the size of the sample with respect to which distance profiles have been obtained and m is the size of the new sample in addition distance profiles have been obtained with respect to the sample of size n.
n	The size of the sample with respect to which distance profiles have been obtained, i.e. n in the explanation of qDistPrfl. Default: nrow(qDistPrfl$ymat).
fun	A function giving the monotonic transformation applied for certain purpose, e.g., making ranks lying between 0 and 1. Default: expit.

Details

The argument qDistPrfl can be obtained by using GetDistPrfl with type = 'qf' or type = 'all', and the field named 'qf' in the output list yields input of qDistPrfl.

Value

A vector holding the transport ranks.

Examples

```
d <- 2
n <- 10
m <- 5
data <- matrix( rnorm( n * d ), ncol = d )
newdata <- matrix( rnorm( m * d ), ncol = d )
distmat <- as.matrix( stats::dist( rbind(data,newdata) ) )[, 1:n, drop=FALSE]
profile <- GetDistPrfl( distmat = distmat )
res <- GetTpRankFromDistPrfl( qDistPrfl = profile$qf, n = n )
```

---

l2metric *L^2 metric between square integrable functions.*

Description

L^2 metric between square integrable functions.

Usage

```
l2metric(x, y, sup)
```

Arguments

x, y	Vectors holding the values of two functions evaluated on sup.
sup	Support grid.

Value

The L^2 metric between the two functions.

`MakeObjMdsPlot` *Make an object MDS plot*

Description

Make a plot of 2-dimensional classical metric multidimensional scaling of the data based on the pairwise distance matrix or a scatterplot of 2-dimensional Euclidean data.

Usage

```
MakeObjMdsPlot(
  distmat,
  data,
  color_by,
  nGroup,
  shape_by = NULL,
  shape = 3,
  id,
  id_size = 3,
  xlab = "Coordinate 1",
  ylab = "Coordinate 2",
  colorlab = NULL,
  shapelab = NULL,
  use_default_colors = TRUE
)
```

Arguments

`distmat`	An n-by-n symmetric matrix holding the pairwise distances between observations.
`data`	An n-by-2 matrix or data frame holding the data; This is only applicable for 2-dimensional Euclidean data and overrides `distmat` if there is a proper input.
`color_by`	A vector of length n holding the variable according to which colors are assigned to each point. If missing, all points will be in black.
`nGroup`	Number of groups for optional grouping according to `color_by`. If given, the sample is divided into `nGroup` groups according to quantiles of `color_by`, which should be numeric in this case; the i-th group contains the points with the corresponding value in `color_by` falling in $(q_{1-i/\texttt{nGroup}}, q_{1-(i-1)/\texttt{nGroup}}]$, except for $i =$ `nGroup`, for which the corresponding interval is $[q_0, q_{1/\texttt{nGroup}}]$. Here, q_α is the α-quantile of `color_by`, given by `quantile(color_by,alpha)`.
`shape_by`	A vector of length n holding the variable according to which shapes are assigned to each point. Default: `NULL`.
`shape`	Shape of points when `shape_by` is `NULL`. Default: 3. See `geom_point` for details.
`id`	A vector of length n holding the label of each point. Default: `rownames(data)` if `data` is given and `rownames(distmat)` otherwise.

id_size	Size of the text labels of points. Default: 3.
xlab, ylab	The labels of x-axis and y-axis. Default: 'Coordinate 1' and 'Coordinate 2'.
colorlab	The label of the color variable when color_by is given. Default: NULL.
shapelab	The label of the shape variable when shape_by is given. Default: NULL.
use_default_colors	Logical, whether to use default colors when color_by is given. Default: TRUE.

Value

A ggplot object.

See Also

GetTpRank, MakePrflMdsPlot

Examples

```
n <- 100
p <- 2
set.seed(1)
data <- matrix( rnorm( n * p ), ncol = p )
res <- GetTpRank( data = data )
pl <- MakeObjMdsPlot( data = data, color_by = res$rank, nGroup = 10 )
```

MakePrflMdsPlot *Make a profile MDS plot*

Description

Make a plot of 2-dimensional classical metric multidimensional scaling of the data based on the distance profiles with respect to 2-Wasserstein metric.

Usage

```
MakePrflMdsPlot(qDistPrfl, color_by, nGroup, ...)
```

Arguments

qDistPrfl	A list of two fields, x and ymat, holding the quantile functions corresponding to the distance profiles for each observation in the data. **x** A vector holding the (common) support grid of n quantile functions, in a strictly increasing order between 0 and 1. **ymat** A matrix with n rows, each row holding the values of the quantile function for one data point.
color_by	A vector of length n holding the variable according to which colors are assigned to each point. Default: transport ranks computed based on qDistPrfl.

nGroup	Number of groups for optional grouping according to color_by. If given, the sample is divided into nGroup groups according to quantiles of color_by, which should be numeric in this case; the i-th group contains the points with the corresponding value in color_by falling in $(q_{1-i/\texttt{nGroup}}, q_{1-(i-1)/\texttt{nGroup}}]$, except for $i = \texttt{nGroup}$, for which the corresponding interval is $[q_0, q_{1/\texttt{nGroup}}]$. Here, q_α is the α-quantile of color_by, given by quantile(color_by,alpha). Default: 10 if the sample size n is larger than 10 and n otherwise.
...	Other arguments of MakeObjMdsPlot.

Value

A ggplot object.

See Also

GetTpRank, MakeObjMdsPlot

Examples

```
n <- 100
p <- 2
set.seed(1)
data <- matrix( rnorm( n * p ), ncol = p )
res <- GetTpRank( data = data )
pl <- MakePrflMdsPlot( qDistPrfl = res$profile$qf, color_by = res$rank, nGroup = 10 )
```

---

prflHomTest *Perform two-sample homogeneity test for random objects*

Description

Perform two-sample test for random objects based on comparing the distance profiles.

Usage

```
prflHomTest(distmat, n, nRegGrid = 101, nPerm = 999)
```

Arguments

distmat	An $(n+m)$-by-$(n+m)$ matrix holding the pairwise distances between observations in the two samples of sizes n and m, respectively.
n	Size of the first sample.
nRegGrid	Number of equidistant grid points from the minimum to the maximum of all pairwise distances, on which the integrals of difference between distance profiles will be taken. Default: 101.
nPerm	Number of permutations. Default: 999.

Value

The p-value based on permutations.

Examples

```
n <- m <- 50
data <- rep( rnorm(n,0,1), 2 )
distmat <- abs( matrix( data, nr = n+m, nc = n+m ) - matrix( data, nr = n+m, nc = n+m, byrow = TRUE ) )
pval <- prflHomTest( distmat = distmat, n = n )
```