Package 'qad'

December 14, 2022

Type Package

Title Quantification of Asymmetric Dependence

Version 1.0.4

Author Thimo Kasper <thimo.kasper@plus.ac.at>,
Florian Griessenberger <florian.griessenberger@plus.ac.at>,
Robert R. Junker <>,
Valentin Petzel <valetin.petzel@sbg.ac.at>,
Wolfgang Trutschnig <wolfgang.trutschnig@plus.ac.at>

Description A copula-based measure for quantifying asymmetry in dependence and associations. Documentation and theory about 'qad' is provided by the paper by Junker, Griessenberger & Trutschnig (2021, <doi:10.1016/j.csda.2020.107058>), and the paper by Trutschnig (2011, <doi:10.1016/j.jmaa.2011.06.013>).

License GPL-2

URL https://github.com/griefl/qad

BugReports https://github.com/griefl/qad/issues

Encoding UTF-8

Depends R (>= 2.10)

Imports ggplot2, data.table, copula, viridis, ggExtra, dplyr, cowplot, Rcpp (>= 1.0.6)

RoxygenNote 7.1.2

LinkingTo Rcpp

Suggests rmarkdown, reshape2, knitr, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

LazyData true

NeedsCompilation yes

Maintainer Nicolas Dietrich <nicolaspascal.dietrich@plus.ac.at>

Repository CRAN

Date/Publication 2022-12-14 16:50:02 UTC

R topics documented:

---

qad-package *Quantification of Asymmetric Dependencies*

Description

A copula-based measure for quantifying asymmetry in dependence and associations.

Details

Package:	qad
Type:	Package
Version:	1.0.0
Date:	2021-02-26

Author(s)

Florian Griessenberger: <florian.griessenberger@sbg.ac.at>,
Robert R. Junker: <Robert.Junker@sbg.ac.at>,
Valentin Petztel,
Wolfgang Trutschnig: <Wolfgang.Trutschnig@sbg.ac.at>

Examples

```
# #Create data set
# n <- 100
# x <- rnorm(n,0,2)
# y <- x^2 + rnorm(n)
```

```
# sample <- data.frame(x,y)
# plot(sample, pch = 16)
#
#
# #Function: qad()
# qad(sample, p.value = TRUE, p.value_asymmetry = TRUE)
# fit <- qad(sample)
# plot(fit)
# plot(fit, copula = TRUE)
# plot(fit, copula = TRUE, addSample = T)
#
# #Functions: summary(), coef()
# summary(fit)
# coef(fit)
#
# #Function: predict()
# values <- c(-2.4,1,0,2.6)
# predict.qad(fit, values = values, conditioned = 'x1')
# predict(fit, values, conditioned = "x1", pred_plot = TRUE, panel.grid = FALSE)
#
# values <- c(0.1,0.5)
# predict(fit, values, conditioned = "x2", copula = TRUE, pred_plot = TRUE)
#
# #Function: pairwise.qad and heatmap.qad
# df <- iris[1:4]
# mod <- pairwise.qad(df)
# heatmap.qad(mod, select = 'dependence')
```

D1 *Calculate the D1 distance between two dependence structures*

Description

Computation of the D1 distance between two checkerboard copulas A and B, corresponding to the random vectors (X1,Y1) and (X2,Y2), respectively. The function `D1()` computes the difference between the dependence structures of two random vectors. The function `D1.ECBC()` computes the D1-distance between two checkerboard copulas with the same resolution. The function `zeta1()` is defined as 3D1(A,Pi), where Pi denotes the independence copula and returns the dependence measure computed in qad.

Usage

```
D1(x1, y1, x2, y2, resolution = NULL)

D1.ECBC(A, B)

zeta1(X, Y, resolution = NULL)
```

Arguments

x1	a (non-empty) numeric vector of data values for the first random vector (first coordinate)
y1	a (non-empty) numeric vector of data values for the first random vector (second coordinate)
x2	a (non-empty) numeric vector of data values for the second random vector (first coordinate)
y2	a (non-empty) numeric vector of data values for the second random vector (second coordinate)
resolution	integer indicating the resolution of the checkerboard copula. (default = NULL)
A	Numeric matrix of dimension NxN indicating the mass of the first N-checkerboad copula
B	Numeric matrix of dimension NxN indicating the mass of the second N-checkerboad copula
X	Numeric vector of values in the first coordinate
Y	Numeric vector of values in the second coordinate

Value

`D1()` returns the D1 distance, introduced in (Trutschnig, 2011).

`D1.ECBC()` returns the D1-distance between to checkerboard copulas A and B with same resolution

`zeta1()` returns the directed dependence from x to y.

References

Trutschnig, W. (2011). On a strong metric on the space of copulas and its induced dependence measure. Journal of Mathematical Analysis and Applications. 384 (2), 690-705.

Junker, R.R., Griessenberger, F. and Trutschnig, W. (2021). Estimating scale-invariant directed dependence of bivariate distributions. Computational Statistics and Data Analysis, 153, 107058.

Examples

```
n <- 100
x1 <- runif(n)
y1 <- x1
x2 <- runif(n)
y2 <- 1-x2
D1(x1,y1,x2,y2)

n <- 1000
x <- runif(n, 0, 1)
y1 <- ifelse(x < 0.5, runif(length(x < 0.5), 0,0.5), runif(length(x >= 0.5), 0.5, 1))
y2 <- ifelse(x > 0.5, runif(length(x < 0.5), 0,0.5), runif(length(x >= 0.5), 0.5, 1))
A <- ECBC(x,y1, resolution = 50)
B <- ECBC(x,y2, resolution = 50)
```

```
#plot_density(A)
#plot_density(B)
D1.ECBC(A,B)
```

ECBC *Calculate empirical checkerboard copula*

Description

The function ECBC() computes the mass distribution of the empirical (checkerboard) copula, given a bi-variate sample. If resolution equals sample size, the bi-linearly extended empirical copula is returned. Note, if there are ties in the sample an adjusted empirical copula is calculated. The function ECBC.eval() evaluates the checkerboard copula at given points.

Usage

```
ECBC(X, Y, resolution = NULL)

ECBC.eval(CB, eval.points)
```

Arguments

X	Numeric vector of values in the first coordinate
Y	Numeric vector of values in the second coordinate
resolution	Integer indicating the resolution of the checkerboard aggregation, i.e. the number of vertical/horizontal strips of the checkerboard copula. (default = NULL, sets resolution = floor(sqrt(sample size)))
CB	A numeric mass matrix of a checkerboard copula (ECBC)
eval.points	A numeric matrix or data.frame indicating the eval.points (x,y)

Details

If the observations are drawn from a continuous distribution (no ties in the sample), the function ECBC() returns the commonly used empirical checkerboard copula. If there are ties in the sample, the empirical copula is adjusted and calculated in the following way:
Let (u_i,v_i) := (F_n(x_i),G_n(y_i)) be the pseudo-observations for i in {1,...,n} and (u_1',v_1'),..., (u_m',v_m') the distinct pairs of pseudo-observations with m leq n. Moreover set S_1:={0, u_1, ..., u_m_1} and S_2:={0, v_1,..., v_m_2} and define the quantities t_i,r_i and s_i for i=1,..., m by

$$t_i := sum_{j=1}^{n} 1_{(u_i', v_i')}(u_j, v_j)$$

$$r_i := sum_{j=1}^{n} 1_{u_i}(u_j)$$

$$s_i := sum_{j=1}^{n} 1_{v_i}(v_j)$$

where 1 defines the indicator function. Define the empirical subcopula A'_n: S_1 x S_2 to {0,1/n, ..., (n-1)/n,1} by

$$A_n'(s_1, s_2) = 1/n sum_{i=1}^{m} t_i * 1_{[0,s_1]x[0,s_2]}(u_i', v_i') = 1/n sum_{i=1}^{n} 1_{[0,s_1]x[0,s_2]}(u_i, v_i)$$

for all s_1 in S_1 and s_2 in S_2.
We extend the subcopula A'_n to a copula by defining the transformations w_i:[0,1]^2 to [u_i'-r_i/n,u_i'] x [v_i'-s_i/n,v_i'] by

$$w_i(x,y) = (u_i' - r_i/n + r_i * x/n, v_i' - s_i/n + s_i y/n)$$

and set the measure of the empirical copula mu_A_n^B := 1/n sum_i=1^m t_i mu_B^w_i, where B denotes the product copula.

Value

`ECBC()` returns a matrix with the mass distribution of the empirical (checkerboard) copula.

References

Deheuvels, P. (1979). La fonction de dependance empirique et ses proprietas: un test non parametrique d'independance, Acad. Roy. Belg. Bull. Cl. Sci., 5th Ser. 65, 274-292.

Li, X., Mikusinski, P. and Taylor, M.D. (1998). Strong approximation of copulas, Journal of Mathematical Analysis and Applications, 255, 608-623.

Genest, C., Neshlehova J.G. and Remillard, B. (2014). On the empirical multilinear copula process for count data. Bernoulli, 20 (3), 1344-1371.

Junker, R.R., Griessenberger, F. and Trutschnig, W. (2021). Estimating scale-invariant directed dependence of bivariate distributions. Computational Statistics and Data Analysis, 153, 107058.

Examples

```
##Generate data drawn from the product copula and compute the empirical (checkerboard) copula
n <- 100
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
mass <- ECBC(x,y, resolution = 10)
plot_density(mass)
mass <- ECBC(x,y, resolution = n)
plot_density(mass)

## Compute empirical checkerboard copula of a sample with ties and plot density
n <- 100
x <- sample(runif(n, -1, 1), n, replace=TRUE)
y <- x^2 + rnorm(n, 0, 1)
mass <- ECBC(x,y)
plot_density(mass)
```

`emp_c_copula` *The empirical checkerboard copula*

Description

The function `emp_c_copula()` computes the mass distribution of the empirical (checkerboard) copula, given a bivariate sample. `emp_c_copula_eval()` evaluates the the empirical (checkerboard) copula at given points. If `smoothing` = FALSE, the empirical copula is computed (if there are ties in the sample an adjusted empirical copula is computed), otherwise the empirical checkerboard copula - a smoothed version of the empirical copula - is computed. For more information of the calculations, see details.

Usage

```
emp_c_copula(X, smoothing = TRUE, resolution)

emp_c_copula_eval(X, u, smoothing = TRUE, resolution)
```

Arguments

`X`	a data frame with two columns containing the observations of the sample. Each row contains one observation.
`smoothing`	a logial indicating whether the checkerboard aggregation is computed (default = TRUE).
`resolution`	an integer indicating the resolution of the checkerboard aggregation, i.e. the number of vertical/horizontal strips of the checkerboard copula.
`u`	a data.frame with two columns containing the evaluation points. Each row consists of a x and y value.

Details

If the observations come from a distribution with continuous margins, i.e. there are no ties in the sample, the function `emp_c_copula()` gives the same result as the function `C.n()` in the `copula` package. If there are ties in the sample, the empirical copula is adjusted and calculated in the following way:
Let (u_i,v_i) := (F_n(x_i),G_n(y_i)) be the pseudo-observations for i in {1,. . . ,n} and (u_1',v_1'),. . . , (u_m',v_m') the distinct pairs of pseudo-observations with m leq n. Moreover set S_1:={0, u_1, . . . , u_m_1} and S_2:={0, v_1,. . . , v_m_2} and define the quantities t_i,r_i and s_i for i=1,. . . , m by

$$t_i := sum_{j=1}^{n} 1_{(u_i', v_i')}(u_j, v_j)$$
$$r_i := sum_{j=1}^{n} 1_{u_i}(u_j)$$
$$s_i := sum_{j=1}^{n} 1_{v_i}(v_j)$$

where 1 defines the indicator function. Define the empirical subcopula A'_n: S_1 x S_2 to {0,1/n, . . . , (n-1)/n,1} by

$$A_n'(s_1, s_2) = 1/n sum_{i=1}^{m} t_i * 1_{[0,s_1]x[0,s_2]}(u_i', v_i') = 1/n sum_{i=1}^{n} 1_{[0,s_1]x[0,s_2]}(u_i, v_i)$$

for all s_1 in S_1 and s_2 in S_2.
We extend the subcopula A'_n to a copula by defining the transformations w_i:[0,1]^2 to [u_i'-r_i/n,u_i'] x [v_i'-s_i/n,v_i'] by

$$w_i(x, y) = (u_i' - r_i/n + r_i * x/n, v_i' - s_i/n + s_i y/n)$$

and set the measure of the empirical copula mu_A_n^B := 1/n sum_i=1^m t_i mu_B^w_i, where B denotes the product copula.

The checkerboard aggregation is computed as usual (see references).

Value

`emp_c_copula()` returns a matrix with the mass distribution of the empirical (checkerboard) copula.

`emp_c_copula_eval()` returns a vector of evaluations of the empirical (checkerboard) copula.

Note

The calculation of the empirical copula with a high sample size (and resolution rate) can take time.

References

Deheuvels, P. (1979). La fonction de dependance empirique et ses proprietas: un test non parametrique d'independance, Acad. Roy. Belg. Bull. Cl. Sci., 5th Ser. 65, 274-292.

Li, X., Mikusinski, P. and Taylor, M.D. (1998). Strong approximation of copulas, Journal of Mathematical Analysis and Applications, 255, 608-623.

Genest, C., Neshlehova J.G. and Remillard, B. (2014). On the empirical multilinear copula process for count data. Bernoulli, 20 (3), 1344-1371.

`heatmap.qad` *Heatmap of dependence measures*

Description

The pairwise computed dependence measures (output of the function `pairwise.qad()`) are illustrated by a heatmap.

Usage

```
heatmap.qad(
  pw_qad,
  select = c("dependence", "max.dependence", "asymmetry"),
  fontsize = 4,
  significance = FALSE,
  use_p.adjust = TRUE,
  sign.level = 0.05,
  scale = "abs",
  color = "plasma",
  white_font = 0.7,
  rb_values = c(10, 0.315, 0.15),
  title = ""
)
```

Arguments

pw_qad	output of the function pairwise.qad().
select	a character indicating which dependence value is plotted. Options are c("dependence", "max.dependence", "asymmetry").
fontsize	a numeric specifying the font size of the values.
significance	a logical indicating whether significant values with respect to the (adjusted) qad p.values are denoted by a star.
use_p.adjust	a logical indicating if the adjusted p.values are used (default = TRUE).
sign.level	numeric value indicating the significance level.
scale	character indicating whether the heatmap uses a relative or absolute scale. Options are 'rel' or 'abs' (default).
color	Select the color palette. Options are c("plasma" (default), "viridis", "inferno", "magma", "cividis", "rainbow").
white_font	numeric between 0 and 1 denoting the start value for white text font (default = 0.7)
rb_values	a vector of size 3 with number of values, start value and end value in the rainbow colors space (if color = 'rainbow').
title	The text for the title

Details

If the output of pairwise.qad() contains p-values, significant values can be highlighted by stars by setting significance=TRUE.

Value

a heatmap

Examples

```
n <- 100
x1 <- runif(n, 0, 1)
x2 <- x1^2 + rnorm(n, 0, 0.1)
x3 <- runif(n, 0, 1)
x4 <- x3 - x2 + rnorm(n, 0, 0.1)
sample_df <- data.frame(x1,x2,x3,x4)
#Fit qad
model <- pairwise.qad(sample_df, p.value = FALSE)
heatmap.qad(model, select = "dependence", fontsize = 6)
```

`pairwise.qad` *Pairwise quantification of (asymmetric and directed) dependencies*

Description

Pairwise computation of the function qad(). qad() is applied on each pair of variables of a numeric data.frame.

Usage

```
pairwise.qad(
  data_df,
  remove.00 = FALSE,
  min.res = 3,
  p.value = TRUE,
  nperm = 1000,
  p.adjust.method = "fdr",
  p.value_asymmetry = FALSE,
  nboot = 1000
)
```

Arguments

`data_df`	a data frame containing numeric columns with the observations of the sample.
`remove.00`	a logical indicating whether double 0 entries should be excluded (default = FALSE)
`min.res`	an integer indicating the necessary minimum resolution of the checkerboard grid to compute qad, otherwise the result is NA (default = 3).
`p.value`	a logical indicating whether to return a p-value of rejecting independence (based on permutation).
`nperm`	an integer indicating the number of permutation runs.
`p.adjust.method`	a character string denoting the p.value correction method (see function p.adjust in stats). Options are c('holm', 'hochberg', 'hommel', 'bonferroni', 'BH', 'BY', 'fdr' (default), 'none')
`p.value_asymmetry`	a logical indicating whether a p-value (based on bootstrap) is computed for the measure of asymmetry.
`nboot`	an integer indicating the number of bootstrapping runs.

Value

a list, containing data.frames with the dependence measures, corresponding p.values, the resolution of the checkerboard aggregation and the number of removed double zero entries (only if remove.00 = TRUE). The output of pairwise.qad() can be illustrated using the function `heatmap.qad()`.

Examples

```
n <- 100
x1 <- runif(n, 0, 1)
x2 <- x1^2 + rnorm(n, 0, 0.1)
x3 <- runif(n, 0, 1)
x4 <- x3 - x2 + rnorm(n, 0, 0.1)
sample_df <- data.frame(x1,x2,x3,x4)
#Fit qad
model <- pairwise.qad(sample_df, p.value = TRUE, p.adjust.method = "fdr")
heatmap.qad(model, select = "dependence", fontsize = 6)
```

PlantDiversity *Plant diversity*

Description

A dataset containing 140 plots with estimated year of deglaciation and plant Shannon diversity.

Usage

```
PlantDiversity
```

Format

A data frame with 140 rows and 2 variables:

deglacYear estimated year of deglaciation

div Shannon diversity of plant ecology

Details

The data set contains the estimated year of deglaciation and plant diversity of 140 plots along the successional gradient of the forefield of the Ödenwinkelkees glacier (Austria). The plots (1 to 135) were established within the glacier forefield, the plots (136 to 140) were established outside the glacier forefield (i.e., age greater than 170 years). Since qad is invariant w.r.t. transformations the estimated year of these 5 plots is set to 1800.

#' @source doi:10.5194/we20952020

plot.qad *Plot conditional probabilites*

Description

Visualizes the conditional probabilities for each strip of the checkerboard copula in the copula setting or in the retransformed sample setting.

Usage

```
## S3 method for class 'qad'
plot(
  x,
  addSample = FALSE,
  copula = FALSE,
  density = FALSE,
  margins = FALSE,
  title = "",
  x.axis = "X1",
  y.axis = "X2",
  point.size = 0.9,
  panel.grid = TRUE,
  color = "plasma",
  rb_values = c(10, 0.315, 0.15),
  ...
)
```

Arguments

x	an object of class qad.
addSample	a logical indicating whether the observations are returned. In the copula setting the mass squares of the empirical copula are added too. (default = FALSE).
copula	a logical indicating whether the plot depicts the conditional probabilities of the empirical checkerboard copula or of the retransformed data setting (default = FALSE).
density	a logical indicating whether the density should be plotted instead of the conditional probabilites (default = FALSE). Only works in the copula setting, i.e. if copula = TRUE.
margins	a logical indicating whether the margin distribution is added in form of a rug plot.
title	The text for the title
x.axis	The text for the x-axis
y.axis	The text for the y-axis
point.size	a numeric specifying the point size of the sample (relevant if addSample = TRUE).

panel.grid	a logical indicating whether the panel grid is plotted. (default = TRUE)
color	a color palette of the viridis package or rainbow. options are c("viridis", "magma", "inferno", "plasma", "cividis", "rainbow")
rb_values	a vector of size 3 with number of values, start value and end value in the rainbow colors space.
...	some methods for this generic require additional arguments. None are used in this method.

Note

The conditional probabilities are constant at squares in the copula setting. If the squares are retransformed in the data setting, the resulting objects are rectangles.

Examples

```
## Example 1
n <- 100
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
sample <- data.frame(x, y)

#qad
fit <- qad(sample)
plot(fit, addSample = TRUE, copula = FALSE)

## Example 2
n <- 100
x <- runif(n, -1, 1)
y <- x^2 + rnorm(n, 0, 0.1)
sample <- data.frame(x, y)

#qad
fit <- qad(sample)
plot(fit, addSample = TRUE, copula = TRUE)
plot(fit, addSample = TRUE, copula = FALSE)
```

---

plot_density *Plot density of empirical checkerboard copula*

Description

Plots the density/mass of the empirical checkerboard copula.

Usage

```
plot_density(
  mass_matrix,
  density = TRUE,
  color = "plasma",
  rb_values = c(10, 0.315, 0.15)
)
```

Arguments

mass_matrix	a squared matrix containing the mass distribution, e.g. output of the function emp_c_copula().
density	a logical (TRUE = default) whether the density or the mass is plotted.
color	Select the color palette. Options are c("plasma" (default), "viridis", "inferno", "magma", "cividis").
rb_values	a vector of size 3 with number of values, start value and end value in the rainbow colors space.

Value

a density plot (or mass distribution)

Examples

```
n <- 1000
x <- runif(n,0,1)
y <- runif(n,0,1)
plot(x,y,pch = 16)

mass <- ECBC(x,y,resolution = 10)
plot_density(mass, density=TRUE)
plot_density(mass, density=FALSE)
```

---

predict.qad *Predict conditional probabilities*

Description

The function predict.qad() can be utilized to predict the probabilities of the event that Y lies in specific intervals given X=x, or vice versa. Thereby, the intervals are calculated as retransformed intervals (by using the empirical quantil function) defining the checkerboard grid. Additionally, the mass of the conditional distribution functions are plotted. The prediction can be computed in the sample setting as well as in the copula setting (pseudo-observation in the unit square).

Usage

```
## S3 method for class 'qad'
predict(
  object,
  values,
  conditioned = "x1",
  nr_intervals = NULL,
  prediction_interval = NULL,
  copula = FALSE,
  pred_plot = FALSE,
  panel.grid = TRUE,
  ...
)
```

Arguments

object	an object of class 'qad', which determines the underlying checkerboard aggregation.
values	a vector containing the x or the y values for which the conditional probabilities should be predicted.
conditioned	a character specifying on which variable is conditioned. Options are "x1" (default) or "x2".
nr_intervals	an integer, which determines a different number of intervals for the prediction (only possible in the copula setting).
prediction_interval	a vector specifying the interval boundaries for which the conditional probability is computed. Options are NULL (default) to predict the conditional probabilities for all intervals or a vector c(lower boundary, upper boundary) indicating the boundaries.
copula	a logical (default = FALSE) determining whether the empirical checkerboard copula is used or the retransformed data.
pred_plot	a logical indicating if the conditional probabilities are plotted.
panel.grid	a logical indicating whether the panel.grid is plotted.
...	some methods for this generic require additional arguments. None are used in this method.

Value

a list containing a data.frame with the computed intervals (lower and upper boundaries) and the prediction probabilities (i.e., the probability that Y lies in the interval Ii given X = x). Furthermore, a heatmap depicting the mass of the conditional distribution functions is returned.

Note

Predictions are only possible for values within the range of the sample (or between 0 and 1 in the copula setting). For given values exceeding the range NA is returned.

Examples

```
set.seed(1)
n <- 100
x <- runif(n, -4 ,4)
y <- x^2 + rnorm(n, 0, 1)
sample <- data.frame(x, y)

##(Not Run)
qad.fit <- qad(sample)
predict.qad(qad.fit, values = c(-2,0.6), conditioned = "x1", pred_plot = TRUE)
predict.qad(qad.fit, values = c(1,9), conditioned = "x2", pred_plot = TRUE)
predict.qad(qad.fit, values = c(-2,0.6), conditioned = "x1", pred_plot = FALSE,
        nr_intervals = 4)
predict.qad(qad.fit, values = c(-2,0.6), conditioned = "x1", pred_plot = FALSE,
            prediction_interval = c(4,6))
predict.qad(qad.fit, values = c(4,0.6), conditioned = "x2", pred_plot = FALSE,
            prediction_interval = c(2,3))

qad.pred <- predict.qad(qad.fit, values = c(-2,0.6), conditioned = "x1", pred_plot = FALSE)
qad.pred$prediction
qad.pred$plot
```

---

qad *Measure of (asymmetric and directed) dependence*

Description

Quantification of (asymmetric and directed) dependence structures between two random variables X and Y.

Usage

```
qad(x, ...)

## S3 method for class 'data.frame'
qad(
  x,
  resolution = NULL,
  p.value = TRUE,
  nperm = 1000,
  p.value_asymmetry = FALSE,
  nboot = 1000,
  print = TRUE,
  remove.00 = FALSE,
  ...
)
```

```
## S3 method for class 'numeric'
qad(
  x,
  y,
  resolution = NULL,
  p.value = TRUE,
  nperm = 1000,
  p.value_asymmetry = FALSE,
  nboot = 1000,
  print = TRUE,
  remove.00 = FALSE,
  ...
)
```

Arguments

x	a data.frame containing two columns with the observations of the bi-variate sample or a (non-empty) numeric vector of data values
...	Further arguments passed to 'qad' will be ignored
resolution	an integer indicating the number of strips for the checkerboard aggregation (see ECBC). We recommend to use the default value (resolution = NULL)
p.value	a logical indicating whether to return a p-value of rejecting independence (based on permutation).
nperm	an integer indicating the number of permutation runs (if p.value = TRUE)
p.value_asymmetry	a logical indicating whether to return a (heuristic) p-value for the measure of asymmetry (based on bootstrap).
nboot	an integer indicating the number of runs for the bootstrap.
print	a logical indicating whether the result of qad is printed.
remove.00	a logical indicating whether double 0 entries should be excluded (default = FALSE)
y	a (non-empty) numeric vector of data values.

Details

qad is the implementation of a strongly consistent estimator of the copula based dependence measure zeta_1 introduced in Trutschnig 2011. We first compute the empirical copula of a two-dimensional sample, aggregate it to the so called empirical checkerboard copula (ECBC), and calculate zeta_1 of the ECBC and its transpose. In order to test for independence (in both directions), a built-in p-value is implemented (a permutation test with nperm permutation runs to estimate the p-value). Furthermore, a (heuristic) bootstrap test with nboot runs can be applied to estimate a p-value for the measure of asymmetry a.

Value

qad returns an object of class qad containing the following components:

`data`	a data.frame containing the input data.
`q(X,Y)`	influence of X on Y
`q(Y,X)`	influence of Y on X
`max.dependence`	maximal dependence
`results`	a data.frame containing the results of the dependence measures.
`mass_matrix`	a matrix containing the mass distribution of the empirical checkerboard copula.
`resolution`	an integer containing the used resolution of the checkerboard aggregation.
`n`	Sample size.

References

Trutschnig, W. (2011). On a strong metric on the space of copulas and its induced dependence measure, Journal of Mathematical Analysis and Applications 384, 690-705.

Junker, R., Griessenberger, F. and Trutschnig, W. (2021). Estimating scale-invariant directed dependence of bivariate distributions. Computational Statistics and Data Analysis, 153.

See Also

A tutorial can be found at http://www.trutschnig.net/software.html.

Examples

```
#Example 1 (independence)

n <- 100
x <- runif(n,0,1)
y <- runif(n,0,1)
sample <- data.frame(x,y)
qad(sample)

###

#Example 2 (mutual complete dependence)

n <- 500
x <- runif(n,0,1)
y <- x^2
sample <- data.frame(x,y)
qad(sample)

#Example 3 (complete dependence)

n <- 1000
x <- runif(n,-10,10)
y <- sin(x)
```

```
sample <- data.frame(x,y)
qad(sample)

#Example 4 (Asymmetry)

n <- 100
x <- runif(n,0,1)
y <- (2*x) %% 1
qad(x, y)
```

---

qad_distribution *Distribution of qad (H0: independence)*

Description

Distribution function - P_H0(qad <= q) - and quantile function for the qad distribution with regard to the null hypthesis (H0) stating independence between X and Y.

Usage

```
pqad(q, n, R = 1000, resolution = NULL)

qqad(p, n, R = 1000, resolution = NULL)
```

Arguments

q	vector of quantiles.
n	number of observations (or minimum of unique values, if ties occur).
R	number of repetitions (default R = 1000)
resolution	resolution of checkerboard copula (default = NULL)
p	vector of probabilities.

Details

The distribution of qad in the setting of independence, i.e., the random variables X and Y are independent. The distribution is calculated in the following way: Samples of size n are drawn from independent random variables. Then qad is calculated. The procedure is repeated R times. #'

Value

pqad gives the distribution function, i.e. P(qad <= q). qqad gives the quantile function. The length of the result is determined by the length of q or p, respectively.

Examples

```
pqad(0.3, 45)
qqad(0.5, 30)
```

summary.qad *Summarize a qad object*

Description

Summary and coefficients of a qad output. The function summary() prints the dependence measures, sample size and resolution of the checkerboard copula and returns a list with the mentioned values. The function coef() returns a named vector with the selected values.

Usage

```
## S3 method for class 'qad'
summary(object, ...)

## S3 method for class 'qad'
coef(
  object,
  select = c("q(x1,x2)", "q(x2,x1)", "max.dependence", "asymmetry", "p.q(x1,x2)",
    "p.q(x2,x1)", "p.max.dependence", "p.asymmetry"),
  ...
)
```

Arguments

object	an object of class 'qad'
...	some methods for this generic require additional arguments. None are used in this method.
select	a vector of strings indicating which dependence measure should be returned. Options are c('q(x1,x2)', 'q(x2,x1)', 'max.dependence', 'asymmetry')

Value

an object containing the calculated values of a qad object.

Examples

```
n <- 100
x <- runif(n, 0, 1)
y <- x^2 + rnorm(n, 0, 0.1)
sample <- data.frame(x, y)
##(Not Run)
fit <- qad(sample)
summary(fit)
coef(fit)
coef(fit, select = c('q(x1,x2)','p.q(x1,x2)'))
```

Index