Package 'qad'

May 29, 2020

Type Package

Title Quantification of Asymmetric Dependence

Version 0.2.0

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

License GPL-2

Encoding UTF-8

LazyData true

Depends ggplot2, R (>= 2.10)

Imports data.table, foreach, doParallel, copula, parallel, viridis, ggExtra, dplyr

RoxygenNote 7.1.0

NeedsCompilation no

Author Florian Griessenberger [aut, cre], Robert R. Junker [aut], Wolfgang Trutschnig [aut]

Maintainer Florian Griessenberger <florian.griessenberger@sbg.ac.at>

Repository CRAN

Date/Publication 2020-05-29 12:00:03 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:	0.1.1
Date:	2018-12-18

Author(s)

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

Examples

```
##Create data set
# n <- 1000
# x <- rnorm(n,0,2)
# y <- x^2 + rnorm(n)
# sample <- data.frame(x,y)
# plot(sample)

##Function: empirical copula
# eval <- data.frame(x=c(0,0.1,0.2,0.45,1), y=c(0,0.1,0.5,0.23,1))
# emp_c_copula_eval(sample, eval, resolution = 10)
# mass <- emp_c_copula(sample, resolution=10)

##Function: qad()
# qad(sample, resolution = NULL, permutation = FALSE, nperm = 100, DoParallel = TRUE, ncores = NULL)
# help(qad)
# mod <- qad(sample)
# mod <- qad(sample, resolution = NULL, permutation = TRUE, nperm = 10, DoParallel = TRUE)

##Functions: summary(), coef()
# help(summary.qad)

# mod <- qad(sample)
```

```
# summary(mod)
# coef(mod)
# coef(mod, select = c('q(x2,x1)','mean.dependence'))

##Function: plot()
# help(plot.qad)

# plot(mod)
# plot(mod, addSample = TRUE, copula = FALSE, margins = TRUE, point.size = 0.7, panel.grid = FALSE)

##Function: cci()
# help(cci)

# n <- 1000
# cci(n, alternative = "one.sided")
# cci(n, alternative = "two.sided")

##Function: predict()
# help(predict.qad)
# values <- c(-2.4,1,0,2.6)
# predict.qad(mod, values = values, conditioned = 'x1')
# predict(mod, values, conditioned = "x1", nr_intervals = 10, pred_plot = TRUE, panel.grid = FALSE)

# values <- c(0.1,0.5)
# predict(mod, values, conditioned = "x2", nr_intervals = 5, copula = TRUE, pred_plot = TRUE)

##Function: pairwise.qad and heatmap.qad
# df <- iris[1:4]
# mod <- pairwise.qad(df)
# heatmap.qad(mod, select = 'dependence')
```

`cci` *Conditional confidence interval*

Description

Conditioned on the sample size n, approximated confidence intervals of the dependence measure qad(x,y) for independent random variables are computed. `cci()` can be used to test two random variables for independence.

Usage

```
cci(n, alternative = c("one.sided", "two.sided"))
```

Arguments

`n`	an integer indicating the sample size.
`alternative`	character string specifying the type of the confidence interval; must be one of "one.sided" (default) or "two.sided".

Details

`alternative = "one.sided"` provides a one-sided confidence interval, which can be interpreted that in 95 of 100 realizations of two independet random variables X and Y the calculated dependence measure qad(x,y) is less than the upper interval boundary. If `alternative = "two.sided"` 95 of 100 realizations lie in between the interval boundaries.

Value

a named vector indicating the lower and upper boundary of the confidence interval.

---

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.

Examples

```
## Generate data X from the product copula and compute the empirical copula
n <- 1000
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
X <- data.frame(x,y)
#(Not Run)
# mass_product <- emp_c_copula(X, smoothing = TRUE, resolution = 50)
```

```
# eval_points <- data.frame(x = c(0.3,0.6), y = c(0.5,0.9))
# eval_points$emp_cop <- emp_c_copula_eval(X, eval_points, smoothing = TRUE, resolution = 50)
# eval_points$cop <- eval_points$x * eval_points$y

## Compute empirical checkerboard copula of a sample with ties and plot density
n <- 1000
x <- sample(runif(n, -1, 1), n, replace=TRUE)
y <- x^2 + rnorm(n, 0, 1)
X <- data.frame(x,y)
#(Not Run)
# mass <- emp_c_copula(X, smoothing = TRUE, resolution = 10)
# plot_density(mass)
```

`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", "mean.dependence", "asymmetry"),
  fontsize = 4,
  significance = FALSE,
  sign.level = 0.05,
  scale = "abs",
  color = "plasma",
  rb_values = c(10, 0.315, 0.15)
)
```

Arguments

`pw_qad`	output of the function `pairwise.qad()`.
`select`	a character indicating which dependence value is plotted. Options are c("dependence", "mean.dependence", "asymmetry").
`fontsize`	a numeric specifying the font size of the values.
`significance`	a logical indicating whether significant values - with respect to the qad p.values - are denoted by a star.
`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").

rb_values — a vector of size 3 with number of values, start value and end value in the rainbow colors space.

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 <- 1000
x <- runif(n, 0, 1)
y <- x^2 + rnorm(n, 0, 1)
z <- runif(n, 0, 1)
sample_df <- data.frame(x, y, z)

#qad (Not Run)
model <- pairwise.qad(sample_df, permutation = FALSE)
heatmap.qad(model, select = "dependence", fontsize = 6)
```

---

mcData_independence *Dataset which is used internally in the package*

Description

Dataset which is used internally in the package

Usage

```
mcData_independence
```

Format

An object of class list of length 2.

Details

Results of the monte carlo simulation for qad according to the setting of independence.

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,
  resolution = NULL,
  remove.00 = FALSE,
  min.res = 3,
  permutation = FALSE,
  nperm = 1000,
  DoParallel = FALSE,
  registerC = registerDoParallel,
  ncores = NULL
)
```

Arguments

data_df	a data frame containing numeric columns with the observations of the sample.
resolution	an integer indicating the number of strips for the checkerboard aggregation (see emp_c_copula()). Default (NULL) uses the optimal resolution, computed out of the sample size.
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).
permutation	a logical indicating whether a p-value (based on permutations) is computed; (otherwise the p-value is computed by MC-simulation - see pqad()).
nperm	an integer indicating the number of permutation runs.
DoParallel	a logical value indicating whether the permutation test is computed parallelized.
registerC	function to register the parallel backend. It is recommended to use registerDoParallel() of the doParallel package (default). Other option is for example on a linux based system to install the doMC package and use registerDoMC
ncores	an integer indicating the number of cores used for parallelization. Default (NULL) uses the maximum number of cores minus 1.

Value

a list, containing 8 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
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
z <- runif(n, 0, 1)
sample_df <- data.frame(x,y,z)

#qad
model <- pairwise.qad(sample_df, permutation = FALSE)
heatmap.qad(model, select = "dependence", fontsize = 5, significance = TRUE, sign.level = 0.05)
```

`plot.qad` *Plot conditional probabilites*

Description

Plots conditional probabilities for each strip of the checkerboard copula in the copula setting or in the retransformed data setting.

Usage

```
## S3 method for class 'qad'
plot(
  x,
  addSample = FALSE,
  copula = FALSE,
  density = FALSE,
  margins = FALSE,
  point.size = 0.8,
  panel.grid = TRUE,
  color = "plasma",
  rb_values = c(10, 0.315, 0.15),
  ...
)
```

Arguments

`x`	an object of class qad.
`addSample`	a logical determining whether the observations (or pseudo-observations) are added in the plot (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.
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 <- 1000
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
sample <- data.frame(x, y)

#qad (Not Run)
# mod <- qad(sample)
# plot(mod, addSample = TRUE, copula = FALSE)

## Example 2
n <- 1000
x <- runif(n, -1, 1)
y <- x^2 + rnorm(n, 0, 0.1)
sample <- data.frame(x, y)

#qad (Not Run)
# mod <- qad(sample)
# plot(mod, addSample = TRUE, copula = TRUE)
# plot(mod, 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)
sample <- data.frame(x,y)
plot(sample)

mass <- emp_c_copula(sample, resolution=8)
plot_density(mass, density=TRUE)
plot_density(mass, density=FALSE)
```

predict.qad *Predict conditional probabilities*

Description

The function predict.qad() predicts the probabilities to end up in specific intervals given x or y values and plots the conditional probabilities. The prediction can be computed in the copula setting or in the data setting.

Usage

```
## S3 method for class 'qad'
predict(
  object,
  values,
  conditioned = c("x1", "x2"),
  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 the number of intervals for the prediction. Note, that in the copula setting the intervals are equidistant, in the data setting the retransformed intervals have different lengths. (default = NULL: the number of intervals is the resolution of the checkerboard copula)
prediction_interval	a vector specifying the interval boundaries for which the conditional probability is computed. Options are NULL (default) to predict the conditional probabilites 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 probabilites 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 named data.frame and a plot (optional). Each row stands for an evaluation point and the columns contain the conditional probabilities of the intervals.

Note

Predictions are only possible for values within the range of the sample (or between 0 and 1 in the copula setting). Values exceeding the range are removed.

Examples

```
n <- 1000
x <- runif(n, -1 ,1)
y <- x^2 + rnorm(n, 0, 1)
sample <- data.frame(x, y)

##(Not Run)
#mod <- qad(sample)
#val <- c(-0.5, 0,1)
#predict(mod, values = val, conditioned = "x1", copula = FALSE, pred_plot = TRUE)
#predict(mod, values = val, conditioned = "x1", copula = TRUE)
#predict(mod, values = val, conditioned = "x1", copula = TRUE, pred_plot = TRUE)
```

---

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,
  permutation = FALSE,
  nperm = 1000,
  DoParallel = TRUE,
  registerC = registerDoParallel,
```

```
  ncores = NULL,
  print = TRUE,
  remove.00 = FALSE,
  ...
)

## S3 method for class 'numeric'
qad(
  x,
  y,
  resolution = NULL,
  permutation = FALSE,
  nperm = 1000,
  DoParallel = TRUE,
  registerC = registerDoParallel,
  ncores = NULL,
  print = TRUE,
  remove.00 = FALSE,
  ...
)
```

Arguments

x	a data.frame containing two columns with the observations of the bivariate 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 emp_c_copula). Default = NULL uses the optimal resolution.
permutation	a logical indicating whether a p-value (based on permutations) is computed; otherwise a p-value is computed on MC-simulations (see pqad()).
nperm	an integer indicating the number of permutation runs.
DoParallel	a logical value indicating whether the repetitions in the permutation test is computed parallel.
registerC	function to register the parallel environment. It is recommended to use registerDoParallel(), contained in the doParallel package (default). Another option, especially for a linux based system, is to install the doMC package and use registerDoMC
ncores	an integer indicating the number of cores used for parallel computation. (Default = NULL, which is defined by max(cores)-1)
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 (ECB), and calculate zeta_1 of the ECB copula and its transpose. In order to test for independence (in both directions), the distribution (and hence the p-value) of a Monte-Carlo simulation is provided (default). Alternatively, a permutation test can be used to obtain p-values for the direction and asymmetry.

Value

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

`data`	a data.frame containing the input data.
`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.

Note

The computation of the p-values (aggregated by permutations) take some time.

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.

Examples

```
#Example 1 (independence)

n <- 1000
x <- runif(n,0,1)
y <- runif(n,0,1)
sample <- data.frame(x,y)
qad(sample)

###

#Example 2 (mutual complete dependence)

n <- 1000
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)
```

---

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)

qqad(p, n)
```

Arguments

q	vector of quantiles.
n	number of observations.
p	vector of probabilities.

Details

The distribution of qad was computed in the setting of independence between the random variables X and Y in the following way:

For n < 1000, Monte Carlo (MC) simulation of H0 with 20.000 repetitions were executed for each sample size. According to these values the empirical cumulative distribution functions and the quantile functions were computed and then approximated on a coarser grid.

For n >= 1000, MC simulations were executed again, but this time on a coarser sample size grid (steps of 100) until the size of 10.000. The so obtained quantiles were approximated using the parametric function a*n^b+c, whereby the parameters a,b,c were estimated using the R-function nls. Using the so calculated quantiles, the empirical distribution function and the quantile functions were approximated.

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)", "mean.dependence", "asymmetry", "p.q(x1,x2)",
    "p.q(x2,x1)", "p.mean.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)', 'mean.dependence', 'asymmetry')

Value

an object containing the calculated values of a qad object.

Examples

```
n <- 1000
x <- runif(n, 0, 1)
y <- runif(n, 0, 1)
sample <- data.frame(x, y)
##(Not Run)
# mod <- qad(sample, permutation = TRUE, nperm = 100, print = FALSE)
# summary(mod)
# coef(mod)
# coef(mod, select = c('q(x1,x2)','p.q(x1,x2)'))
```

Index