Package ‘RDM’

July 21, 2025

Type Package

Title Quantify Dependence using Rearranged Dependence Measures

Version 0.1.1

Description Estimates the rearranged dependence measure ('RDM') of two continuous random variables for different underlying measures.
Furthermore, it provides a method to estimate the (SI)-rearrangement copula using empirical checkerboard copulas.
It is based on the theoretical results presented in Strothmann et al. (2022) <doi:10.48550/arXiv.2201.03329> and Strothmann (2021) <doi:10.17877/DE290R-22733>.

URL `https://github.com/ChristopherStrothmann/RDM`

BugReports `https://github.com/ChristopherStrothmann/RDM/issues`

License GPL-2

Language en-GB

Encoding UTF-8

Depends R (>= 3.5.0)

Imports Rfast (>= 2.0.0), Rcpp (>= 1.0.8.3)

LinkingTo Rcpp

RoxygenNote 7.2.3

Suggests testthat (>= 3.0.0), copula (>= 1.0.0), qad (>= 1.0.0)

Config/testthat/edition 3

NeedsCompilation yes

Author Holger Dette [aut] (ORCID: <`https://orcid.org/0000-0001-7048-474X`>),
Karl Friedrich Siburg [aut],
Christopher Strothmann [aut, cre] (ORCID: <`https://orcid.org/0000-0002-3537-5019`>),
qad contributors [cph] (Authors of the modified code.cpp of the R-package 'qad' listed in inst/qad-authors.txt (GPL-2))

Maintainer Christopher Strothmann `<rdmpackage@gmail.com>`

Repository CRAN

Date/Publication 2023-02-24 19:50:05 UTC

Contents

`checkerboardDensity` *Estimate the checkerboard mass density*

Description

Estimate a non-square checkerboard mass density

Usage

```
checkerboardDensity(X, Y, resolution1, resolution2)
```

Arguments

`X`	First coordinate of the observations.
`Y`	Second coordinate of the observations.
`resolution1`	A natural number specifying the resolution of the first component.
`resolution2`	A natural number specifying the resolution of the second component.

Details

This implementation modifies the code of build_checkerboard_weights() published in 'qad', version 1.0.4, available at https://CRAN.R-project.org/package=qad, to allow for non-square checkerboard mass densities. For more details on the implementation see `ECBC` and for more information on the implemented changes, see the file 'src/code.cpp'.

Value

The estimated checkerboard mass density.

Examples

```
checkerboardDensity(runif(20), runif(20), 3, 3)
```

`checkerboardDensityIndex`

Estimate a single entry of the checkerboard mass density

Description

Estimate the value A_{kl} of the non-square checkerboard mass density.

Usage

```
checkerboardDensityIndex(X, Y, k, l, resolution1, resolution2)
```

Arguments

`X`	First coordinate of the observations.
`Y`	Second coordinate of the observations.
`k`	Index of the first component.
`l`	Index of the second component.
`resolution1`	A natural number specifying the resolution of the first component.
`resolution2`	A natural number specifying the resolution of the second component.

Details

This implementation modifies the code of build_checkerboard_weights() published in 'qad', version 1.0.4, available at https://CRAN.R-project.org/package=qad, to allow for the evaluation of a single index of the non-square checkerboard mass densities. For more details on the implementation see `ECBC` and for more information on the implemented changes, see the file 'src/code.cpp'.

Value

The estimated checkerboard mass density A_{kl}.

Examples

```
U <- runif(20)
V <- runif(20)
checkerboardDensity(U, V, 3, 3)
checkerboardDensityIndex(U, V, 1, 2, 3, 3)
```

`computeBandwidth` *Compute bandwidth via cross-validation*

Description

An implementation of the cross-validation principle for the bandwidth selection as presented in Strothmann, Dette and Siburg (2022) <arXiv:2201.03329>.

Usage

```
computeBandwidth(X, sL, sU, method = c("cvsym", "cvasym"), reduce = TRUE)
```

Arguments

`X`	A bivariate data.frame containing the observations. Each row contains one observation.
`sL`	Lower bound N^{sL} for the possible bandwidth parameters (where N is the number of observations).
`sU`	Upper bound N^{sU} for the possible bandwidth parameters (where N is the number of observations).
`method`	"cvsym" uses either a symmetric cross-validation principle (N_1 = N_2) and "cvasym" uses an asymmetric cross-validation principle (i.e. N_1 and N_2 may attain different values).
`reduce`	In case reduce is set to TRUE, the parameter is chosen from N, N+2, ... instead of N, N+1, N+2, ...

Details

This function computes the optimal bandwidth given the bivariate observations X of length N. Currently, there are two different algorithms implemented:

- "cvsym" - Computes the optimal bandwidth choice for a square checkerboard mass density according to the cross-validation principle. The bandwidth is a natural number between $N^{sL}, ..., N^{sU}$
- "cvasym" - Computes the optimal bandwidth choice (N_1, N_2) for a non-square checkerboard mass density according to the cross-validation principle. The bandwidths N_1, N_2 are natural numbers between $N^{sL}, ..., N^{sU}$ and may possibly attain different values.

Value

The chosen bandwidth depending on the data.frame X.

Examples

```
n <- 20
X <- cbind(runif(n), runif(n))
computeBandwidth(X, sL = 0.25, sU = 0.5, method="cvsym", reduce=TRUE)
```

---

computeCBMeasure *Dependence measures for the checkerboard copula*

Description

Computes $\mu(C^{\#}(A))$ for some underlying measure for the checkerboard copula $C^{\#}(A)$. This measure depends only on the input matrix A.

Usage

```
computeCBMeasure(A, method = c("spearman", "kendall", "bkr", "dss", "zeta1"))
```

Arguments

`A`	A (possibly non-square) checkerboard mass density.
`method`	Determines the underlying dependence measure. Options include "spearman", "kendall", "bkr", "dss", "chatterjee" and "zeta1".

Details

This function computes $\mu(C^{\#}(A))$ for one of several underlying measures for a given checkerboard copula $C^{\#}(A)$. Most importantly, the value only depends on the (possibly non-square) matrix A and implicitly assumes the form of $C^{\#}(A)$ given in Strothmann, Dette and Siburg (2022) <arXiv:2201.03329>. Currently, the following underlying measures are implemented:

- "spearman" Implements the concordance measure Spearman's ρ,
- "kendall" Implements the concordance measure Kendall's τ,
- "bkr" Implements the Blum–Kiefer–Rosenblatt R, also known as the L^2-Schweizer-Wolff-measure <doi:10.1214/aos/1176345528>,
- "dss" Implements the Dette-Siburg-Stoimenov measure of complete dependence <doi:10.1111/j.1467-9469.2011.00767.x>, also known as Chatterjee's ξ <doi:10.1080/01621459.2020.1758115>,
- "zeta1" Implements the ζ_1-measure of complete dependence established by W. Trutschnig <doi:10.1016/j.jmaa.2011.06.013>.

Value

The value of $\mu(C^{\#}(A))$. For a sorted A, this corresponds to the rearranged dependence measure $R_\mu(C^{\#}(A))$.

Examples

```
n <- 10
A <- diag(n)/n
computeCBMeasure(A, method="spearman")
```

`rdm` *Rearranged dependence measure*

Description

This function estimates the asymmetric dependence between X and Y using the rearranged dependence measure $R_\mu(X, Y)$ for different possible underlying measures μ. A value of 0 characterizes independence of X and Y, while a value of 1 characterizes a functional relationship between X and Y, i.e. $Y = f(X)$.

Usage

```
rdm(
  X,
  method = c("spearman", "kendall", "dss", "zeta1", "bkr", "all"),
  bandwidth_method = c("fixed", "cv", "cvsym"),
  bandwidth_parameter = 0.5,
  permutation = FALSE,
  npermutation = 1000,
  checkInput = FALSE
)
```

Arguments

`X`	A bivariate data.frame containing the observations. Each row contains one bivariate observation.
`method`	Options include "spearman", "kendall", "bkr", "dss", "chatterjee" and "zeta1".The option "all" returns the value for all aforementioned methods.
`bandwidth_method`	A character string indicating the use of either a cross-validation principle (square or non-square) or a fixed bandwidth (oftentimes called resolution).
`bandwidth_parameter`	A numerical vector which contains the necessary optional parameters for the exponent of the chosen bandwidth method. In case of N observations, the bandwidth_parameter (s_1, s_2) determines a lower bound N^{s_1} and upper bound N^{s_2} for the cross-validation methods or a single number s for the fixed bandwidth method resulting in N^s. The parameters have to lie in $(0, 1/2)$ and fulfil $s_1 < s_2$.
`permutation`	Whether or not to perform a permutation test
`npermutation`	Number of repetitions of the permutation test
`checkInput`	Whether or not to perform validity checks of the input

Details

This function estimates $R_\mu(X, Y)$ using the empirical checkerboard mass density A. To arrive at $R_\mu(X, Y)$, A is appropriately sorted and then evaluated for the underlying measure. The estimated R_μ always takes values between 0 and 1 with

- $R_\mu(X, Y) = 0$ if and only if X and Y are independent.
- $R_\mu(X, Y) = 1$ if and only if $Y = f(X)$ for some measurable function f.

Currently, the following underlying measures are implemented:

- "spearman" Implements the concordance measure Spearman's ρ (which is identical to the L_1-Schweizer-Wolff-measure),
- "kendall" Implements the concordance measure Kendall's τ,
- "bkr" Implements the Blum–Kiefer–Rosenblatt R, also known as the L^2-Schweizer-Wolff-measure <doi:10.1214/aos/1176345528>,
- "dss" Implements the Dette-Siburg-Stoimenov measure of complete dependence <doi:10.1111/j.1467-9469.2011.00767.x>, also known as Chatterjee's ξ <doi:10.1080/01621459.2020.1758115>,
- "zeta1" Implements the ζ_1-measure of complete dependence established by W. Trutschnig <doi:10.1016/j.jmaa.2011.06.013>.

The estimation of the checkerboard mass density A depends on the choice of the bandwidth for the checkerboard copula. For a detailed discussion of "cv" and "cvsym", see `computeBandwidth`.

Value

The estimated value of the rearranged dependence measure

Examples

```
n <- 50
X <- cbind(runif(n), runif(n))
rdm(X, method="spearman", bandwidth_method="fixed", bandwidth_parameter=.3)
n <- 20
U <- runif(n)
rdm(cbind(U, U), method="spearman", bandwidth_method="cv", bandwidth_parameter=c(0.25, 0.5))
```

`sortDSMatrix` *Sort a (possibly non-square) doubly stochastic matrix*

Description

Sorts an arbitrary doubly stochastic $N_1 \times N_2$ matrix A into the matrix $A^\uparrow$ such that the induced checkerboard copula $C(A^\uparrow)$ is stochastically increasing.

Usage

```
sortDSMatrix(A)
```

Arguments

`A`	A (possibly non-square) doubly stochastic matrix or (possibly non-square) checkerboard mass density.

Details

The algorithm to sort a doubly stochastic matrix A is given in Strothmann, Dette and Siburg (2022) <arXiv:2201.03329>. Since this implementation does not depend on the appropriate scaling of the matrix A, both doubly stochastic matrices and checkerboard mass densities are admissible inputs.

Value

The sorted version $A^{\uparrow}$ of the matrix A.

Examples

```
n <- 4
A <- diag(n)[n:1, ]
print(A)
sortDSMatrix(A)
```

Index