Package 'MMDCopula'

August 10, 2021

Type Package

Title Robust Estimation of Copulas by Maximum Mean Discrepancy

Version 0.2.0

Description Provides functions for the robust estimation of parametric families of copulas using minimization of the Maximum Mean Discrepancy, following the article Alquier, Chérief-Abdellatif, Derumigny and Fermanian (2020) <arXiv:2010.00408>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.1.1

Depends R (>= 3.6.0)

Imports VineCopula, cubature, pcaPP, randtoolbox, pbapply

Suggests knitr, rmarkdown

VignetteBuilder knitr

BugReports https://github.com/AlexisDerumigny/MMDCopula/issues

NeedsCompilation no

Author Alexis Derumigny [aut, cre] (<https://orcid.org/0000-0002-6163-8097>), Pierre Alquier [aut] (<https://orcid.org/0000-0003-4249-7337>), Jean-David Fermanian [aut] (<https://orcid.org/0000-0001-5960-5555>), Badr-Eddine Chérief-Abdellatif [aut]

Maintainer Alexis Derumigny <a.f.f.derumigny@tudelft.nl>

Repository CRAN

Date/Publication 2021-08-10 08:30:08 UTC

R topics documented:

`BiCopConfIntMMD` *Confidence intervals for the estimated parameter of a bivariate parametric copula using MMD estimation*

Description

Confidence intervals for the estimated parameter of a bivariate parametric copula using MMD estimation

Usage

```
BiCopConfIntMMD(
  x1,
  x2,
  family,
  nResampling = 100,
  subsamplingSize = length(x1),
  corrSubSampling = TRUE,
  level = 0.95,
  ...
)
```

Arguments

`x1`	vector of observations of the first coordinate.
`x2`	vector of observations of the second coordinate.
`family`	parametric family of copulas. Supported families are: • `1`: Gaussian copulas • `3`: Clayton copulas • `4`: Gumbel copulas • `5`: Frank copulas • `MO`: Marshall-Olkin copulas
`nResampling`	number of resampling times.
`subsamplingSize`	size of the subsample. By default it is `length(u1)`, i.e. this corresponds to the nonparametric boostrap.
`corrSubSampling`	this parameter is only used for subsampling-based confidence intervals. If `TRUE`, the confidence interval uses the corrected subsample empirical process.
`level`	the nominal confidence level.
`...`	other parameters to be given to `BiCopEstMMD` or `BiCopEst.MO`.

Value

a list with the confidence intervals CI.Tau for Kendall's tau and CI.Par for the corresponding parameter.

References

Alquier, P., Chérief-Abdellatif, B.-E., Derumigny, A., and Fermanian, J.D. (2020). Estimation of copulas via Maximum Mean Discrepancy. ArXiv preprint arxiv:2010.00408.

Kojadinovic I., and Stemikovskaya, K. (2019) Subsampling (weighted smooth) empirical copula processes. Journal of Multivariate Analysis, 173, 704-723, doi: 10.1016/j.jmva.2019.05.007.

Examples

```
data = VineCopula::BiCopSim(N = 50, family = 1, par = 0.3)
result = BiCopConfIntMMD(x1 = data[,1], x2 = data[,2], family = 1,
  nResampling = 2, subsamplingSize = 10, niter = 10)

data_ = VineCopula::BiCopSim(N = 1000, family = 1, par = 0.3)
result_ = BiCopConfIntMMD(x1 = data_[,1], x2 = data_[,2], family = 1)
result_$CI.Tau
result_$CI.Par
```

`BiCopEst.MO` *Estimation of Marshall-Olkin copulas*

Description

Estimation of Marshall-Olkin copulas

Usage

```
BiCopEst.MO(
  u1,
  u2,
  method,
  par.start = 0.5,
  kernel = "gaussian.Phi",
  gamma = 0.95,
  alpha = 1,
  niter = 100,
  C_eta = 1,
  ndrawings = 10,
  naveraging = 1
)
```

Arguments

u1	vector of observations of the first coordinate, in $[0, 1]$.
u2	vector of observations of the second coordinate, in $[0, 1]$.
method	a character giving the name of the estimation method, among: • curve: α is estimated by inversion of the probability measure of the diagonal $\{(u, v) : u = v\}$ • itau: α is estimated by inversion of Kendall's tau • MMD: α is estimated by MMD optimization
par.start	starting parameter of the gradient descent. (only used for method = "MMD")
kernel	the kernel used in the MMD distance (only used for method = "MMD") : it can be a function taking in parameter (u1,u2,v1,v2,gamma,alpha) or a name giving the kernel to use in the list: • gaussian: Gaussian kernel $k(x, y) = \exp(-\|\frac{x-y}{\gamma}\|_2^2)$ • exp.l2: $k(x, y) = \exp(-\|\frac{x-y}{\gamma}\|_2)$ • exp.l1: $k(x, y) = \exp(-\|\frac{x-y}{\gamma}\|_1)$ • inv.l2: $k(x, y) = 1/(1 + \|\frac{x-y}{\gamma}\|_2)^\alpha$ • inv.l1: $k(x, y) = 1/(1 + \|\frac{x-y}{\gamma}\|_1)^\alpha$ Each of these names can receive the suffix ".Phi", such as "gaussian.Phi" to indicates that the kernel $k(x, y)$ is replaced by $k(\Phi^{-1}(x), \Phi^{-1}(y))$ where Φ^{-1} denotes the quantile function of the standard Normal distribution.
gamma	parameter γ to be used in the kernel. (only used for method = "MMD")
alpha	parameter α to be used in the kernel, if any. (only used for method = "MMD")
niter	the stochastic gradient algorithm is composed of two phases: a first "burn-in" phase and a second "averaging" phase. If niter is of size 1, the same number of iterations is used for both phases of the stochastic gradient algorithm. If niter is of size 2, then niter[1] iterations are done for the burn-in phase and niter[2] for the averaging phase. (only used for method = "MMD")
C_eta	a multiplicative constant controlling for the size of the gradient descent step. The step size is then computed as C_eta / sqrt(i_iter) where i_iter is the index of the current iteration of the stochastic gradient algorithm. (only used for method = "MMD")
ndrawings	number of replicas of the stochastic estimate of the gradient drawn at each step. The gradient is computed using the average of these replicas. (only used for method = "MMD")
naveraging	number of full run of the stochastic gradient algorithm that are averaged at the end to give the final estimated parameter. (only used for method = "MMD")

Value

the estimated parameter (alpha) of the Marshall-Olkin copula.

References

Alquier, P., Chérief-Abdellatif, B.-E., Derumigny, A., and Fermanian, J.D. (2020). Estimation of copulas via Maximum Mean Discrepancy. ArXiv preprint arxiv:2010.00408

See Also

`BiCopSim.MO` for the estimation of Marshall-Olkin copulas. `BiCopEstMMD` for the estimation of other parametric copula families by MMD.

Examples

```
U <- BiCopSim.MO(n = 1000, alpha = 0.2)
estimatedPar <- BiCopEst.MO(u1 = U[,1], u2 = U[,2], method = "MMD", niter = 1, ndrawings = 1)

estimatedPar <- BiCopEst.MO(u1 = U[,1], u2 = U[,2], method = "MMD")
```

`BiCopEstMMD` *Estimation of parametric bivariate copulas using stochastic gradient descent on the MMD criteria*

Description

This function uses computes the MMD-estimator of a bivariate copula family. This computation is done through a stochastic gradient algorithm, that is itself computed by the function `BiCopGradMMD()`. The main arguments are the two vectors of observations, and the copula family. The bidimensional copula families are indexed in the same way as in `VineCopula::BiCop()` (which computes the MLE estimator).

Usage

```
BiCopEstMMD(
  u1,
  u2,
  family,
  tau = NULL,
  par = NULL,
  par2 = NULL,
  kernel = "gaussian",
  gamma = 0.23,
  alpha = 1,
  niter = 100,
  C_eta = 1,
  epsilon = 1e-04,
  method = "QMCV",
  quasiRNG = "sobol",
  ndrawings = 10
)
```

Arguments

u1	vector of observations of the first coordinate, in $[0,1]$.
u2	vector of observations of the second coordinate, in $[0,1]$.
family	the chosen family of copulas (see the documentation of the class VineCopula::BiCop() for the available families).
tau	the copula family can be parametrized by the parameter par or by Kendall's tau. Here, the user can choose the initial value of tau for the stochastic gradient algorithm. If NULL, a random value is chosen instead.
par	if different from NULL, the parameter tau is ignored, and the initial parameter must be given here. The initial Kendall's tau is then computed thanks to VineCopula::BiCopPar2Tau().
par2	initial value for the second parameter, if any. (Works only for Student copula).
kernel	the kernel used in the MMD distance: it can be a function taking in parameter (u1,u2,v1,v2,gamma,alpha) or a name giving the kernel to use in the list: • gaussian: Gaussian kernel $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_2^2)$ • exp.l2: $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_2)$ • exp.l1: $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_1)$ • inv.l2: $k(x,y) = 1/(1+\|\frac{x-y}{\gamma}\|_2)^\alpha$ • inv.l1: $k(x,y) = 1/(1+\|\frac{x-y}{\gamma}\|_1)^\alpha$ Each of these names can receive the suffix ".Phi", such as "gaussian.Phi" to indicates that the kernel $k(x,y)$ is replaced by $k(\Phi^{-1}(x), \Phi^{-1}(y))$ where Φ^{-1} denotes the quantile function of the standard Normal distribution.
gamma	parameter γ to be used in the kernel.
alpha	parameter α to be used in the kernel, if any.
niter	the stochastic gradient algorithm is composed of two phases: a first "burn-in" phase and a second "averaging" phase. If niter is of size 1, the same number of iterations is used for both phases of the stochastic gradient algorithm. If niter is of size 2, then niter[1] iterations are done for the burn-in phase and niter[2] for the averaging phase.
C_eta	a multiplicative constant controlling for the size of the gradient descent step. The step size is then computed as C_eta / sqrt(i_iter) where i_iter is the index of the current iteration of the stochastic gradient algorithm.
epsilon	the differential of VineCopula::BiCopTau2Par() is computed thanks to a finite difference with increment epsilon.
method	the method of computing the stochastic gradient: • MC: classical Monte-Carlo with ndrawings replications. • QMCV: usual Monte-Carlo on U with ndrawings replications, quasi Monte-Carlo on V.
quasiRNG	a function giving the quasi-random points in $[0,1]^2$ or a name giving the method to use in the list: • sobol: use of the Sobol sequence implemented in randtoolbox::sobol

- halton: use of the Halton sequence implemented in randtoolbox::halton
- torus: use of the Torus sequence implemented in randtoolbox::torus

ndrawings number of replicas of the stochastic estimate of the gradient drawn at each step. The gradient is computed using the average of these replicas.

Value

an object of class VineCopula::BiCop() containing the estimated copula.

References

Alquier, P., Chérief-Abdellatif, B.-E., Derumigny, A., and Fermanian, J.D. (2020). Estimation of copulas via Maximum Mean Discrepancy. ArXiv preprint arxiv:2010.00408

See Also

VineCopula::BiCopEst() for other methods of estimation such as Maximum Likelihood Estimation or Inversion of Kendall's tau. BiCopGradMMD() for the computation of the stochastic gradient. BiCopEst.MO for the estimation of Marshall-Olkin copulas by MMD.

Examples

```
# Estimation of a bivariate Gaussian copula with correlation 0.5.
dataSampled = VineCopula::BiCopSim(N = 500, family = 1, par = 0.5)
estimator = BiCopEstMMD(u1 = dataSampled[,1], u2 = dataSampled[,2], family = 1, niter=10)
estimator$par


# Estimation of a bivariate Student copula with correlation 0.5 and 5 degrees of freedom
dataSampled = VineCopula::BiCopSim(N = 1000, family = 2, par = 0.5, par2 = 5)
estimator = BiCopEstMMD(u1 = dataSampled[,1], u2 = dataSampled[,2], family = 2)
estimator$par
estimator$par2


# Comparison with maximum likelihood estimation with and without outliers
dataSampled = VineCopula::BiCopSim(N = 500, family = 1, par = 0.5)
estimatorMMD = BiCopEstMMD(u1 = dataSampled[,1], u2 = dataSampled[,2], family = 1)
estimatorMMD$par
estimatorMLE = VineCopula::BiCopEst(u1 = dataSampled[,1], u2 = dataSampled[,2],
  family = 1, method = "mle")
estimatorMLE$par

dataSampled[1:10,1] = 0.999
dataSampled[1:10,2] = 0.001
estimatorMMD = BiCopEstMMD(u1 = dataSampled[,1], u2 = dataSampled[,2], family = 1)
estimatorMMD$par
estimatorMLE = VineCopula::BiCopEst(u1 = dataSampled[,1], u2 = dataSampled[,2],
  family = 1, method = "mle")
estimatorMLE$par
```

```
# Estimation of a bivariate Gaussian copula with real data
data("daxreturns", package = "VineCopula")
BiCopEstMMD(u1 = daxreturns[,1], u2 = daxreturns[,2], family = 1)
estimator$par
```

`BiCopGradMMD` *Computation of the gradient of the MMD criterion for parametric bivariate copulas models*

Description

This function computes a stochastic estimate of the gradient of the MMD criterion for parametric estimation of bidimensional copula family. The main arguments are the two vectors of observations, and the copula family. The family is parametrized as in `VineCopula::BiCop()`, using the Kendall's tau instead of the first parameter. This function is used by `BiCopEstMMD()` to perform parameter estimation via MMD minimization.

Usage

```
BiCopGradMMD(
  u1,
  u2,
  family,
  tau,
  par = NULL,
  par2 = 0,
  kernel = "gaussian.Phi",
  gamma = 0.95,
  alpha = 1,
  epsilon = 1e-04,
  method = "QMCV",
  quasiRNG = "sobol",
  ndrawings = 10
)
```

Arguments

`u1`	vector of observations of the first coordinate, in $[0, 1]$.
`u2`	vector of observations of the second coordinate, in $[0, 1]$.
`family`	the chosen family of copulas (see the documentation of the class `VineCopula::BiCop()` for the available families).
`tau`	the copula family can be parametrized by the parameter `par` or by Kendall's tau. This function assumes a Kendall tau parametrization. Thus, the user can choose the value of Kendall tau at which the stochastic gradient should be computed.

`par` if different from `NULL`, the user must instead of `tau` specify the corresponding parameter `par`. The value of `tau` is then ignored.

`par2` value for the second parameter, if any. (Works only for Student copula).

`kernel` the kernel used in the MMD distance: it can be a function taking in parameter `(u1,u2,v1,v2,gamma,alpha)` or a name giving the kernel to use in the list:

- `gaussian`: Gaussian kernel $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_2^2)$
- `exp.l2`: $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_2)$
- `exp.l1`: $k(x,y) = \exp(-\|\frac{x-y}{\gamma}\|_1)$
- `inv.l2`: $k(x,y) = 1/(1+\|\frac{x-y}{\gamma}\|_2)^\alpha$
- `inv.l1`: $k(x,y) = 1/(1+\|\frac{x-y}{\gamma}\|_1)^\alpha$

Each of these names can receive the suffix ".Phi", such as "gaussian.Phi" to indicates that the kernel $k(x,y)$ is replaced by $k(\Phi^{-1}(x), \Phi^{-1}(y))$ where Φ^{-1} denotes the quantile function of the standard Normal distribution.

`gamma` parameter γ to be used in the kernel.

`alpha` parameter α to be used in the kernel, if any.

`epsilon` the differential of `VineCopula::BiCopTau2Par()` is computed thanks to a finite difference with increment `epsilon`.

`method` the method of computing the stochastic gradient:

- `MC`: classical Monte-Carlo with `ndrawings` replications.
- `QMCV`: usual Monte-Carlo on U with `ndrawings` replications, quasi Monte-Carlo on V.

`quasiRNG` a function giving the quasi-random points in $[0,1]^2$ or a name giving the method to use in the list:

- `sobol`: use of the Sobol sequence implemented in `randtoolbox::sobol`
- `halton`: use of the Halton sequence implemented in `randtoolbox::halton`
- `torus`: use of the Torus sequence implemented in `randtoolbox::torus`

`ndrawings` number of replicas of the stochastic estimate of the gradient drawn at each step. The gradient is computed using the average of these replicas.

Value

the value of the gradient.

References

Alquier, P., Chérief-Abdellatif, B.-E., Derumigny, A., and Fermanian, J.D. (2020). Estimation of copulas via Maximum Mean Discrepancy. ArXiv preprint arxiv:2010.00408

See Also

`BiCopEstMMD()` for the estimation of parametric bivariate copulas by stochastic gradient descent on the MMD criteria.

Examples

```
# Simulation from a bivariate Gaussian copula with correlation 0.5.
dataSampled = VineCopula::BiCopSim(N = 500, family = 1, par = 0.5)

# computation of the gradient of the MMD criteria at different points
# Gradient is small at the true parameter
BiCopGradMMD(dataSampled[,1], dataSampled[,2], family = 1, par = 0.5)
# Gradient is negative when below the parameter
BiCopGradMMD(dataSampled[,1], dataSampled[,2], family = 1, par = 0.1)
# and positive when above
BiCopGradMMD(dataSampled[,1], dataSampled[,2], family = 1, par = 0.8)
```

BiCopParamDistLp *Compute the distance between 2 parametric copulas*

Description

This function uses the numerical integration procedure `cubature::hcubature()` to numerical integrate the distance between the distribution or between the densities of two bivariate copulas.

Usage

```
BiCopParamDistLp(
  family,
  par,
  par_p,
  par2 = par,
  par2_p = par_p,
  family_p = family,
  p,
  type,
  maxEval = 0
)
```

Arguments

`family`	family of the first copula.
`par`	first parameter of the first copula.
`par_p`	first parameter of the second copula.
`par2`	second parameter of the first copula (only useful for two-parameter families of copulas).
`par2_p`	second parameter of the first copula (only useful for two-parameter families of copulas).
`family_p`	family of the second copula.
`p`	determines the L_p distance that is used.

type	type of the functions considered. Can be cdf for the distance between the two cumulative distribution functions or pdf for the distance between the two probability density functions.
maxEval	maximum number of evaluation of the function be integrated. If 0, then no maximum limit is given.

Value

a list of four items

- distance the value of the distance
- integral the value of the integral, which is the p-th power of the distance.
- error the estimated relative error of the integral
- returnCode the integer return code of the C routine called by cubature::hcubature(). This should be 0 if there is no error.

Examples

```
# Distance between the densities of a Gaussian copula with correlation 0.5
# and a Gaussian copula with correlation 0.2
BiCopParamDistLp(family = 1, par = 0.5, par_p = 0.2, p = 2, type = "cdf", maxEval = 10)

# Distance between the cdf of a Student copula
# with correlation 0.5 and 4 degrees of freedom
# and a Student copula with the same correlation but 20 degrees of freedom
BiCopParamDistLp(family = 2, par = 0.5, par_p = 0.5,
par2 = 5, par2_p = 20, p = 2, type = "pdf", maxEval = 10)

# Distance between the densities of a Gaussian copula with correlation 0.5
# and of a Student copula with correlation 0.5 and 15 degrees of freedom
BiCopParamDistLp(family = 1, par = 0.5, par_p = 0.5, par2_p = 15,
family_p = 2, p = 2, type = "pdf", maxEval = 10)
```

BiCopSim.MO *Simulation of Marshall-Olkin copula*

Description

This functions simulates independent realizations from the Marshall-Olkin copula.

Usage

```
BiCopSim.MO(n, alpha)
```

Arguments

n	number of samples
alpha	parameter of the Marshall-Olkin copula

Value

an $n \times 2$ matrix containing the samples

See Also

BiCopEst.MO for the estimation of Marshall-Olkin copulas.

Examples

```
# Simulation from a Marshall-Olkin copula with parameter alpha = 0.5
BiCopSim.MO(n = 100, alpha = 0.5)
```

Index