Package ‘pacotest’

December 15, 2020

Type Package

Title Testing for Partial Copulas and the Simplifying Assumption in Vine Copulas

Version 0.4.0

Date 2020-12-15

Maintainer Malte S. Kurz <malte.simon.kurz@uni-hamburg.de>

Description Routines for two different test types, the Constant Conditional Correlation (CCC) test and the Vectorial Independence (VI) test are provided (Kurz and Spanhel (2017) <arXiv:1706.02338>). The tests can be applied to check whether a conditional copula coincides with its partial copula. Functions to test whether a regular vine copula satisfies the so-called simplifying assumption or to test a single copula within a regular vine copula to be a (j-1)-th order partial copula are available. The CCC test comes with a decision tree approach to allow testing in high-dimensional settings.

License MIT + file LICENSE

Imports Rcpp (>= 0.11.4), VineCopula (>= 2.0.5), numDeriv, ggplot2(>= 2.0.0), gridExtra, methods

LinkingTo Rcpp, RcppArmadillo

Suggests testthat, covr

BugReports https://github.com/MalteKurz/pacotest/issues

NeedsCompilation yes

Author Malte S. Kurz [aut, cre]

Repository CRAN

Date/Publication 2020-12-15 13:10:02 UTC

R topics documented:

pacotest-package .......................................................... 2
pacotest ................................................................. 3
pacotestRvineSeq ....................................................... 6
pacotestRvineSingleCopula ........................................... 8
pacotestset ............................................................. 10
Description

The pacotest package provides functions, which allow to test for partial copulas and the simplifying assumption in vine copulas. The package consists of two different test types, the Constant Conditional Correlation (CCC) test and the Vectorial Independence (VI) test. The function pacotestset can be used to create and alter pacotest options lists and the function pacotest can be used to test for the partial copula and the simplifying assumption for a single bivariate conditional copula.

The function pacotestRvineSeq can be used with a RVineMatrix from the VineCopula-package to test all pair-copulas being building blocks in a R-vine copula to be (j-1)-th order partial copulas, which is equivalent to testing the simplifying assumption. A single building blog of a R-vine copula could be tested to be a (j-1)-th order partial copula by applying the function pacotestRvineSingleCopula to a RVineMatrix from the VineCopula-package.

Author(s)

Malte S. Kurz

References


See Also

Development for pacotest can be followed via the GitHub repository at https://github.com/MalteKurz/pacotest.
The function can be used to test for the partial copula and the simplifying assumption for a bivariate conditional copula using different tests. Two different test types, the Constant Conditional Correlation (CCC) test and the Vectorial Independence (VI) test are implemented. For all tests different options can be set by generating a pacotest options list using the pacotestset function.

**Arguments**

- **U** A \((n \times 2)\) matrix of \([0,1]\) data (probability integral transforms), which are the arguments of the conditional copula of \((Y,Z)|W\) for which the simplifying assumption should be tested. The first column is given by the conditional distribution function of \(Y|W\) evaluated at the observed values of \(Y\) and \(W\). Analogously, the second column is defined as the conditional distribution function of \(Z|W\) evaluated at the observed values of \(Z\) and \(W\). If the probability integral transforms are obtained from the partial vine copula (PVC), i.e., partial probability integral transforms (PPITs) are used, the function can be used to test for \((j-1)\)-th order partial copulas.

- **W** A \((n \times K)\) matrix of observed values for the vector of random variables on which the conditioning is done.

- **pacotestOptions** A options list generated by the pacotestset function or the test type as a string, i.e., CCC or VI.

**Details**

Applying a test with default options (cf. pacotestset) and with known (i.e., not estimated) PITs (probability integral transforms) in \(U\).

```r
out = pacotest(U,W,'CCC')
```

```r
out = pacotest(U,W,'VI')
```

Applying a test with options specified in pacotestOptions

```r
out = pacotest(U,W,pacotestOptions)
```

Note that when calling pacotest\((U,W,'CCC')\), the default options for the CCC test are used (cf. pacotestset), but the two parameters withEstUncert = FALSE and estUncertWithRanks = FALSE are altered. In contrast when calling pacotestOptions = pacotestset('CCC'), the two parameters are set to withEstUncert = TRUE and estUncertWithRanks = TRUE. For the CCC test, under the default setting, it is assumed that estimated PPITs are provided and the test statistic is
computed under consideration of estimation uncertainty of the probability integral transforms, i.e., withEstUncert = TRUE and estUncertWithRanks = TRUE. To apply pacotest with withEstUncert = TRUE, three additional inputs have to be provided (data, svcmDataFrame and cPitData).

In the vine copula context, PPTs are usually estimated and not known. Therefore, in the vine copula context it is recommended to use the functions pacotestRvineSeq or pacotestRvineSingleCopula instead of pacotest. These functions automatically pass through the additional arguments data, svcmDataFrame, cPitData to the function pacotest and the CCC test can be applied in its default setting with consideration of estimation uncertainty of the probability integral transforms, i.e., withEstUncert = TRUE and estUncertWithRanks = TRUE.

Value

A list which can, depending on the chosen test, consist of the following elements:

- pValue: The p-value of the test.
- testStat: The value of the test statistic.
- decisionTree: The decision tree used to partition the support Lambda0 of the conditioning variable W. It is provided as a list consisting of three nodes (CentralNode, LeftNode and RightNode) represented as lists and the variable LeavesForFinalComparison. Each node consists of the Variable used to perform the split, the corresponding Quantile and Threshold.
- S: The bootstrapped values of the test statistic (only for the test type VI).

Author(s)

Malte S. Kurz

References


See Also

pacotest-package, pacotestset, pacotestRvineSeq, pacotestRvineSingleCopula

Examples

# Generate an options list, e.g., the constant conditional correlation (CCC)
# test with default options. We use known PITs and don't estimate the parameters
# in the lower trees of the vine copulas and therefore additionally alter the
# two parameters withEstUncert and estUncertWithRanks to FALSE.
pacotestOptions=pacotestset(testType='CCC', withEstUncert = FALSE, estUncertWithRanks = FALSE)
# Use the specified options to test for the simplifying assumption

#### Example 1: Non-simplified three-dim. C-Vine ####

# Simulate from a three-dimensional C-Vine copula with $C_{12}$ and $C_{13}$
# being product copulas and $C_{23|1}$ being a Frank copula with
# functional parameter $\theta(x_{1}) = (4x_{1}-2)^3$

N = 500
X = matrix(runif(3*N),N,3)
theta = (4*X[,1]-2)^3
etheta = expm1(-theta);
X[,3] = -1/theta*log(1+etheta/(exp(-theta*X[,2])*(1/X[,3]-1)+1));

Result = pacotest(X[,c(2,3)],X[,1],pacotestOptions)
Result$pValue

#### Example 2: Non-simplified three-dim. C-Vine ####

# Simulate from a three-dimensional C-Vine copula with $C_{12}$ and $C_{13}$
# being product copulas and $C_{23|1}$ being a Frank copula with
# functional parameter $\theta(x_{1}) = 12 + 8\sin(0.4(3x_{1}+2)^2)$

X = matrix(runif(3*N),N,3)
theta = 12 + 8*sin(0.4*(3*X[,1]+2)^2)
etheta = expm1(-theta);
X[,3] = -1/theta*log(1+etheta/(exp(-theta*X[,2])*(1/X[,3]-1)+1));

Result = pacotest(X[,c(2,3)],X[,1],pacotestOptions)
Result$pValue

#### Example 3: Simplified three-dim. C-Vine ####

# Simulate from a three-dimensional C-Vine copula with $C_{12}$ and $C_{13}$
# being Clayton copulas with parameter $\theta$ and $C_{23|1}$ being a Clayton copula with
# functional parameter $\theta(x_{1}) = \theta / (1+\theta)$

W = matrix(runif(3*N),N,3)
X = matrix(NA,N,3)
theta = 2
X[,1] = W[,1]
X[,2] = (W[,1]ˆ(-theta) * (W[,2]ˆ((-theta)/(1+theta))-1)+1)ˆ(-1/theta);
theta_23_1 = theta / (1+theta)
X[,3] = (W[,2]ˆ(-theta_23_1) * (W[,3]ˆ((-theta_23_1)/(1+theta_23_1))-1)+1)ˆ(-1/theta_23_1);
X[,3] = (W[,1]ˆ(-theta) * (X[,3]ˆ((-theta)/(1+theta))-1)+1)ˆ(-1/theta);

# Get pseudo-observations from the conditional copula $C_{23|1}$
U = matrix(NA,N,2)
U[,1] = (X[,1]ˆtheta*W[,2]ˆ((-theta)-1)+1)ˆ(-1+theta/theta);
U[,2] = (X[,1]ˆtheta*W[,3]ˆ((-theta)-1)+1)ˆ(-1+theta/theta);
Result = pacotest(U,X[,1],pacotestOptions)
Result$pValue
**Description**

The function can be used to test the simplifying assumption for R-vine copulas in a sequential manner. Each pair-copula from the second tree on is tested to be a \((j-1)\)-th order partial copula. To apply the function one needs to provide the data and a specified/estimated R-vine copula model in form of a RVineMatrix from the VineCopula-package. Additionally, a pacotest options list, which can be generated with the pacotestset function, needs to be provided.

**Usage**

```
pacotestRvineSeq(data, RVM, pacotestOptions, level = 0.05, illustration = 2, stopIfRejected = TRUE)
```

**Arguments**

- **data**
  A \((n \times d)\) matrix (or data frame) of \([0,1]\) data (i.e. uniform margins).

- **RVM**
  An RVineMatrix object (VineCopula-package) which includes the structure, the pair-copula families and parameters of an R-vine copula.

- **pacotestOptions**
  A options list generated by the pacotestset function or the test type as string, i.e., CCC or VI.

- **level**
  The level of the test.

- **illustration**
  Either 1 or 2. If illustration = 1, the p-value for each test for a \((j-1)\)-th order partial copula is displayed. If illustration = 2, a progress information is displayed for each tree. It consists of the individual test level and the number of H0 rejections.

- **stopIfRejected**
  A logical variable indicating whether the sequential test procedure should be stopped in the first tree where an H0 for one of the conditional copulas is rejected.

**Value**

A list consisting of the following elements:

- **pacotestResultLists**
  A matrix in the same structure like the Matrix, family, par, etc. entries in the RVineMatrix object from the VineCopula-package. Each entry of the matrix is a list containing the test result from a test for a \((j-1)\)-th order partial copula obtained from a call to pacotest. Depending on the chosen test, it could consist of different elements. A documentation of the pacotestResultLists can be found in the documentation of pacotest.
A matrix in the same structure like the `Matrix`, `family`, `par`, etc. entries in the `RVineMatrix` object from the `VineCopula` package. Each entry of the matrix is a p-value corresponding to the test result from a test for a (j-1)-th order partial copula.

A data frame summarizing the test results. The first column, `Tree`, is the tree number. The second column, `NumbOfRejections`, is the number of rejections in the corresponding tree. The third column, `IndividualTestLevel`, is the level at which each individual test has been performed. The fourth column, `Interpretation`, provides an interpretation of the test result.

Malte S. Kurz


pacotest-package, pacotest, pacotestset, pacotestRvineSingleCopula

# Sample data and R-vine copula selection are taken
# from the documentation of RVineStructureSelect
# of the VineCopula package.

data(daxreturns, package = "VineCopula")
dataSet = daxreturns[1:750, 1:4]

# Specify an R-vine copula model
# (can be obtained by calling: RVM = VineCopula::RVineStructureSelect(dataSet))
vineStructure = matrix(c(3, 4, 1, 2, 0, 2, 1, 2, 0, 0, 1, 4, 0, 0, 4, 4, 4), 4, 4)
families = matrix(c(0, 5, 2, 2, 0, 0, 2, 14, 0, 0, 14, 0, 0, 0, 14), 4, 4)
par = matrix(c(0, 0.8230664, 0.1933472, 0.6275062,
              0, 0, 0.2350109, 1.6619945,
              0, 0, 1.599363, 0,
              0, 0, 0, 0), 4, 4)
par2 = matrix(c(0, 0, 11.757700, 4.547847,
               0, 0, 17.15717, 0,
               0, 0, 0, 0), 4, 4)
RVM = VineCopula::RVineMatrix(vineStructure, families, par, par2)

# Specify a pacotestOptions list:
pacotestOptions = pacotestset('CCC')

# Test for the simplifying assumption.
pacotestResultList = pacotestRvineSeq(dataSet, RVM, pacotestOptions)

pacotestRvineSingleCopula

Testing for a Single (j-1)-th Order Partial Copula in a R-Vine Copula

Description

The function can be used to test a single copula in a R-vine copula to be a (j-1)-th order partial copula. To apply the function one needs to provide the data and a specified/estimated R-vine copula model in form of a RVineMatrix from the VineCopula-package. Additionally, a pacotest options list, which can be generated with the pacotestset function, needs to be provided.

Usage

pacotestRvineSingleCopula(data, RVM, pacotestOptions, tree, copulaNumber)

Arguments

data A (n x d) matrix (or data frame) of [0,1] data (i.e. uniform margins).

RVM An RVineMatrix object (VineCopula-package) which includes the structure, the pair-copula families and parameters of an R-vine copula.

pacotestOptions A options list generated by the pacotestset function or the test type as string, i.e., CCC or VI.

tree The tree number (j>=2) of the copula which should be tested to be a (j-1)-th order partial copula.

copulaNumber The number (1<= copulaNumber <= j-1) of the copula in the normalized RVineMatrix which should be tested to be a (j-1)-th order partial copula.

Value

A list which can, depending on the chosen test, consist of the following elements:

pValue The p-value of the test.

testStat The value of the test statistic.

decisionTree The decision tree used to partition the support Lambda0 of the conditioning variable W. It is provided as a list consisting of three nodes (CentralNode, LeftNode and RightNode) represented as lists and the variable LeavesForFinalComparison. Each node consists of the Variable used to perform the split, the corresponding Quantile and Threshold.

S The bootstrapped values of the test statistic (only for the test type VI).
Author(s)

Malte S. Kurz

References


See Also

pacotest-package, pacotest, pacotestset, pacotestRvineSeq

Examples

# Sample data and R-vine copula selection are taken
# from the documentation of RVineStructureSelect
# of the VineCopula package.

# Obtain sample data
data(daxreturns, package = "VineCopula")
dataSet = daxreturns[1:750, 1:4]

# Specify an R-vine copula model
# (can be obtained by calling: RVM = VineCopula::RVineStructureSelect(dataSet))
vineStructure = matrix(c(3, 4, 1, 2, 0, 2, 4, 1, 0, 1, 4, 0, 0, 4), 4, 4)
families = matrix(c(0, 5, 2, 2, 0, 0, 2, 14, 0, 0, 14, 0, 0, 0, 4), 4, 4)
par = matrix(c(0, 0, 0.8230664, 0.1933472, 0.6275062,
              0, 0, 0, 0.2350109, 1.6619945,
              0, 0, 0, 0, 1.599363), 4, 4)
par2 = matrix(c(0, 0, 0, 0, 11.757700, 4.547847,
               0, 0, 17.15717, 0,
               0, 0, 0, 0, 0, 0, 0, 0, 4, 4)
RVM = VineCopula::RVineMatrix(vineStructure, families, par, par2)

# Specify a pacotestOptions list:
# For illustrating the functioning of the decision tree,
# grouped scatterplots and a decision tree plot are activated.
pacotestOptions = pacotestset(testType = "CCC",
                              groupedScatterplots = TRUE,
                              decisionTreePlot = TRUE)

# Test for a 2-nd order partial copula
# corresponding to the variables BAYN.DE, BMW.DE
# and conditioning set ALV.DE, BAS.DE
tree = 3
copulaNumber = 1
pacotestResultList = pacotestRvineSingleCopula(dataSet, RVM,
pacotestOptions, tree, copulaNumber)

pacotestset

Create and Alter a Pacotest Options List

Description

The function creates or updates a list object, which is required for applying the pacotest function.

Arguments

pacotestOptions

A options list for the pacotest function generated by the pacotestset function.

testType

A string which specifies the type of the test for testing the simplifying assumption.

Possible values: CCC | VI

grouping

For testType = CCC:

The grouping method which is used to obtain a partitioning of the support of the conditioning variable W.

Possible values: TreeCCC | SumMedian | SumThirdsI | SumThirdsII | SumThirdsIII | SumQuartiles | ProdMedian | ProdThirdsI | ProdThirdsII | ProdThirdsIII | ProdQuartiles | TreeEC | TreeECOV

expMinSampleSize

For testType = CCC with grouping = TreeCCC | TreeECOV | TreeEC:

The minimum number of observations which are allocated to a group in the decision tree learning process. The default value is 100.

aggInfo

For testType = CCC with grouping = TreeCCC | TreeECOV | TreeEC:

The method used for aggregating information in the conditioning set. The information in the conditioning set can be aggregated by either taking the mean of all variables or the pairwise mean. The result is added as an additional variable which can be used by the decision tree to partition the support of the conditioning variable W.

Possible values: none | meanAll | meanPairwise

withEstUncert

For testType = CCC:

A logical variable indicating whether the asymptotic-variance covariance matrix of the estimated correlations should be corrected for the estimation uncertainty of the probability integral transforms.

estUncertWithRanks

For testType = CCC:

A logical variable indicating whether the asymptotic-variance covariance matrix of the estimated correlations should be corrected for the estimation uncertainty induced by using a semiparametric estimator for the vine copula, i.e., empirical cdf’s for the univariate margins and parametric copula families as building blocks of the R-vine copula.
**finalComparison**

For testType = CCC with grouping = TreeCCC | TreeECOV | TreeEC:
A variable specifying whether at the end of the decision tree all subsets being part of the partition are compared against each other or whether only the pair with the highest value of the test statistic is used.
Possible values: pairwiseMax | all

**penaltyParams**

For testType = CCC with grouping = TreeCCC | TreeECOV | TreeEC:
A vector of length two, specifying the functional form of the penalty. The penalty is a function of the sample size n and chosen to be \(\lambda(n) = cn^{-\beta}\). The first entry of the vector is specifying the level c of the penalty and needs to be a positive real number. The second entry of the vector is specifying the power beta of the penalty and needs to be chosen from the interval (0,1).

**gamma0Partition**

For testType = CCC with grouping = TreeCCC | TreeECOV | TreeEC:
The gamma0 partition. I.e., the partition which is favoured via the penalty under the H0.
Possible values: SumMedian | SumThirdsI | SumThirdsII | SumThirdsIII | SumQuartiles | ProdMedian | ProdThirdsI | ProdThirdsII | ProdThirdsIII | ProdQuartiles

**groupedScatterplots**

For testType = CCC:
A logical whether grouped scatterplots should be produced.

**decisionTreePlot**

For testType = CCC:
A logical whether the partition of the support of W should be illustrated as a decision tree plot.

**numbBoot**

For testType = VI:
The number of bootstrap replications for computing p-values using the multiplier bootstrap approach.

### Details

Calling without any arguments prints all possible options.

```r
pacotestset()
```

Calling with a string, that specifies the test type, gives back a option list with the default values corresponding to each test.

```r
pacotestOptions = pacotestset('CCC')
pacotestOptions = pacotestset('VI')
```

Calling with pairs of parameter names and values creates an pacotestOptions list in which the named parameters have the specified values.

```r
pacotestOptions = pacotestset('Name1',Value1,'Name2',Value2,...)
```
Calling with an existing pacotestOptions list checks the list for consistency.

pacotestset(pacotestOptions)

Calling with an existing pacotestOptions list and pairs of parameter names and values creates a copy of the existing list, where the named parameters are updated with the provided values.

pacotestOptionsNew = pacotestset(pacotestOptions,'Name1',Value1,'Name2',Value2,...)

Value

The function returns a pacotestOptions list which can be used as input argument for the functions pacotest, pacotestRvineSeq and pacotestRvineSingleCopula.

Author(s)

Malte S. Kurz

References


See Also

pacotest-package, pacotest, pacotestRvineSeq, pacotestRvineSingleCopula
Index

pacotest, 2, 3, 4, 6, 7, 9, 10, 12
pacotest-package, 2
pacotestRvineSeq, 2, 4, 6, 9, 12
pacotestRvineSingleCopula, 2, 4, 7, 8, 12
pacotestset, 2–4, 6–10, 10
RVineMatrix, 2, 6–8
VineCopula-package, 2, 6–8