Package 'pencopulaCond'

October 14, 2022

Type Package

Title Estimating Non-Simplified Vine Copulas Using Penalized Splines

Version 0.2

Date 2017-05-31

Depends R (>= 3.1.1), lattice, fda, latticeExtra, pacotest

Suggests MASS

Imports quadprog, doParallel, foreach, TSP, igraph

Author Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

Maintainer Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

Description Estimating Non-Simplified Vine Copulas Using Penalized Splines.

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2017-05-31 19:29:36 UTC

R topics documented:

`pencopulaCond-package` *Estimating Non-Simplified Vine Copulas Using Penalized Splines*

Description

Estimating Non-Simplified Vine Copulas Using Penalized Splines

Details

Package:	pencopulaCond
Type:	Package
Version:	0.2
Date:	2017-05-31
License: GPL (>= 2) LazyLoad:	yes

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

Examples

```
#Simulating from a three-dimensional frank copula with
#kendell's tau equal to 0.25, sample size N.set=100.
#Please enlarge N.set for further studies.
#require(copula)
#N.set<-100
#cop <- archmCopula(family = "frank", dim = 3, param =2.39)
#parMarg<-list(list(min=0,max=1),list(min=0,max=1),list(min=0,max=1))
```

```
#distr.cop <- mvdc(cop, margins=rep("unif",3), paramMargins = parMarg,marginsIdentical=TRUE)
 #c.X <- rMvdc(mvdc=distr.cop, n=N.set)
 #Y <- punif(c.X)
 #vine.copula<-vine(Y,d=2,d2=2,D=4,D3=4,q=1,m=2,cores=1,lambda=c(10000,100))
```

`cal.Dvine` *Estimating Non-Simplified Vine Copulas Using Penalized Splines*

Description

Calculating the density of the estimated Dvine at the point(s) val.

Usage

```
cal.Dvine(obj,val)
```

Arguments

`obj`	object of class 'penDvine', result of 'Dvine'.
`val`	Values in which the current Dvine should be evaluated.

Details

The current Dvine is evaluated in val and the corresponding density values are returned.

Value

The returing values are the density of the current Dvine at the point(s) 'val'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

cal.vine *Estimating Non-Simplified Vine Copulas Using Penalized Splines*

Description

Calculating the density of the estimated regular vine at the point(s) val.

Usage

```
cal.vine(obj,val,cores)
```

Arguments

obj	Vine object of class 'pencopula'.
val	Values in which the current R-vine should be evaluated.
cores	Default=NULL, the number of cpu cores used for parallel computing can be specified.

Details

The current R-vine is evaluated in val and the corresponding density values are returned.

Value

The returing values are the density of the current R-vine at the point(s) 'val'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

Derv1 *Calculating the first derivative of the pencopula likelihood function w.r.t. parameter b*

Description

Calculating the first derivative of the pencopula likelihood function w.r.t. parameter b.

Usage

```
Derv1(penden.env,temp.lam=FALSE,temp.ck=FALSE)
```

Arguments

penden.env	Containing all information, environment of pencopula().
temp.lam	Calculating with temporal smoothing parameter lambda
temp.ck	Calculating with temporal weights ck of the spline basis functions

Value

Derv1.pen	first order derivation of the penalized likelihood.

Derv1.pen is saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

Derv2 *Calculating the second order derivative with and without penalty.*

Description

Calculating the second order derivative with and without penalty.

Usage

```
Derv2(penden.env, temp.lam = FALSE,temp.ck=FALSE,lam.fit=NULL)
```

Arguments

penden.env	Containing all information, environment of pendensity()
temp.lam	Calculating with temporal smoothing parameter lambda
temp.ck	Calculating with temporal weights ck of the spline basis functions
lam.fit	Indicating if the iterations for a new lambda are running

Details

We approximate the second order derivative in this approach with the negative fisher information.

Value

Derv2.pen	second order derivative w.r.t. beta with penalty
Derv2.cal	second order derivative w.r.t. beta without penalty. Needed for calculating of e.g. AIC.

Derv2.cal and Derv2.pen are saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`distr.func.help` *These functions are used for calculating the integral of the B-spline density basis.*

Description

These functions cooperate with each other for calculating the integral of the B-spline density basis. functions. 'distr.func.help' is the main program, calling 'poly.part', calculating the integral of the B-spline density basis in sections between neighbouring knots. 'distr.func.help' calculates analytical functions of the integral. Therefore the function 'poly.part' is needed to construct these functions.

Usage

```
distr.func.help(base,knots,penden.env,q,y,index)
poly.part(i,j,knots,help.env,q, yi=NULL, poly=FALSE)
```

Arguments

base	values of the considered B-spline basis
knots	knots of the considered B-spline basis
penden.env	Containing all information, environment of pencopula()
q	degree of the B-Spline basis
y	data of the marginal B-spline basis
index	columns of the whole B-spline basis, each hierarchy level is integrated separately
i	internal values for calculating the polynomials of each B-Spline
j	internal values for calculating the polynomials of each B-Spline

help.env	internal environment of pencopula() for calculating the integral
yi	internal values for calculating the polynomials of each B-Spline
poly	internal value, TRUE/FALSE

Value

distr.func.help	creating environment 'help.env', creating help points between each two neighbouring knots and calculates the integral each basis
poly.part	using in 'distr.func.help' for creating the polynomial functions of each interval of each two neighbouring knots

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

---

f.hat.val *Calculating the actual fitted values 'f.hat.val' of the estimated density function*

Description

Calculating the actual fitted values of the response, depending on the actual parameter set b

Usage

```
f.hat.val(penden.env,cal=FALSE,temp=FALSE)
```

Arguments

penden.env	Containing all information, environment of pencopula()
cal	if TRUE, the final weights of one iteration are used for the calculation of the fitted values.
temp	if TRUE, the iteration for optimal weights is still in progress and the temporary weights are used for calculation of the fitted values.

Value

f.hat.val	Fitted values for the current coefficents

. f.hat.val is saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`hierarch.bs` *Construction of the hierarchical B-spline density basis.*

Description

Construction of the hierarchical B-spline density basis.

Usage

```
hierarch.bs(x, d, plot.bsp, typ, penden.env, int=FALSE)
```

Arguments

`x`	Marginal data for construction.
`d`	Hierarchy level of the marginal hierarchical B-spline density.
`plot.bsp`	Default = FALSE. If TRUE, each B-spline basis is plotted.
`typ`	typ==1 without open B-splines at the boundary typ==2 with open B-splines at the boundary.
`penden.env`	Containing all information, environment of pencopula().
`int`	Default = FALSE. If TRUE, the integral of the hierarchical B-spline density basis is calculated (used for the distribution function of the estimation).

Details

First, the marginal hierarchical B-spline density basis is constructed for each covariate 'B.tilde'.

After the construction of each marginal basis, the hierarchical B-spline density basis is built in the main program pencopula(), using an object 'Index.basis.D' (saved in the environment). 'Index.basis.D' notes which compenent of the marginal basis has to be selected. In the main program the object 'tilde.Psi.d.D' is constructed. D refers to the maximum hierachy level and 'd' is the hierarchy level of the marginal hierarchical B-spline.

Value

`B.tilde`	'B.tilde' is the hierarchical B-spline density basis, returned by 'hierarch.bs'.
`int.B.tilde`	If 'int=TRUE', the integral of the hierachical B-spline density basis is calculated and returned by 'hierarch.bs'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`knots.start` *Calculating the knots.*

Description

Calculating the equidistant knots for the estimation. Moreover, transformation of the knots are possible.

Usage

```
knots.start(penden.env)
knots.transform(d,alpha = 0, symmetric = TRUE)
knots.order(penden.env)
```

Arguments

`penden.env`	Containing all information, environment of pencopula()
`d`	Hierarchy level of the marginal hierarchical B-spline basis.
`alpha`	Default = 0. Alpha is a tuning parameter, shifting the knots.
`symmetric`	Default = TRUE. If FALSE, the knots are selected without symmetry.

Details

'Knots.order' sorts the knots in the order, in which they disappear in the hierarchical B-spline basis.

Value

`knots`	Selected and sorted marginal knots for the estimation.
`knots.help`	Extended set of knots. It is needed for calculating the distribution function, help points for the integration of the B-spline density basis.
`k.order`	Order of the knots, corresponding to their order in the hierarchical B-spline density basis.
`knots.t`	The knots ordered with 'k.order' for further fucntions.
`tilde.Psi.knots.d`	Hierarchical B-Spline density basis for 'knots'.
`tilde.Psi.knots.d.help`	Hierarchical B-Spline density basis for 'knots.help'.

All values are saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

---

marg.likelihood *Calculating the marginal likelihood*

Description

Calculating the marginal likelihood of paircopula().

Usage

```
marg.likelihood(penden.env,pen.likelihood,temp=FALSE)
```

Arguments

penden.env	Containing all information, environment of paircopula().
pen.likelihood	Actual penalized likelihood for calculation, temporary or not.
temp	Default=FALSE, indicating if temporary values throughout iteration are calculated.

Value

marg.log.like	Marginal log-likelihood, saved in the environment

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

my.bspline *my.bspline*

Description

'my.bspline' Integrates the normal B-Spline basis to a B-spline density basis. The dimension of the basis depends on the input of number of knots 'k' and of the order of the B-spline basis 'q'. 'int.my.bspline' is a function for transformation of open B-spline basis at the boundary to become a B-spline basis density.

Usage

```
my.bspline(h, q, knots, y, K, plot.bsp, typ)
int.my.bspline(help.env)
```

Arguments

h	if equidistant knots are used (default in pencopula()), h is the distance between two neighbouring knots
q	selected order of the B-spline basis
knots	selected values for the knots
y	values of the response variable
K	the number of knots for the construction of the base
plot.bsp	Indicator variable TRUE/FALSE if the integrated B-spline basis should be plotted
typ	typ==1 without open B-splines at the boundary typ==2 with open B-splines at the boundary
help.env	Internal environment of my.bspline().

Details

Firstly, the function constructs the B-spline basis to the given number of knots 'K' and the given locations of the knots.

Value

base.den	The integrated B-Spline base of order q
stand.num	The coefficients for standardization of the ordinary B-Spline basis
knots.val	This return is a list. It consider of the used knots 'knots.val\$val', the help knots 'knots.val\$help' and the additional knots 'knots.val\$all', used for the construction of the base and the calculation of the distribution function of each B-Spline.
K	The transformed value of K, due to used order 'q' and the input of 'K'

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

---

my.IC *Calculating the AIC-value*

Description

Calculating the AIC-value and cAIC-value of the copula density estimation.

Usage

```
my.IC(penden.env,temp=FALSE)
```

Arguments

penden.env	Containing all information, environment of paircopula()
temp	Default=FALSE, if TRUE temporary values of AIC and cAIC are calculated.

Details

AIC is calculated as $AIC(\lambda) = -2 * l(\mathbf{u}, \hat{\mathbf{b}}) + 2 * df(\lambda)$

cAIC is calculated as $cAIC(\lambda) = -2 * l(\mathbf{u}, \hat{\mathbf{b}}) + 2 * df(\lambda) + \frac{2df(\lambda)(df(\lambda)+1)}{n-df(\lambda)-1}$

BIC is calculated as $BIC(\lambda) = 2 * l(\mathbf{u}, \hat{\mathbf{b}}) + 2 * df(\lambda) * log(n)$

Value

AIC	sum of twice the negative non-penalized log likelihood and df(lambda)
cAIC	sum of twice the negative non-penalized log likelihood and df(lambda) and (2df(lambda)(df(lambda)+1))/(n-df(lambda)-1)
BIC	sum of twice the non-penalized log likelihood and log(n)*df(lambda)

All values are saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`my.loop` *Iterative loop for calculating the optimal coefficients 'b'.*

Description

Calculating the optimal coefficients 'b' iteratively, using quadratic programing.

Usage

```
my.loop(penden.env)
```

Arguments

`penden.env`	Containing all information, environment of pencopula()

Details

'my.loop' optimates the log-likelihhod iteratively. Therefore, the routine checks the relative chance in the weights and stops the iteration, if the relative change of all weights 'b' is less than one percent. During the calculations of new weights 'b' in the routine 'new.weights', most of the values are called '.temp'. This add on unterlines the temporarily values. 'my.loop' checks the relative change in the weights. If the change is greater than one percent, the the real values are overwritten with the '.temp' values.

Value

`liste`	The results of each iteration are written in a matrix called 'liste', saved in the environment. 'liste' contains the penalized log-likelihood, the log-likelihood, 'lambda' and the weights 'b'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`my.positive.definite.solve` *my.positive.definite.solve*

Description

Reverses a quadratic positive definite matrix.

Usage

```
my.positive.definite.solve(A, eps = 1e-15)
```

Arguments

`A`	quadratic positive definite matrix
`eps`	level of the lowest eigenvalue to consider

Details

The program makes an eigenvalue decomposition of the positive definite matrix A and searches all eigenvalues greater than eps. The value of return is the inverse matrix of A, constructed with the matrix product of the corresponding eigenvalues and eigenvectors.

Value

The return is the inverse matrix of A.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

new.weights *Calculating new weights b.*

Description

Calculating new weights b using quadratic programing.

Usage

```
new.weights(penden.env,start=FALSE)
```

Arguments

penden.env	Containing all information, environment of pencopula()
start	Default=FALSE, for the first calculation some specifications are introduced.

Details

If the quadratic program does not find a new feasible solution, the whole program terminates. For solving the quadratic program, we use the function 'solve.QP' from the R-package 'quadprog'.

Value

ck.val.temp	Calculated new values for the weights 'b'. The add on 'temp' means, that there is a check in the next step if the weights 'b' have been converted or not. If converted, the new values 'ck.val.temp' are unnoted. If not converted, 'ck.val.temp' become the ordinary 'ck.val' for the next iteration. This check is done in my.loop.

'ck.val.temp' is saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

pen.log.like *Calculating the log likelihood*

Description

Calculating the considered log likelihood.

Usage

```
pen.log.like(penden.env,cal=FALSE,temp.lam=FALSE,temp.ck=FALSE)
```

Arguments

penden.env	Containing all information, environment of pencopula()
cal	if TRUE, the final weights of one iteration are used for the calculation of the penalized log likelihood.
temp.lam	Calculating with temporal smoothing parameter lambda
temp.ck	Calculating with temporal weights ck of the spline basis functions

Details

The calculation depends on the estimated weights b, the penalized hierarchical B-splines Phi and the penalty paramters lambda.

Value

pen.log.like	Penalized log likelihood of the copula density.
log.like	Log-Likelihood of the copula density.

The values are saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`penalty.matrix` *Calculating the penalty matrix P(lambda)*

Description

Calculating the penalty matrix P depends on the number of covariates 'p', the order of differences to be penalized 'pen.order', the number of observations 'n' and the penalty parameters 'lambda".

Usage

```
penalty.matrix(penden.env, temp = FALSE)
```

Arguments

`penden.env`	Containing all information, environment of pencopula().
`temp`	If TRUE, the iteration for a new 'b' is not finished and a temporary penalty matrix is calculated, default = FALSE.

Value

`DDD.sum` Penalty matrix P

Matrix is saved in the environment.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`pencopula` *Calculating penalized (conditional) copula density with penalized hierarchical B-splines*

Description

Calculating penalized (conditional) copula density with penalized hierarchical B-splines

Usage

```
pencopula(data,d=3,D=d,q=1,base="B-spline",max.iter=20,test.ind=FALSE,
         lambda=c(100,100),pen.order=2,data.frame=parent.frame(),cond=FALSE,
         fix.lambda=FALSE,id=NULL)
```

Arguments

data	'data' contains the data. 'data' has to be a matrix or a data.frame. The number of columns of 'data' is p.
d	refers to the hierachy level of the marginal hierarchical B-spline, default is d=3.
D	referes to the maximum hierachy level, default is D=3. If D<d, it follows D<-d.
q	degree of the marginal hierarchical B-spline.
base	By default, the used marginal basis is a 'B-spline'. Second possible option is 'Bernstein', using a Bernstein polynomial basis.
max.iter	maximum number of iteration, the default is max.iter=20.
test.ind	Default=FALSE. If test.ind=TRUE, the fitted log-likelihood of each pair-copula is evaluated. If ("log.like"/"n"<0.001), where "n" is the sample size, the program set the corresponding pair copula as independence copula. We do not use this in our simulations or applications in the article.
lambda	p-dimensional vector of penalty parameters, the values can be different. Default is lambda=c(100,100).
pen.order	The order of differences for the penalization, default is pen.order=2.
data.frame	reference to the data. Default reference is the parent.frame().
cond	Determining if a conditional copula is estimated. Default=FALSE, only suitable for p=3.
fix.lambda	Default=FALSE, using the algorithm in the paper for estimating the optimal penalty parameter. If fix.lambda=TRUE, lambda is constant throughout the estimation.
id	Optional, one set id to any value. Especially important for simulations, starting with several starting values for lambda.

Value

Returning an object of class pencopula. The class pencopula consists of the environment 'penden.env', which includes all calculated values of the estimation approach. For a fast overview of the main results, one can use the function 'print.pencopula()'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

`pendenForm` *Formula interpretation and data transfer*

Description

Function 'pendenForm' interprets the input 'form' of the function pencopula(),transfers the data back to the main program.

Usage

```
pendenForm(penden.env)
```

Arguments

`penden.env`	environment used in pendensity()

Value

Returning the values 'Y', the number of values 'n' and covariates 'p'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

plot.pencopula *Plot the estimated copula density or copula distribution.*

Description

The function plots the estimated copula density or the copula distrubtion for a paircopula, using the R-package 'lattice'.

Usage

```
## S3 method for class 'pencopula'
plot(x, val = NULL, marg = TRUE, plot = TRUE, int = FALSE,
main.txt = NULL, sub.txt = NULL, contour = FALSE, cond = NULL, cuts =
20, cex = 1, cex.axes = 1, cex.contour=1, xlab = NULL, ylab = NULL,
zlab=NULL, zlim=NULL, biv.margin=NULL, show.observ=FALSE,cond.cop=FALSE,
cond.par,margin.normal=FALSE,...)
```

Arguments

x	object of class 'pencopula'.
val	Default val = NULL, one can calculate the estimated density in for p-dimensional vector, e.g. val=c(0.5,1) for the two dimensional case.
marg	Default = TRUE, plotting the marginal densities.
plot	Default = TRUE, if 'FALSE' no plot is shown, e.g. for calculations with val != NULL.
int	Default = FALSE, if TRUE, the integral, i.e. the distribution of the copula density is plotted.
main.txt	Default = NULL shows 'd', 'D', the values of lambda, the penalty order and the degree of the B-splines.
sub.txt	Default = NULL shows the log-likelihood, the penalized log-likelihood and the AIC-value of the estimation.
contour	If TRUE, a contour plot is shown. Default = FALSE.
cond	Default = NULL, if the dimension of data 'p' is higher than 2, one can plot a two-dimensional conditional plot. The user specifies p-2 values for the plot, indicating with '-1'. So for a three-dimensional plot, cond=c(0,-1,-1) shows the density/distribution ith fixed first covariate and the second and third covariates vary.
cuts	Number of cuts for the contour plots, if contour=TRUE.
cex	Default = 1, determing the size of the main of the plot.
cex.axes	Default = 1, determing the size of the labels at the axes.
cex.contour	Default = 1, determing the size of the labels at the cuts of the contourplot.
xlab	Default = NULL and no text is printed at the xlab
ylab	Default = NULL and no text is printed at the ylab

zlab	Default = NULL and 'density' is printed at the zlab for int=FALSE and 'distribution' for int=TRUE.
zlim	For Default = NULL, the range of the estimated values determin zlim. Alternatively, one can suggest the range as a vector.
biv.margin	Determines for which parameter the bivariate marginal distribution/density is presented.
show.observ	Default = FALSE. If TRUE, plotting the original observation into a contourplot. For multivariate copulas the data corresponding to 'biv.margin' is plotted. Show.observ is not possible in combination with option 'cond'.
cond.cop	Default=FALSE. If cond.cop=TRUE, the object x have to be condtional copula - this option will disapper as the object itself contains this information.
cond.par	If cond.cop=TRUE, the plot is created for the conditioning argument cond.par
margin.normal	Default = FALSE. If TRUE, the plot is presented with margins following standard normal distribution.
...	further arguments

Details

For the two dimensional plots, a equidistant grid of 51 values between 0 and 1 is constructed. The plot consists of the density or distribution values in this grid points. For plots of high dimensional data (p>2), one has to fix p-2 covariates (see 'cond').

Value

If 'val' is not NULL, the function returns a matrix with the calculated density or distribution values for the set 'val'.

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

---

print.pencopula *Printing the main results of the penalized copula density estimation*

Description

Printing the call of the estimation, the used basis, lambda and the corresponding values of AIC and BIC. Need an object of class pencopula.

Usage

```
## S3 method for class 'pencopula'
print(x, ...)
```

Arguments

x	x has to be object of class pencopula
...	further arguments

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

---

vine *"Estimating Non-Simplified Vine Copulas Using Penalized Splines"*

Description

Estimating Non-Simplified Vine Copulas Using Penalized Splines

Usage

```
vine(data,d=2,d2=2,D=4,D3=6,lambda=c(100,50),type="Rvine",order.Dvine=FALSE,m=2,
cores=NULL,q=1,mod.cond=TRUE,max.iter=51,fix.lambda=FALSE,RVM=NULL,cal.cond=FALSE,
id=NULL,test.ind=FALSE,test.cond=2,lambda.search=FALSE,lam1.vec=NULL,lam2.vec=NULL)
```

Arguments

data	'data' contains the data. 'data' has to be a matrix or a data.frame with two columns.
d	refers to the hierachy level of the marginal hierarchical B-spline for copulas in the first tree of the vine, default is d=2.
d2	refers to the hierachy level of the marginal hierarchical B-spline for copulas in the second tree and in the following trees of the vine, default is d2=2.
D	referes to the maximum hierachy level for copulas in the first tree of the vine, default is D=4. If D<d, it follows D<-d.
D3	referes to the maximum hierachy level for copulas in the second tree and in the following trees of the vine, default is D3=6.
lambda	Starting values for lambda, first start values for copulas in the first tree, second start value for copulas in the second tree and in the following trees of the vine, default is lambda=c(100,50).

type	Default is type="Rvine", fitting a regular vine copula. An alternative is type="Dvine", fitting a D-vine copula.
order.Dvine	Only relevant for type="Dvine". Indicating if the first level of the Dvine is ordered, default order.Dvine=TRUE.
m	Indicating the order of differences to be penalised. Default is "m=2".
cores	Default=NULL, the number of cpu cores used for parallel computing can be specified.
q	Degree of B-splines. Default is q=1.
mod.cond	Default=TRUE. If mod.cond=FALSE each pair-copula in the vine is estimated as simplified copula. The argument test.cond varies the test for the simplyfing assumption, which is imported from the R-package pacotest.
max.iter	maximum number of iteration, the default is max.iter=51.
fix.lambda	Default=FALSE, using the algorithm in the paper for estimating the optimal penalty parameter. If fix.lambda=TRUE, lambda is constant throughout the estimation.
RVM	Default=NULL. If RVM is a RVine-Matrix, this matrix determines the structure of the vine.
cal.cond	Default=FALSE. If cal.cond=TRUE each copula in the second tree and in the following trees of the vine is estimated as conditional copula.
id	Optional, one set id to any value. Especially important for simulations, starting with several starting values for lambda.
test.ind	Default=FALSE. If test.ind=TRUE, the fitted log-likelihood of each pair-copula is evaluated. If ("log.like"/"n"<0.001), where "n" is the sample size, the program set the corresponding pair copula as independence copula. We do not use this in our simulations or applications in the article.
test.cond	If test.cond=2, testType='ECORR' is chosen for the test of the simplyfing assumption as proposed in the article. There is an additional second test available in the R-package pactotest. testType="VI" is chosen with test.cond=1.
lambda.search	TRUE/FALSE, indicating if a search about several starting values for lambda should be performed. If search is selected, the starting value 'lambda' does not work anymore.
lam1.vec	Vector of candidate values for penalty parameter lambda for copulas in the first tree of the vine.
lam2.vec	Vector of candidate values for penalty parameter lambda for copulas in the second tree and in the following trees of the vine.

Details

The calculation of the vine is done stepwise. The specifications in 'vine' are done for every pair-copula in the vine with the identical specification. There is no option to change parameters for some pair-copulas.

Value

Returning a list containing

`vine`	The estimated vine copula, an object of class 'pencopulaCond'
`log.like`	the estimated log-likelihood
`log.like.vec`	A vector with the estimated log.like.vec of each pair-copula
`AIC`	AIC value
`AIC.vec`	A vector with the estimated AIC of each pair-copula
`cAIC`	corrected AIC value
`cAIC.vec`	A vector with the estimated cAIC of each pair-copula
`d`	Used d
`d2`	Used d2
`D`	Used D
`D3`	Used D3
`order`	the used order of the first level (reported only for D-vines)
`S`	Sequence seq(1:(dim(data)[2]))
`N`	Number of observations, that is dim(data)[1]
`base`	Used basis function
`q`	Used degree of the B-spline basis
`no.cond.dens`	Estimated number of condtional copulas
`pca`	Indicating the used number of pca
`D.struc`	Used D.struc
`type`	Selected type of the vine copula
`VineMatrix`	VineMatrix, reported for type="Rvine"

Author(s)

Christian Schellhase <cschellhase@wiwi.uni-bielefeld.de>

References

Flexible Copula Density Estimation with Penalized Hierarchical B-Splines, Kauermann G., Schellhase C. and Ruppert, D. (2013), Scandinavian Journal of Statistics 40(4), 685-705.

Estimating Non-Simplified Vine Copulas Using Penalized Splines, Schellhase, C. and Spanhel, F. (2017), Statistics and Computing.

Index