Package 'GRIDCOPULA'

April 5, 2024

Type Package

Title Bivariate Copula Functions Based on Regular Grid

Description Estimates grid type bivariate copula functions, calculates some association measures and provides several copula graphics.

Version 1.0.1

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

Imports Rsolnp, fields, limSolve, ggplot2, pracma, reshape2, evmix

NeedsCompilation no

Author Rogelio Salinas Gutiérrez [aut, cre, cph]
(<https://orcid.org/0000-0002-1669-4460>),
Angélica Hernández Quintero [aut, cph]
(<https://orcid.org/0009-0000-0400-1244>),
Pedro Abraham Montoya Calzada [aut, cph]

Maintainer Rogelio Salinas Gutiérrez <rsalinas@correo.uaa.mx>

Repository CRAN

Date/Publication 2024-04-05 15:23:09 UTC

R topics documented:

`aic.grid` *Calculates the Akaike Information Criterion "AIC" of a grid type copula*

Description

This function receives a grid type copula as a parameter and returns the value of the AIC.

Usage

```
aic.grid(gc)
```

Arguments

gc	a grid type copula object.

Value

Returns a number with the AIC of a grid type copula.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
aic.grid(copula.grid)

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
aic.grid(copula.grid)
```

bic.grid *Calculates the Bayesian Information "BIC" of a grid type copula*

Description

This function receives a grid type copula as a parameter and returns the value of the BIC.

Usage

```
bic.grid(gc)
```

Arguments

gc	a grid type copula object.

Value

Returns a number with the BIC of a grid type copula.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
bic.grid(copula.grid)

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
bic.grid(copula.grid)
```

contour_grid *Draws the density / distribution function of a grid copula with contours and colors*

Description

Draws the density / distribution function of a grid copula with contours and colors

Usage

```
contour_grid(gc,
FUN = "d.grid",
color.name = "none",
color.size = 7,
show.points = FALSE,
copula.domain = TRUE,
normal.marginal = TRUE)
```

Arguments

gc	a grid type copula object.
FUN	the name of the function to be applied (d.grid for density, p.grid for distribution), default is 'd.grid'.
color.name	indicates the palette of colors
color.size	indicates the number of colors.
show.points	a logical value indicating if the data must be showed or not, default is FALSE.
copula.domain	Indicates whether it is going to be graphed in the domain of the copulas $U(0,1)$ or in the domain of the original of the variables.
normal.marginal	Indicates whether the marginals should be taken as normal distributions. The default value is TRUE, otherwise the gaussian kernel is used as marginal distribution. This argument is neccesary only if the argument copula.domain is FALSE.

Value

Returns a graph of the density / distribution.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
k <- 10
m <- 10
copula.grid <- estimate.gridCopula(U = df, k = k, m = m , method = "ml")
contour_grid(gc = copula.grid, FUN = 'd.grid', color.name = "rainbow")
contour_grid(gc = copula.grid, FUN = 'p.grid', color.name = "rainbow")

#Iris
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = k, m = m , method = "ls")
```

```
contour_grid(gc = copula.grid, FUN = 'd.grid', color.name= "rainbow",
color.size = 10, copula.domain=FALSE)
contour_grid(gc = copula.grid, FUN = 'p.grid', color.name = "rainbow",
color.size = 10, copula.domain=FALSE)
```

`contour_image_grid` *Draws the density / distribution function of a grid copula with contours and colors*

Description

Draws the density / distribution function of a grid copula with contours and colors

Usage

```
contour_image_grid(
  gc,
  FUN = "p.grid",
  u1 = seq(0, 1, length.out = 100),
  u2 = seq(0, 1, length.out = 100),
  color.name = "heat.colors",
  color.size = 40
)
```

Arguments

`gc`	a grid type copula object.
`FUN`	the name of the function to be applied (d.grid, p.grid), default is 'p.grid'.
`u1`	indicates the place for lines on axis u_1.
`u2`	indicates the place for lines on axis u_2.
`color.name`	indicates the palette of colors.
`color.size`	indicates the number of colors.

Value

Returns a graph of the density / distribution.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
```

```
df <- cbind(u,v)
k <- 10
m <- 10
copula.grid <- estimate.gridCopula(U = df, k = k, m = m , method = "ls")
contour_image_grid(gc = copula.grid, FUN = 'd.grid', color.name= "rainbow", color.size = 10)
contour_image_grid(gc = copula.grid, FUN = 'p.grid', color.name = "rainbow", color.size = 10)

#Iris
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = k, m = m , method = "ls")
contour_image_grid(gc = copula.grid, FUN = 'd.grid', color.name= "rainbow", color.size = 10)
contour_image_grid(gc = copula.grid, FUN = 'p.grid', color.name = "rainbow", color.size = 10)
```

`d.grid` *Evaluates the density of a grid type copula*

Description

Returns the corresponding density values of a grid type copula.

Usage

```
d.grid(U, V = NULL, gc)
```

Arguments

U	a matrix of size $nx2$ with the observed values. It can also be a vector of size $kx1$ with the values of the U_1 variable.
V	optional, a vector of size $kx1$ with the values of the U_2 variable.
gc	a grid type copula object.

Value

Returns a vector with the corresponding density.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
d.grid(df,gc=copula.grid)
```

```
# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
d.grid(copula.grid$U,gc=copula.grid)
```

`data.grid` *Draws the scatter plot of bivariate data in the unit square*

Description

Draws the scatter plot of bivariate data in the unit square

Usage

```
data.grid(U, draw.lines = TRUE, k = 4, m = 4)
```

Arguments

`U`	matrix of size $kx2$ with the values of both variables.
`draw.lines`	draws lines inside the unit square or not.
`k`	positive integer indicating the number of subintervals for the U_2 variable.
`m`	positive integer indicating the number of subintervals for the U_1 variable.

Value

Returns a scatter plot of bivariate data in the unit square.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
k <- 10
m <- 10
data.grid(U=df, draw.lines = FALSE, k = k, m = m)
data.grid(U=df, draw.lines = TRUE, k = k, m = m)
```

`estimate.gridCopula` *Estimates the parameters of a grid type copula*

Description

This function estimates grid type copulas by one of the following methods: maximum likelihood or least squares (See reference).

Usage

```
estimate.gridCopula(
  X = NULL,
  U = NULL,
  k = NULL,
  m = NULL,
  method = "ml",
  D.ini = NULL,
  criterion = "AIC"
)
```

Arguments

`X`	a matrix of size $nx2$ with the observed values in any domain, optional if U is provided.
`U`	a matrix of size $nx2$ with the observed values in the copula domain, optional if X is provided.
`k`	a positive integer indicating the number of subintervals for the U_2 variable.
`m`	a positive integer indicating the number of subintervals for the U_1 variable.
`method`	the selected method for estimation, can be least squares "ls" or maximum likelihood "ml". By default "ml".
`D.ini`	an optional matrix with initial density values for the estimation through maximum likelihood.
`criterion`	If the values of k and m are not specified, they will be obtained by the "AIC" or "BIC" criteria, by default "AIC".

Value

Returns a list with a matrix with the density over the grid, a matrix with the quantity of data over the grid, the number of subintervals for the U_2 variable, the number of subintervals for the U_1 variable, a matrix with the values of u_1 and u_2 in the copula domain and a matrix with the original values X.

References

Pfeifer, D., Strassburger, D., & Philipps, J. (2020). Modelling and simulation of dependence structures in nonlife insurance with Bernstein copulas. *arXiv*. Retrieved from https://arxiv.org/abs/2010.15709

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
print(copula.grid$Density)

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
print(copula.grid$Density)
```

---

image_color_grid *Draws the density of a grid copula with colors*

Description

Draws the density of a grid copula with colors

Usage

```
image_color_grid(gc, color.name = "heat.colors", color.size = 7)
```

Arguments

gc	a grid type copula object.
color.name	indicates the palette of colors.
color.size	indicates the number of colors.

Value

Returns a graph of the density.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
```

```
v <- Fy(y)
df <- cbind(u,v)
k <- 10
m <- 10
copula.grid <- estimate.gridCopula(U = df, k = k, m = m , method = "ml")
image_color_grid(gc = copula.grid, color.name = "rainbow", color.size = 10)

#Iris
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = k, m = m , method = "ml")
image_color_grid(gc = copula.grid, color.name = "rainbow", color.size = 10)
```

`measures.grid` *Returns dependency measures for a grid type copula*

Description

Returns dependency measures for a grid type copula

Usage

```
measures.grid(gc, measures = "all")
```

Arguments

`gc`	a grid type copula object.
`measures`	A vector of the measurements to calculate: "gini", "blomqvist", "tail_U", "tail_L", "rho", "tau","mi", by default "all".

Details

`"tau"` Kendall's τ, see Nelsen (2007).

`"rho"` Spearman's ρ, see Nelsen (2007).

`"blomqvist"` Blomqvist's β; computed as $4C(0.5, 0.5) - 1$, see Nelsen (2007).

`"gini"` Gini's γ, see Nelsen (2007).

`"mi"` Mutual information, see Joe (1989).

`"tail_U/tail_L"` Tail dependency, see Nelsen (2007).

Value

A list with dependence measures

References

Nelsen, R. (2007). An introduction to copulas. Springer Science & Business Media.

Joe, H. (1989). Relative Entropy Measures of Multivariate Dependence. Journal of the American Statistical Association.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
measures.grid(copula.grid)
measures.grid(copula.grid, measures = c("rho","tau","mi"))

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
measures.grid(copula.grid, measures = c("gini", "blomqvist", "tail_U", "tail_L", "rho"))
```

---

mosaic.grid *Draws the density of a grid copula with mosaics*

Description

Draws the density of a grid copula with mosaics

Usage

```
mosaic.grid(gc, number.size = 5)
```

Arguments

gc	a grid type copula object.
number.size	indicates the size of numbers.

Value

Returns a graph.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
```

```
df <- cbind(u,v)
k <- 10
m <- 10
copula.grid <- estimate.gridCopula(U = df, k = k, m = m , method = "ml")
mosaic.grid(gc = copula.grid, number.size = 5)

#Iris
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = k, m = m , method = "ml")
mosaic.grid(gc = copula.grid, number.size = 5)
```

---

p.grid *Evaluates the distribution function of a grid type copula*

Description

Returns the corresponding distribution function values.

Usage

```
p.grid(U, V = NULL, gc)
```

Arguments

U	a matrix of size $nx2$ with the observed values. It can also be a vector of size $kx1$ with the values of the U_1 variable.
V	optional, a vector of size $kx1$ with the values of theU_2 variable.
gc	a grid type copula object.

Value

Returns a vector with the corresponding distribution.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 5, m = 4 , method = "ml")
p.grid(df,gc=copula.grid)

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
p.grid(copula.grid$U,gc=copula.grid)
```

perspective.grid *Draws the density / distribution function of a grid copula with perspective*

Description

Draws the density / distribution function of a grid copula with perspective

Usage

```
perspective.grid(
  gc,
  FUN = "d.grid",
  u1 = seq(0, 1, length.out = 21),
  u2 = seq(0, 1, length.out = 21),
  ang.theta = -30,
  ang.phi = 25,
  distancia = 10
)
```

Arguments

gc	a grid type copula object.
FUN	the name of the function to be applied (d.grid, p.grid), default is 'd.grid'.
u1	indicates the place for lines on axis u_1.
u2	indicates the place for lines on axis u_2.
ang.theta	angle for the azimuthal direction.
ang.phi	angle for the colatitude.
distancia	the distance of the eyepoint from the centre of the box.

Value

Returns a graph of the density / distribution.

Examples

```
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
k <- 10
```

```
m <- 10
copula.grid <- estimate.gridCopula(U = df, k = k, m = m , method = "ml")
perspective.grid(gc = copula.grid, ang.theta = 90 , ang.phi = 80, distancia = 3)
perspective.grid(gc = copula.grid, FUN = "p.grid")

#Iris
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = k, m = m , method = "ml")
perspective.grid(gc = copula.grid, ang.theta = 90 , ang.phi = 80, distancia = 3)
perspective.grid(gc = copula.grid, FUN = "p.grid")
```

`r.grid` *Generates a random sample from a grid type copula*

Description

Generates a random sample from a grid type copula

Usage

```
r.grid(n, gc)
```

Arguments

n	an integer number indicating the size of the sample.
gc	a grid type copula object.

Value

Returns a matrix of size $nx2$ with the random sample.

Examples

```
# Generating simulated data with a transformation to the copula domain
n <- 500
x <- rgamma(n,4,1/2)
e <- rnorm(n,0,.3)
y <- sin(x+e)
Fx <- ecdf(x)
Fy <- ecdf(y)
u <- Fx(x)
v <- Fy(y)
df <- cbind(u,v)
copula.grid <- estimate.gridCopula(U = df, k = 15, m = 15 , method = "ml")
df2 <- r.grid(n = n, gc = copula.grid)
data.grid(copula.grid$U, k = 15, m = 15)
data.grid(df2, k = 15, m = 15)

# Using the Iris dataset, transformation is not mandatory
copula.grid <- estimate.gridCopula(X = iris[,1:2], k = 3, m = 7 , method = "ml")
```

```
df2 <- r.grid(n = n, gc = copula.grid)
data.grid(copula.grid$U, k = 3, m = 7)
data.grid(df2, k = 3, m = 7)
```

Index