Package ‘fCopulae’

January 7, 2023

Title    Rmetrics - Bivariate Dependence Structures with Copulae
Date     2023-01-03
Version  4022.85
Author   Diethelm Wuertz [aut],
          Tobias Setz [aut],
          Yohan Chalabi [ctb],
          Paul Smith [cre]
Maintainer Paul Smith <paul@waternumbers.co.uk>
Description Provides a collection of functions to
          manage, to investigate and to analyze bivariate financial returns by
          Copulae. Included are the families of Archimedean, Elliptical,
          Extreme Value, and Empirical Copulae.
Depends  R (>= 2.15.1), timeDate, timeSeries, fBasics, fMultivar
Imports  grDevices, graphics, stats
Suggests methods, RUnit, tcltk, mvtnorm, sn
License  GPL (>= 2)
URL      https://www.rmetrics.org
NeedsCompilation no
Repository CRAN
Date/Publication 2023-01-07 22:50:11 UTC

R topics documented:

  fCopulae-package .........................................................  2
  ArchimedeanCopulae .....................................................  6
  ArchimedeanDependency ...............................................  8
  ArchimedeanGenerator ................................................  9
  ArchimedeanModelling ................................................. 11
  CopulaClass .......................................................... 12
  CopulaEnv ............................................................. 14
  EllipticalCopulae ..................................................... 14
fCopulae-package

EllipticalDependency .................................................. 17
EllipticalGenerator ....................................................... 20
EllipticalModelling ....................................................... 23
EmpiricalCopulae .......................................................... 25
ExtremeValueCopulae ..................................................... 27
ExtremeValueDependency ............................................... 29
ExtremeValueGenerator ................................................ 31
ExtremeValueModelling ................................................ 32

Index

fCopulae-package Modelling Copulae and Dependence Structures

Description

The Rmetrics fCopulae package is a collection of functions to manage, to investigate and to analyze bivariate financial returns by Copulae. Included are the families of Archemedean, Elliptical, Extreme Value, and Empirical Copulae.

Details

Package: fCopulae
Type: Package
Version: R 3.0.1
Date: 2014
License: GPL Version 2 or later
Copyright: (c) 1999-2014 Rmetrics Association
URL: https://www.rmetrics.org

1 Introduction

The package fCopulae was written to explore and investigate bivariate copulae and dependence structures.

2 Archimedean Copulae

This chapter contains functions for analysing and modeling Archemedean copulae.

Archimedean Copula Density, Probability and Random Numbers:

darchmCopula Computes Archimedean copula density
parchmCopula Computes Archimedean copula probability
rarchmCopula Generates Archimedean copula random variates
For the Gumbel Copula we have a fast implementation.

- `rgumbelCopula`: Generates fast gumbel random variates
- `dgumbelCopula`: Computes bivariate Gumbel copula density
- `pgumbelCopula`: Computes bivariate Gumbel copula probability

**Archimedean Copula Dependency Structure:**

- `archmTau`: Returns Kendall's tau for Archimedean copulae
- `archmRho`: Returns Spearman's rho for Archimedean copulae

- `archmTailCoeff`: Computes tail dependence for Archimedean copulae
- `archmTailPlot`: Plots Archimedean tail dependence function

**Archimedean Copula Generator:**

- `archmList`: Returns list of implemented Archimedean copulae
- `archmParam`: Sets Default parameters for an Archimedean copula
- `archmRange`: Returns the range of valid alpha values
- `archmCheck`: Checks if alpha is in the valid range

- `Phi`: Computes Archimedean Phi, inverse and derivatives
- `PhiSlider`: Displays interactively generator function
- `Kfunc`: Computes Archimedean Density Kc and its Inverse
- `KfuncSlider`: Displays interactively the density and concordance

**Archimedean Copula Modeling:**

- `archmCopulaSim`: Simulates bivariate elliptical copula
- `archmCopulaFit`: Fits the parameter of an elliptical copula

**Archimedean Copula Slider:**

- `darchmSlider`: Displays interactively archimedean density
- `parchmSlider`: Displays interactively Archimedean probability
- `rarchmSlider`: Displays interactively Archimedean probability
3 Elliptical Copulae

This chapter contains functions for analysing and modeling elliptical copulae.

Elliptical Copula Density, Probability and Random Numbers:

- `dellipticalCopula` Computes elliptical copula density
- `pellipticalCopula` Computes elliptical copula probability
- `rellipticalCopula` Generates elliptical copula variates

Elliptical Copula Slider:

- `dellipticalSlider` Generates interactive plots of density
- `pellipticalSlider` Generates interactive plots of probability
- `rellipticalSlider` Generates interactive plots of random variates

Elliptical Copula Dependency Structures:

- `ellipticalTau` Computes Kendall's tau for elliptical copulae
- `ellipticalRho` Computes Spearman's rho for elliptical copulae
- `ellipticalTailCoeff` Computes tail dependence for elliptical copulae
- `ellipticalTailPlot` Plots tail dependence function

Elliptical Copula Generator:

- `ellipticalList` Returns list of implemented Elliptical copulae
- `ellipticalParam` Sets default parameters for an elliptical copula
- `ellipticalRange` Returns the range of valid rho values
- `ellipticalCheck` Checks if rho is in the valid range
- `gfunc` Generator function for elliptical distributions
- `gfuncSlider` Slider for generator, density and probability

Elliptical Copula Modeling:

- `ellipticalCopulaSim` Simulates bivariate elliptical copula
- `ellipticalCopulaFit` Fits the parameter of an elliptical copula
4 Extreme Value Copulae

This chapter contains functions for analysing and modeling extreme value copulae.

**Extremem Value Copula Density, Probability and Random Numbers:**

- devCopula: Computes extreme value copula density
- pевCopula: Computes extreme value copula probability
- revCopula: Generates extreme value copula random variates

- devSlider: Displays interactively plots of density
- pевSlider: Displays interactively plots of probability
- revSlider: Displays interactively plots of random variates

**Extreme Value Copula Dependency Structures:**

- evTau: Returns Kendall’s tau for extreme value copulae
- evRho: Returns Spearman’s rho for extreme value copulae

- evTailCoeff: Computes tail dependence for extreme value copulae
- evTailCoeffSlider: Plots extreme value tail dependence function

**Extreme Value Copula Generator:**

- evList: Returns list of implemented extreme value copulae
- evParam: Sets Default parameters for an extreme value copula
- evCheck: Checks if parameters are in the valid range
- evRange: Returns the range of valid parameter values

- Afunc: Computes Dependence function
- AfuncSlider: Displays interactively dependence function

**Extreme Value Copula Modeling:**

- evCopulaSim: Simulates bivariate extreme value copula
- evCopulaFit: Fits the parameter of an extreme value copula

5 Empirical Copula.

This chapter contains functions for analysing and modeling empirical copulae.

**Empirical Copulae Density and Probability:**

- pempiricalCopula: Computes empirical copula probability
- dempiricalCopula: Computes empirical copula density
About Rmetrics:
The fCopulae Rmetrics package is written for educational support in teaching "Computational Finance and Financial Engineering" and licensed under the GPL.

---

Archimedean Copulae

Bivariate Archimedean Copulae

Description

A collection and description of functions to investigate bivariate Archimedean copulae.

Archimedean Copulae Functions:

- `rarchmCopula` generates Archimedean copula variates,
- `parchmCopula` computes Archimedean copula probability,
- `darchmCopula` computes Archimedean copula density,
- `rarchmSlider` displays interactive plots of variates,
- `parchmSlider` displays interactive plots of probability,
- `darchmSlider` displays interactive plots of density.

Special Copulae Functions:

- `rgumbelCopula` generates Gumbel copula variates,
- `pgumbelCopula` computes Gumbel copula probability,
- `dgumbelCopula` computes Gumbel copula density.

Usage

```r
rarchmCopula(n, alpha = NULL, type = archmList())
parchmCopula(u = 0.5, v = u, alpha = NULL, type = archmList(), output = c("vector", "list"), alternative = FALSE)
darchmCopula(u = 0.5, v = u, alpha = NULL, type = archmList(), output = c("vector", "list"), alternative = FALSE)
rarchmSlider(B = 10)
parchmSlider(type = c("persp", "contour"), B = 10)
darchmSlider(type = c("persp", "contour"), B = 10)
rgumbelCopula(n, alpha = 2)
pgumbelCopula(u = 0.5, v = u, alpha = 2, output = c("vector", "list"))
dgumbelCopula(u = 0.5, v = u, alpha = 2, output = c("vector", "list"))
```

Arguments

- `alpha` [\Phi\ast[^{archmCopula}]] - the parameter of the Archimedean copula. A numerical value.
alternative [*Copula] -
Should the probability be computed alternatively ...

B [*Slider] -
the maximum slider menu value when the boundary value is infinite. By default this is set to 10.

n [rarchmCopula] -
the number of random deviates to be generated, an integer value.

output [*archmCopula] -
output - a character string specifying how the output should be formatted. By default a vector of the same length as u and v. If specified as "list" then u and v are expected to span a two-dimensional grid as outputted by the function grid2d and the function returns a list with elements \$x, \$y, and \$z which can be directly used for example by 2D plotting functions.

type [*archmCopula] -
the type of the Archimedean copula. A character string ranging beween "1" and "22". By default copula No. 1 will be chosen.

[*archmSlider] -
the type of the plot. A character string either specifying a perspective or contour plot.

u, v [*archmCopula] -
two numeric values or vectors of the same length at which the copula will be computed. If u is a list then the the \$x and \$y elements will be used as u and v. If u is a two column matrix then the first column will be used as u and the the second as v.

Value

The function pcopula returns a numeric matrix of probabilities computed at grid positions xly.

The function parchmCopula returns a numeric matrix with values computed for the Archimedean copula.

The function darchmCopula returns a numeric matrix with values computed for the density of the Archimedean copula.

The functions Phi* return a numeric vector with the values computed from the Archimedean generator, its derivatives, or its inverse.

The functions cK and cKInv return a numeric vector with the values of the density and inverse for Archimedian copulae.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.
**ArchimedeanDependency**  
* Bivariate Archimedean Copulae

## Description
A collection and description of functions to investigate bivariate Archimedean copulae.

**Archimedean Copulae Functions:**
- `archmTau` computes Kendall’s tau for Archimedean copulae,
- `archmRho` computes Spearman’s rho for Archimedean copulae,
- `archmTailCoeff` computes tail dependence for Archimedean copulae,
- `archmTailPlot` plots tail dependence for Archimedean copulae.

## Usage

```r
archmTau(alpha = NULL, type = archmList(), lower = 1.0e-10)
archmRho(alpha = NULL, type = archmList(), method = c("integrate2d", "adapt"),
         error = 1.0e-5)
archmTailCoeff(alpha = NULL, type = archmList())
archmTailPlot(alpha = NULL, type = archmList(), tail = c("Upper", "Lower"))
```

## Arguments

- `alpha` the parameter of the Archimedean copula. A numerical value.
- `error` [archmRho] - the error bound to be achieved by the integrate2d integration formula. A numeric value, by default `error=1.0e-5`.
- `lower` [archmTau] - a numeric value setting the lower bound for the internal integration function integrate.
- `tail` [archmTailPlot] - a character string, either "Upper" or "Lower" denoting which of the two tails should be displayed. By default the upper tail dependence will be considered.
- `type` the type of the Archimedean copula. A character string ranging between "1" and "22". By default copula No. 1 will be chosen.
- `method` [archmRho] - a character string that determines which integration method should be used, either "integrate2d" or "adapt". If the second method is selected the contributed R package "adapt" is required.
ArchimedeanGenerator

Value

The function \texttt{pcopula} returns a numeric matrix of probabilities computed at grid positions \texttt{x,y}.

The function \texttt{parchmCopula} returns a numeric matrix with values computed for the Archimedean copula.

The function \texttt{darchmCopula} returns a numeric matrix with values computed for the density of the Archimedean copula.

The functions \texttt{Phi*} return a numeric vector with the values computed from the Archimedean generator, its derivatives, or its inverse.

The functions \texttt{cK} and \texttt{cKInv} return a numeric vector with the values of the density and inverse for Archimedean copulae.

Author(s)

Diethelm Wuertz for the Rmetrics \texttt{R}-port.

ArchimedeanGenerator Bivariate Archimedean Copulae

Description

A collection and description of functions concerned with the generator function for the Archimedean copula and with functions for setting and checking the distributional parameters.

Functions:

- \texttt{evList}: Returns list of implemented Archimedean copulae,
- \texttt{archmParam}: Sets default parameters for an Archimedean copula,
- \texttt{archmRange}: returns the range of valid rho values,
- \texttt{archmCheck}: checks if rho is in the valid range,
- \texttt{Phi}: Computes generator Phi, inverse and derivatives,
- \texttt{PhiSlider}: displays interactively generator function,
- \texttt{Kfunc}: computes copula density and its inverse,
- \texttt{KfuncSlider}: displays interactively density function.

Usage

\texttt{archmList()}
\texttt{archmParam(type = archmList())}
\texttt{archmRange(type = archmList(), B = Inf)}
\texttt{archmCheck(alpha, type = archmList())}
ArchimedeanGenerator

Phi(x, alpha = NULL, type = archmList(), inv = FALSE, deriv = paste(0:2))
PhiSlider(B = 5)

Kfunc(x, alpha = NULL, type = archmList(), inv = FALSE, lower = 1.0e-8)
KfuncSlider(B = 5)

Arguments

alpha [Phi][*archmCopula] - the parameter of the Archemedean copula. A numerical value.

B [archmRange] - the maximum slider menu value when the boundary value is infinite. By default this is set to B=Inf.

[Slider] - the maximum slider menu value when the boundary value is infinite. By default this is set to B=5.

deriv [Phi] - an integer value. Should the function itself, deriv="0", or the first deriv="1", or second deriv="2" derivative be evaluated?

inv [Phi][Kfunc] - a logical flag. Should the inverse function be computed?

lower [Kfunc] - a numeric value setting the lower bound for the internal root finding function uniroot.

type [archmCopula][Phi][Kfunc] - the type of the Archimedean copula. A character string ranging between "1" and "22". By default copula No. 1 will be chosen.

x [Kfunc] - a numeric value or vector ranging between zero and one.
[Phi] - a numeric value or vector.

Value

The function Phi returns a numeric vector with the values computed from the Archemedean generator, its derivatives, or its inverse.

The function Kfunc returns a numeric vector with the values of the density and inverse for Archimedian copulae.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

References

RB Nelson - An Introduction to Copulas
Examples

```r
## archmList -
# Return list of implemented copulae:
archmList()
```

Description

A collection and description of functions to investigate bivariate Archimedean copulae.

Archimedean Copulae Functions:

- `archmCopulaSim` simulates an Archimedean copula,
- `archmCopulaFit` fits the parameters of an Archimedean copula.

Usage

```r
archmCopulaSim(n, alpha = NULL, type = archmList())
archmCopulaFit(u, v = NULL, type = archmList(), ...)
```

Arguments

- **alpha** [\(\Phi^*\)][*archmCopula*] -
  the parameter of the Archimedean copula. A numerical value.
- **n** [rarchmCopula] -
  the number of random deviates to be generated, an integer value.
- **type** the type of the Archimedean copula. A character string ranging between "1" and "22". By default copula No. 1 will be chosen.
- **u, v** [rarchmCopula] -
  two numeric values or vectors of the same length at which the copula will be computed. If `u` is a list then the the \$x and \$y elements will be used as `u` and `v`. If `u` is a two column matrix then the first column will be used as `u` and the the second as `v`.
- **...** [archmCopulaFit] -
  arguments passed to the optimization function in use, `nlminb`.

Value

The function `pcopula` returns a numeric matrix of probabilities computed at grid positions xly.

The function `parchmCopula` returns a numeric matrix with values computed for the Archimedean copula.
The function `darchmCopula` returns a numeric matrix with values computed for the density of the Archemedean copula.

The functions `Phi*` return a numeric vector with the values computed from the Archemedean generator, its derivatives, or its inverse.

The functions `cK` and `cKInv` return a numeric vector with the values of the density and inverse for Archimedian copulae.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

---

**CopulaClass**

**Bivariate Copula Class**

**Description**

A collection and description of functions to specify the copula class and to investigate bivariate Frechet copulae.

The class representation and methods are:

- `fCOPULA` representation for an S4 object of class "fCOPULA".
- `show` S4 print method.

Frechet Copulae:

- `pfrechetCopula` computes Frechet copula probability.

**Usage**

```r
## S4 method for signature 'fCOPULA'
show(object)

pfrechetCopula(u = 0.5, v = u, type = c("m", "pi", "w"),
output = c("vector", "list"))
```

**Arguments**

- `object` [show] - an S4 object of class "fCOPULA".
- `output` [*frechetCopula] - output - a character string specifying how the output should be formatted. By default a vector of the same length as `u` and `v`. If specified as "list" then `u` and `v` are expected to span a two-dimensional grid as outputted by the function
grid2d and the function returns a list with elements $x$, $y$, and $z$ which can be directly used for example by 2D plotting functions.

**type**

[*frechetCopula*] -
the type of the Frechet copula. A character string selected from: "m", "pi", or "w".

**u, v**
two numeric values or vectors of the same length at which the copula will be computed. If $u$ is a list then the the $x$ and $y$ elements will be used as $u$ and $v$. If $u$ is a two column matrix then the first column will be used as $u$ and the the second as $v$.

**Details**

The function `pfrechetCopula` returns a numeric matrix of probabilities computed at grid positions $u$ | $v$. The arguments $u$ and $v$ are two single values or two numeric vectors of the same length. If $v$ is not specified then the same values are taken as for $u$. Alternatively, $u$ may be given as a two column vector or as a list with two entries as vectors. The first column or entry is taken as $u$ and the second as $v$.

**Value**

The print method `show` returns an S4 object of class "fCOPULA". The object contains the following slots:

- **@call** the function call.
- **@copula** the name of the copula.
- **@param** a list whose elements specify the model parameters of the copula.
- **@title** a character string with the name of the copula. This can be overwritten specifying a user defined input argument.
- **@description** a character string with an optional user defined description. By default just the current date will be returned.

The function `pfrechetCopula` returns a numeric vector of probabilities. An attribute named "control" is added which returns the name of the Frechet copula.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## fCOPULA -
getClass("fCOPULA")

## pfrechet -
# The Frechet Copula - m:
pfrechetCopula(0.5)
pfrechetCopula(0.25, 0.75)
pfrechetCopula(runif(5))
```
EllipticalCopulae

## grid2d

grid2d()

pFrechetCopula(grid2d())

---

### CopulaEnv

#### Bivariate Copula Environment

**Description**

Set and Get functions for the Copula environment.

### EllipticalCopulae

#### Bivariate Elliptical Copulae

**Description**

A collection and description of functions to investigate bivariate elliptical copulae.

**Elliptical Copulae Functions:**

- `rellipticalCopula` - Generates elliptical copula variates,
- `pellipticalCopula` - computes elliptical copula probability,
- `dellipticalCopula` - computes elliptical copula density,
- `rellipticalSlider` - displays interactive plots of variates,
- `pellipticalSlider` - displays interactive plots of probability,
- `dellipticalSlider` - displays interactive plots of density.

**Usage**

```r
rellipticalCopula(n, rho = 0.75, param = NULL, type = c("norm", "cauchy", "t"))
pellipticalCopula(u = 0.5, v = u, rho = 0.75, param = NULL, type = ellipticalList(), output = c("vector", "list"), border = TRUE)
dellipticalCopula(u = 0.5, v = u, rho = 0.75, param = NULL, type = ellipticalList(), output = c("vector", "list"), border = TRUE)
rellipticalSlider(B = 100)
pellipticalSlider(type = c("persp", "contour"), B = 20)
dellipticalSlider(type = c("persp", "contour"), B = 20)
```

**Arguments**

- `B` - [*Slider*] - the maximum slider menu value when the boundary value is infinite. By default
EllipticalCopulae

this is set to 10.

border

[ellipticalCopula][dellipticalCopula] -
a logical flag. If the argument u is an integer, say N, greater than one than all points on a square grid \([((0:N)/N)^2\) are computed. If border is FALSE than the border points are removed from the returned value, by default this is not the case.

n

[rellipticalCopula][ellipticalCopulaSim] -
the number of random deviates to be generated, an integer value.

output

[pellipticalCopula][dellipticalCopula] -
a character string specifying how the output should be formatted. By default a vector of the same length as u and v is returned. If specified as "list" then u and v are expected to span a two-dimensional grid as outputted by the function grid2d and the function returns a list with elements $x, y, and z which can be directly used for example by 2D plotting functions. For the grid version, when u is specified as an integer greater than one, always the output in form of a list will be returned.

rho

[*ellipticalCopula] -
is the numeric value setting the correlation strength, ranging between minus one and one.

param

[*ellipticalCopula][gfunc] -
additional distributional parameters: for the Student-t distribution this is "nu", for the Kotz distribution this is "r", and for the Exponential Power distribution these are "r" and "s". If the argument param=NULL then default values are taken. These are for the Student-t param=c(nu=4)), for the Kotz distribution param=c(r=1)), and for the exponential power distribution param=c(r=1, s=1). Note, that the Kotz and exponential power copulae are independent of r, and that r only enters the generator, the density, the probability and the quantile functions.

type

[*ellipticalCopula][gfunc] -
the type of the elliptical copula. A character string selected from: "norm", "cauchy", "t", "logistic", "laplace", "kotz", or "epower". [*ellipticalSlider] -
a character string which indicates what kind of plot should be displayed, either a perspective plot if type="persp", the default value, or a contour plot if type="contour".

u, v

[*ellipticalCopula] -
two numeric values or vectors of the same length at which the copula will be computed. If u is a list then the the $x and $y elements will be used as u and v. If u is a two column matrix then the first column will be used as u and the second as v. If u is an integer value greater than one, say N, than the values for all points on the \([(0:N)/N)^2\) grid spanning the unit square will be returned.

Value

Copula Functions:
EllipticalCopulae

The functions `rpellipticalCopula` return a numeric vector of random variates, probabilities, or densities for the specified copula computed at grid coordinates u,v.

The functions `rpellipticalSlider` display an interactive graph of an perspective copula plot either for random variates, probabilities or densities. Alternatively, an image underlayed contour plot can be shown.

**Copula Dependence Measures:**

The functions `ellipticalTau` and `ellipticalRho` return a numeric value for Kendall’s Tau and Spearman’s Rho.

**Copula Tail Coefficient:**

The function `ellipticalTailCoeff` returns the coefficient of tail dependence for a specified copula. The function `ellipticalTailPlot` displays a whole plot for the upper or alternatively for the lower tail dependence as a function of u for a set of nine rho values.

**Copula Generator Function:**

The function `gfunc` computes the generator function for the specified copula, by default the normal copula. If the argument x is missing, then the normalization constant lambda will be returned, otherwise if x is specified the values for the function g(x) will be returned. The selected type of copula is added to the output as an attribute named “control”. The function `gfuncSlider` allows to display interactively the generator function, the marginal density, the marginal probability, and the contours of the the bivariate density.

**Copula Simulation and Parameter Fitting:**

The function `ellipticalCopulaSim` returns a numeric two-column matrix with randomly generated variates for the specified copula.

The function `ellipticalCopulaFit` returns a fit to empirical data for the specified copula. The returned object is a list with elements from the function `nlminb`.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## [rp]ellipticalCopula
```
EllipticalDependency

---

EllipticalDependency  Bivariate Elliptical Copulae

Description

A collection and description of functions to investigate bivariate elliptical copulae.

Elliptical Copulae Functions:

- `ellipticalTau` Computes Kendall’s tau for elliptical copulae,
- `ellipticalRho` computes Spearman’s rho for elliptical copulae,
- `ellipticalTailCoeff` computes tail dependence for elliptical copulae,
- `ellipticalTailPlot` plots tail dependence for elliptical copulae.
Usage

ellipticalTau(rho)

ellipticalRho(rho, param = NULL, type = ellipticalList(), subdivisions = 500)

ellipticalTailCoeff(rho, param = NULL, type = c("norm", "cauchy", "t"))

ellipticalTailPlot(param = NULL, type = c("norm", "cauchy", "t"),
                   tail = c("Lower", "Upper"))

Arguments

rho  [*ellipticalCopula] -
is the numeric value setting the correlation strength, ranging between minus one
and one.

param [*ellipticalCopula][gfunc] -
additional distributional parameters: for the Student-t distribution this is "nu",
for the Kotz distribution this is "r", and for the Exponential Power distribution
these are "r" and "s". If the argument param=NULL then default values are
taken. These are for the Student-t param=c(nu=4), for the Kotz distribution
param=c(r=1)), and for the exponential power distribution param=c(r=1, s=1).
Note, that the Kotz and exponential power copulae are independent of r, and
that r only enters the generator, the density, the probability and the quantile
functions.

subdivisions [*ellipticalRho] -
an integer value with the number of subdivisions in each direction on the two
dimensional unit square to compute the mean value of Spearman’s Rho. By
default 500 subdivisions are used.

tail [*ellipticalTailPlot] -
a character string, either "Upper" or "Lower" denoting which of the two tails
should be displayed. By default the upper tail dependence will be considered.

type [*ellipticalCopula][gfunc] -
the type of the elliptical copula. A character string selected from: "norm",
"cauchy", "t", "logistic", "laplace", "kotz", or "epower". [*ellipticalSlider] -
a character string which indicates what kind of plot should be displayed, either
a perspective plot if type="persp", the default value, or a contour plot if
type="contour".

Value

Copula Functions:

The functions [rpde]ellipticalCopula return a numeric vector of random variates, probabilities,
or densities for the specified copula computed at grid coordinates ulv.

The functions [rpde]ellipticalSlider display an interactive graph of an perspective copula plot
either for random variates, probabilities or densities. Alternatively, an image underlayed contour
plot can be shown.

**Copula Dependence Measures:**

The functions `ellipticalTau` and `ellipticalRho` return a numeric value for Kendall’s Tau and Spearman’s Rho.

**Copula Tail Coefficient:**

The function `ellipticalTailCoeff` returns the coefficient of tail dependence for a specified copula. The function `ellipticalTailPlot` displays a whole plot for the upper or alternatively for the lower tail dependence as a function of $u$ for a set of nine $\rho$ values.

**Copula Generator Function:**

The function `gfunc` computes the generator function for the specified copula, by default the normal copula. If the argument $x$ is missing, then the normalization constant lambda will be returned, otherwise if $x$ is specified the values for the function $g(x)$ will be returned. The selected type of copula is added to the output as an attribute named “control”. The function `gfuncSlider` allows to display interactively the generator function, the marginal density, the marginal probability, and the contours of the bivariate density.

**Copula Simulation and Parameter Fitting:**

The function `ellipticalCopulaSim` returns a numeric two-column matrix with randomly generated variates for the specified copula.

The function `ellipticalCopulaFit` returns a fit to empirical data for the specified copula. The returned object is a list with elements from the function `nlminb`.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## [rp]ellipticalCopula -
# Default Normal Copula:
rellipticalCopula(10)
pellipticalCopula(10)

## [rp]ellipticalCopula -
# Student-t Copula Probability and Density:
```
EllipticalGenerator

Bivariate Elliptical Copulae

Description

A collection and description of functions concerned with the generator function for the elliptical copula and with functions for setting and checking the distributional parameters.

Functions:

- `ellipticalList()`: Returns list of implemented elliptical copulae.
- `ellipticalParam()`: Sets default parameters for an elliptical copula.
- `ellipticalRange()`: Returns the range of valid rho values.
- `ellipticalCheck()`: Checks if rho is in the valid range.
- `gfunc()`: Generator function for elliptical distributions.
- `gfuncSlider()`: Slider for generator, density and probability.

Usage

```r
ellipticalList()
ellipticalParam(type = ellipticalList())
ellipticalRange(type = ellipticalList())
```
EllipticalGenerator

ellipticalCheck(rho = 0.75, param = NULL, type = ellipticalList())
gfunc(x, param = NULL, type = ellipticalList())
gfuncSlider(B = 10)

Arguments

B [*Slider] -
the maximum slider menu value when the boundary value is infinite. By default
this is set to 10.

rho [*ellipticalCopula] -
is the numeric value setting the correlation strength, ranging between minus one
and one.

param [*ellipticalCopula][gfunc] -
additional distributional parameters: for the Student-t distribution this is "nu",
for the Kotz distribution this is "r", and for the Exponential Power distribu-
tion these are "r" and "s". If the argument param=NULL then default values are
taken. These are for the Student-t param=c(nu=4)), for the Kotz distribution
param=c(r=1)), and for the exponential power distribution param=c(r=1,s=1).
Note, that the Kotz and exponential power copulae are independent of r, and
that r only enters the generator, the density, the probability and the quantile
functions.

type [*ellipticalCopula][gfunc] -
the type of the elliptical copula. A character string selected from: "norm",
"cauchy", "t", "logistic", "laplace", "kotz", or "epower". [*ellipticalSlider] -
a character string which indicates what kind of plot should be displayed, ei-
ther a perspective plot if type="persp", the default value, or a contour plot if
type="contour".

x [*gfunc] -
a numeric value or vector out of the range [0,Inf) at which the generator will
be computed.

Value

Copula Functions:

The functions [rpd]ellipticalCopula return a numeric vector of random variates, probabilities,
or densities for the specified copula computed at grid coordinates ulv.

The functions [rpd]ellipticalSlider display an interactive graph of an perspective copula plot
either for random variates, probabilities or densities. Alternatively, an image underlayed contour
plot can be shown.

Copula Dependence Measures:
The functions `ellipticalTau` and `ellipticalRho` return a numeric value for Kendall’s Tau and Spearman’s Rho.

**Copula Tail Coefficient:**

The function `ellipticalTailCoeff` returns the coefficient of tail dependence for a specified copula. The function `ellipticalTailPlot` displays a whole plot for the upper or alternatively for the lower tail dependence as a function of \( u \) for a set of nine \( \rho \) values.

**Copula Generator Function:**

The function `gfunc` computes the generator function for the specified copula, by default the normal copula. If the argument \( x \) is missing, then the normalization constant lambda will be returned, otherwise if \( x \) is specified the values for the function \( g(x) \) will be returned. The selected type of copula is added to the output as an attribute named "control". The function `gfuncSlider` allows to display interactively the generator function, the marginal density, the marginal probability, and the contours of the bivariate density.

**Copula Simulation and Parameter Fitting:**

The function `ellipticalCopulaSim` returns a numeric two-column matrix with randomly generated variates for the specified copula.

The function `ellipticalCopulaFit` returns a fit to empirical data for the specified copula. The returned object is a list with elements from the function `nlminb`.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## ellipticalList -
# List implemented copulae:
ellipticalList()

## gfunc -
# Generator Function:
gfunc(x <- (0:10)/10, param = 2, type = "t")

## gfuncSlider -
# Try:
## Not run:
gfuncSlider()
## End(Not run)
```
Elliptical Modelling

**Description**

A collection and description of functions to investigate bivariate elliptical copulae.

Elliptical Copulae Functions:

- `ellipticalCopulaSim` simulates an elliptical copula.
- `ellipticalCopulaFit` fits the parameters of an elliptical copula.

**Usage**

```r
ellipticalCopulaSim(n, rho = 0.75, param = NULL, type = c("norm", "cauchy", "t"))
ellipticalCopulaFit(u, v, type = c("norm", "cauchy", "t"), ...)
```

**Arguments**

- `n` [ellipticalCopula][ellipticalCopulaSim] - the number of random deviates to be generated, an integer value.
- `rho` [*ellipticalCopula*] - is the numeric value setting the correlation strength, ranging between minus one and one.
- `param` [*ellipticalCopula][gfunc*] - additional distributional parameters: for the Student-t distribution this is "nu", for the Kotz distribution this is "r", and for the Exponential Power distribution these are "r" and "s". If the argument `param=NULL` then default values are taken. These are for the Student-t `param=c(nu=4)`, for the Kotz distribution `param=c(r=1)`, and for the exponential power distribution `param=c(r=1,s=1)`. Note, that the Kotz and exponential power copulae are independent of `r`, and that `r` only enters the generator, the density, the probability and the quantile functions.
- `type` [*ellipticalCopula][gfunc*] - the type of the elliptical copula. A character string selected from: "norm", "cauchy", "t", "logistic", "laplace", "kotz", or "epower". [*ellipticalSlider*] - a character string which indicates what kind of plot should be displayed, either a perspective plot if `type="persp"`, the default value, or a contour plot if `type="contour"`.
- `u, v` [*ellipticalCopula*] - two numeric values or vectors of the same length at which the copula will be computed. If `u` is a list then the the `x` and `y` elements will be used as `u` and `v`. If `u` is a two column matrix then the first column will be used as `u` and the second as `v`. If `u` is an integer value greater than one, say `N`, than the values for all points on the `[(0:N)/N]^2` grid spanning the unit square will be returned.
Value

Copula Functions:

The functions `ellipticalCopula` return a numeric vector of random variates, probabilities, or densities for the specified copula computed at grid coordinates $u,v$.

The functions `ellipticalSlider` display an interactive graph of a perspective copula plot either for random variates, probabilities or densities. Alternatively, an image underlaid contour plot can be shown.

Copula Dependence Measures:

The functions `ellipticalTau` and `ellipticalRho` return a numeric value for Kendall’s Tau and Spearman’s Rho.

Copula Tail Coefficient:

The function `ellipticalTailCoeff` returns the coefficient of tail dependence for a specified copula. The function `ellipticalTailPlot` displays a whole plot for the upper or alternatively for the lower tail dependence as a function of $u$ for a set of nine $\rho$ values.

Copula Generator Function:

The function `gfunc` computes the generator function for the specified copula, by default the normal copula. If the argument $x$ is missing, then the normalization constant lambda will be returned, otherwise if $x$ is specified the values for the function $g(x)$ will be returned. The selected type of copula is added to the output as an attribute named "control". The function `gfuncSlider` allows to display interactively the generator function, the marginal density, the marginal probability, and the contours of the bivariate density.

Copula Simulation and Parameter Fitting:

The function `ellipticalCopulaSim` returns a numeric two-column matrix with randomly generated variates for the specified copula.

The function `ellipticalCopulaFit` returns a fit to empirical data for the specified copula. The returned object is a list with elements from the function `nlminb`. 

... [ellipticalCopulaFit] - arguments passed to the optimization function `nlminb`. 
Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

## [rp]ellipticalCopula -
# Default Normal Copula:
rellipticalCopula(10)
pellipticalCopula(10)

## [rp]ellipticalCopula -
# Student-t Copula Probability and Density:
u = grid2d(x = (0:25)/25)
# CHECK ERROR
# pellipticalCopula(u, rho = 0.75, param = 4,
# type = "t", output = "list")
# CHECK ERROR DONE
d = dellipticalCopula(u, rho = 0.75, param = 4,
# type = "t", output = "list")
persp(d, theta = -40, phi = 30, col = "steelblue")

## ellipticalTau -
## ellipticalRho -
# Dependence Measures:
ellipticalTau(rho = -0.5)
ellipticalRho(rho = 0.75, type = "logistic", subdivisions = 100)

## ellipticalTailCoeff -
# Student-t Tail Coefficient:
ellipticalTailCoeff(rho = 0.25, param = 3, type = "t")

## gfunc -
# Generator Function:
plot(gfunc(x = 0:10), main = "Generator Function")

## ellipticalCopulaSim -
## ellipticalCopulaSim -
# Simulation and Parameter Fitting:
rv <- ellipticalCopulaSim(n = 100, rho = 0.75)
ellipticalCopulaFit(rv)

EmpiricalCopulae  Bivariate Empirical Copulae

Description

A collection and description of functions to investigate bivariate empirical copulae.

Empirical Copulae Functions:
Empirical Copulae

\texttt{pempiricalCopula} computes empirical copula probability,
\texttt{dempiricalCopula} computes empirical copula density.

Usage

\begin{verbatim}
pempiricalCopula(u, v, N = 10) dempiricalCopula(u, v, N = 10)
\end{verbatim}

Arguments

\begin{itemize}
\item \texttt{N} [\texttt{empiricalCopula}] - ... .
\item \texttt{u, v} [\texttt{*evCopula}][\texttt{*archmCopula}] -
  two numeric values or vectors of the same length at which the copula will be
  computed. If \texttt{u} is a list then the the \$x$ and \$y$ elements will be used as \texttt{u} and \texttt{v}.
  If \texttt{u} is a two column matrix then the first column will be used as \texttt{u} and the the
  second as \texttt{v}.
\end{itemize}

Value

The functions \texttt{*Spec} return an S4 object of class "\texttt{fCOPULA}". The object contains the following slots:

\begin{itemize}
\item \texttt{@call} the function call.
\item \texttt{@copula} the name of the copula.
\item \texttt{@param} a list whose elements specify the model parameters.
\item \texttt{@title} a character string with the name of the copula. This can be overwritten specifying a user defined input argument.
\item \texttt{@description} a character string with an optional user defined description. By default just the current date when the test was applied will be returned.
\end{itemize}

The function \texttt{pcopula} returns a numeric matrix of probabilities computed at grid positions $x|y$.

The function \texttt{parchmCopula} returns a numeric matrix with values computed for the Archemdean copula.

The function \texttt{darchmCopula} returns a numeric matrix with values computed for the density of the Archemedian copula.

The functions \texttt{Phi*} return a numeric vector with the values computed from the Archemedian generator, its derivatives, or its inverse.

The functions \texttt{cK} and \texttt{cKInv} return a numeric vector with the values of the density and inverse for Archimedian copulae.
**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

---

**ExtremeValueCopulae  Bivariate Extreme Value Copulae**

**Description**

A collection and description of functions to investigate bivariate extreme value copulae.

Extreme Value Copulae Functions:

- `revCopula` generates extreme value copula random variates,
- `pevCopula` computes extreme value copula probability,
- `devCopula` computes extreme value copula density,
- `revSlider` displays interactive plots of extreme value random variates,
- `pevSlider` displays interactive plots of extreme value probability,
- `devSlider` displays interactive plots of extreme value density.

**Usage**

```r
revCopula(n, param = NULL, type = evList())
pevCopula(u = 0.5, v = u, param = NULL, type = evList(),
          output = c("vector", "list"), alternative = FALSE)
devCopula(u = 0.5, v = u, param = NULL, type = evList(),
          output = c("vector", "list"), alternative = FALSE)
revSlider(B = 10)
pevSlider(type = c("persp", "contour"), B = 10)
devSlider(type = c("persp", "contour"), B = 10)
```

**Arguments**

- `alternative` [evRho][evTau][*evCopula]
  - Should the probability be computed alternatively in a direct way from the probability formula or by default via the dependency function?
- `B` [*Slider]
  - the maximum slider menu value when the boundary value is infinite. By default this is set to 10.
- `n` [revCopula][evCopulaSim]
  - the number of random deviates to be generated, an integer value.
- `output` [*evCopula]
  - output - a character string specifying how the output should be formatted. By default a vector of the same length as u and v. If specified as "list" then u and v are expected to span a two-dimensional grid as outputted by the function
grid2d and the function returns a list with elements $x$, $y$, and $z$ which can be directly used for example by 2D plotting functions.

**param**

[*ev*][A*] - distribution and copulae parameters. A numeric value or vector of named parameters as required by the copula specified by the variable type. If set to NULL, then the default parameters will be taken.

**type**

[*ev*][Afunc] - the type of the extreme value copula. A character string selected from: "gumbel", "galambos", "husler.reiss", "tawn", or "bb5":

[evSlider] - a character string specifying the plot type. Either a perspective plot which is the default or a contour plot with an underlying image plot will be created.

**u, v**

[*evCopula][*archmCopula] - two numeric values or vectors of the same length at which the copula will be computed. If $u$ is a list then the $x$ and $y$ elements will be used as $u$ and $v$. If $u$ is a two column matrix then the first column will be used as $u$ and the the second as $v$.

**Value**

The function `pcopula` returns a numeric matrix of probabilities computed at grid positions $x_l y$.

The function `parchmCopula` returns a numeric matrix with values computed for the Archimedean copula.

The function `darchmCopula` returns a numeric matrix with values computed for the density of the Archimedean copula.

The functions `Phi*` return a numeric vector with the values computed from the Archimedean generator, its derivatives, or its inverse.

The functions `cK` and `cKInv` return a numeric vector with the values of the density and inverse for Archimedian copulae.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## fCOPULA:
getClass("fCOPULA")
getSlots("fCOPULA")

## revCopula:
# Not yet implemented
# revCopula(n = 10, type = "galambos")```
## pevCopula -
pevCopula(u = grid2d(), type = "galambos", output = "list")

## devCopula -
devCopula(u = grid2d(), type = "galambos", output = "list")

## AfuncSlider -
# Generator, try:
## Not run: AfuncSlider()

---

**ExtremeValueDependency**

*Bivariate Extreme Value Copulae*

### Description
A collection and description of functions to investigate bivariate extreme value copulae.

**Extreme Value Copulae Functions:**

- `evTau` computes Kendall’s tau for extreme value copulae,
- `evRho` computes Spearman’s rho for extreme value copulae,
- `evTailCoeff` computes tail dependence for extreme value copulae,
- `evTailCoeffSlider` plots tail dependence for extreme value copulae.

### Usage

```r
evTau(param = NULL, type = evList(), alternative = FALSE)
evRho(param = NULL, type = evList(), alternative = FALSE)
evTailCoeff(param = NULL, type = evList())
evTailCoeffSlider(B = 10)
```

### Arguments

- `alternative` [evRho][evTau][*evCopula] - Should the probability be computed alternatively in a direct way from the probability formula or by default via the dependency function?
- `B` [*Slider] - the maximum slider menu value when the boundary value is infinite. By default this is set to 10.
- `param` [*ev*][A*] - distribution and copulae parameters. A numeric value or vector of named parameters as required by the copula specified by the variable `type`. If set to NULL, then the default parameters will be taken.
type  [*ev*][Afunc] -
the type of the extreme value copula. A character string selected from: "gumbel", "galambos", "husler.reiss", "tawn", or "bb5".
[evSlider] -
a character string specifying the plot type. Either a perspective plot which is the default or a contour plot with an underlying image plot will be created.

Value

The function pcopula returns a numeric matrix of probabilities computed at grid positions xly.

The function parchmCopula returns a numeric matrix with values computed for the Archemedean copula.

The function darchmCopula returns a numeric matrix with values computed for the density of the Archemedean copula.

The functions Phi* return a numeric vector with the values computed from the Archemedean generator, its derivatives, or its inverse.

The functions cK and cKInv return a numeric vector with the values of the density and inverse for Archimedean copulae.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## fCOPULA -
getClass("fCOPULA")
getSlots("fCOPULA")

## revCopula -
# Not yet implemented
# revCopula(n = 10, type = "galambos")

## pevCopula -
pevCopula(u = grid2d(), type = "galambos", output = "list")

## devCopula -
devCopula(u = grid2d(), type = "galambos", output = "list")

## AfuncSlider -
# Generator, try:
## Not run: AfuncSlider()
```
**Description**

A collection and description of functions concerned with the generator function for the extreme value copula and with functions for setting and checking the distributional parameters.

Functions:

- `evList` returns list of implemented extreme value copulae,
- `evParam` sets default parameters for an extreme value copula,
- `evRange` returns the range of valid rho values,
- `evCheck` checks if rho is in the valid range,
- `Afunc` computes dependence function,
- `AfuncSlider` displays interactively dependence function.

**Usage**

```r
evList()
evParam(type = evList())
evRange(type = evList())
evCheck(param, type = evList())
Afunc(x, param = NULL, type = evList())
AfuncSlider()
```

**Arguments**

- `param` distribution and copulae parameters. A numeric value or vector of named parameters as required by the copula specified by the variable `type`. If set to `NULL`, then the default parameters will be taken.
- `type` the type of the extreme value copula. A character string selected from: "gumbel", "galambos", "husler.reiss", "tawn", or "bb5".
- `x` a numeric value or vector ranging between zero and one.

**Value**

The function `pcopula` returns a numeric matrix of probabilities computed at grid positions xly.

The function `parchmCopula` returns a numeric matrix with values computed for the Archemedean copula.

The function `darchmCopula` returns a numeric matrix with values computed for the density of the Archemedean copula.
The functions $\Phi^*$ return a numeric vector with the values computed from the Archimedean generator, its derivatives, or its inverse.

The functions $c_K$ and $c_{Kinv}$ return a numeric vector with the values of the density and inverse for Archimedean copulae.

**Author(s)**

Diethelm Wuertz for the Rmetrics R-port.

**Examples**

```r
## fCOPULA -
getClass("fCOPULA")
getSlots("fCOPULA")

## revCopula -
# Not yet implemented
# revCopula(n = 10, type = "galambos")

## pevCopula -
pevCopula(u = grid2d(), type = "galambos", output = "list")

## devCopula -
devCopula(u = grid2d(), type = "galambos", output = "list")

## AfuncSlider -
# Generator, try:
## Not run: AfuncSlider()
```

**Description**

A collection and description of functions to investigate bivariate extreme value copulae.

Extreme Value Copulae Functions:

- `evCopulaSim` simulates an extreme value copula,
- `evCopulaFit` fits the parameters of an extreme value copula.

**Usage**

```r
evCopulaSim(n, param = NULL, type = evList())
evCopulaFit(u, v = NULL, type = evList(), ...)
```
Arguments

**n**

[revCopula][evCopulaSim] -
the number of random deviates to be generated, an integer value.

**param**

[*ev*][A*] -
distribution and copulae parameters. A numeric value or vector of named parameters as required by the copula specified by the variable *type*. If set to NULL, then the default parameters will be taken.

**type**

[*ev*][Afunc] -
the type of the extreme value copula. A character string selected from: "gumbel", "galambos", "husler.reiss", "tawn", or "bb5".

[evSlider] -
a character string specifying the plot type. Either a perspective plot which is the default or a contour plot with an underlying image plot will be created.

**u, v**

[*evCopula][archmCopula] -
two numeric values or vectors of the same length at which the copula will be computed. If *u* is a list then the the $x$ and $y$ elements will be used as *u* and *v*. If *u* is a two column matrix then the first column will be used as *u* and the the second as *v*.

[evCopulaFit] -
arguments passed to the optimization function nlmnbl.

Value

The function pcopula returns a numeric matrix of probabilities computed at grid positions $x | y$.

The function parchmCopula returns a numeric matrix with values computed for the Archemedean copula.

The function darchmCopula returns a numeric matrix with values computed for the density of the Archemedean copula.

The functions Phi* return a numeric vector with the values computed from the Archemedean generator, its derivatives, or its inverse.

The functions cK and cKInv return a numeric vector with the values of the density and inverse for Archimedian copulae.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```r
## fCOPULA -
getClass("fCOPULA")
getSlots("fCOPULA")
```
## revCopula -
# Not yet implemented
# revCopula(n = 10, type = "galambos")

## pevCopula -
pevCopula(u = grid2d(), type = "galambos", output = "list")

## devCopula -
devCopula(u = grid2d(), type = "galambos", output = "list")

## AfuncSlider -
# Generator, try:
## Not run: AfuncSlider()
Index

* models
  ArchimedeanCopulae, 6
  ArchimedeanDependency, 8
  ArchimedeanGenerator, 9
  ArchimedeanModelling, 11
  CopulaClass, 12
  CopulaEnv, 14
  EllipticalCopulae, 14
  EllipticalDependency, 17
  EllipticalGenerator, 20
  EllipticalModelling, 23
  EmpiricalCopulae, 25
  ExtremeValueCopulae, 27
  ExtremeValueDependency, 29
  ExtremeValueGenerator, 31
  ExtremeValueModelling, 32

* package
  fCopulae-package, 2
  Afunc (ExtremeValueGenerator), 31
  AfuncSlider (ExtremeValueGenerator), 31
  ArchimedeanCopulae, 6
  ArchimedeanDependency, 8
  ArchimedeanGenerator, 9
  ArchimedeanModelling, 11
  archmCheck (ArchimedeanGenerator), 9
  archmCopulaFit (ArchimedeanModelling), 11
  archmCopulaSim (ArchimedeanModelling), 11
  archmList (ArchimedeanGenerator), 9
  archmParam (ArchimedeanGenerator), 9
  archmRange (ArchimedeanGenerator), 9
  archmRho (ArchimedeanDependency), 8
  archmTailCoeff (ArchimedeanDependency), 8
  archmTailPlot (ArchimedeanDependency), 8
  archmTau (ArchimedeanDependency), 8
  CopulaClass, 12
  CopulaEnv, 14
  darchmCopula (ArchimedeanCopulae), 6
  darchmSlider (ArchimedeanCopulae), 6
  dellipticalCopula (EllipticalCopulae), 14
  dellipticalSlider (EllipticalCopulae), 14
  dempricalCopula (EmpiricalCopulae), 25
  devCopula (ExtremeValueCopulae), 27
  devSlider (ExtremeValueCopulae), 27
dgumelCopula (ArchimedeanCopulae), 6
  ellipticalCheck (EllipticalGenerator), 20
  EllipticalCopulae, 14
  ellipticalCopulaFit (EllipticalModelling), 23
  ellipticalCopulaSim (EllipticalModelling), 23
  EllipticalDependency, 17
  EllipticalGenerator, 20
  ellipticalList (EllipticalGenerator), 20
  EllipticalModelling, 23
  ellipticalParam (EllipticalGenerator), 20
  ellipticalRange (EllipticalGenerator), 20
  ellipticalRho (EllipticalDependency), 17
  ellipticalTailCoeff (EllipticalDependency), 17
  ellipticalTailPlot (EllipticalDependency), 17
  ellipticalTau (EllipticalDependency), 17
  EmpiricalCopulae, 25
evCheck (ExtremeValueGenerator), 31
  evCopulaFit (ExtremeValueModelling), 32
  evCopulaSim (ExtremeValueModelling), 32
  evList (ExtremeValueGenerator), 31
evParam (ExtremeValueGenerator), 31
evRange (ExtremeValueGenerator), 31
evRho (ExtremeValueDependency), 29
evTailCoeff (ExtremeValueDependency), 29
evTailCoeffSlider
ExtremeValueCopulae, 27
ExtremeValueDependency, 29
ExtremeValueGenerator, 31
ExtremeValueModelling, 32
fCOPULA (CopulaClass), 12
fCOPULA-class (CopulaClass), 12
fCopulae (fCopulae-package), 2
fCopulae-package, 2
gfunc (EllipticalGenerator), 20
gfuncSlider (EllipticalGenerator), 20
Kfunc (ArchimedeanGenerator), 9
KfuncSlider (ArchimedeanGenerator), 9
parchmCopula (ArchimedeanCopulae), 6
parchmSlider (ArchimedeanCopulae), 6
pellipticalCopula (EllipticalCopulae), 14
pellipticalSlider (EllipticalCopulae), 14
pempiricalCopula (EmpiricalCopulae), 25
pevCopula (ExtremeValueCopulae), 27
pevSlider (ExtremeValueCopulae), 27
pfrechetCopula (CopulaClass), 12
pgumbelCopula (ArchimedeanCopulae), 6
Phi (ArchimedeanGenerator), 9
PhiSlider (ArchimedeanGenerator), 9
rarchmCopula (ArchimedeanCopulae), 6
rarchmSlider (ArchimedeanCopulae), 6
rellipticalCopula (EllipticalCopulae), 14
rellipticalSlider (EllipticalCopulae), 14
revCopula (ExtremeValueCopulae), 27
revSlider (ExtremeValueCopulae), 27
rgumbelCopula (ArchimedeanCopulae), 6
show, fCOPULA-method (CopulaClass), 12