Package 'tscopula'

July 7, 2021

Type Package

Title Time Series Copula Models

Version 0.2.1

Date 2021-07-03

Maintainer Alexander McNeil <alexanderjmcneil@gmail.com>

Description Functions for the analysis of time series using copula models.
The package is based on methodology described in the following references.
McNeil, A.J. (2021) <doi:10.3390/risks9010014>,
Bladt, M., & McNeil, A.J. (2020) <arXiv:2006.11088>.

Depends R (>= 3.5.0)

License GPL-3

LazyData true

RoxygenNote 7.1.1

Encoding UTF-8

Imports methods, stats, graphics, utils, zoo, xts, FKF, ltsa,
rvinecopulib, FitAR, arfima, Matrix

Collate 'basic_objects.R' 'armacopula.R' 'dvinecopula.R'
'dvinecopula2.R' 'fitting_basic.R' 'margins.R' 'full_models.R'
'vtransforms.R' 'fitting_vtscopula.R' 'helper_vtarma.R'
'data.R'

Suggests knitr, rmarkdown

VignetteBuilder knitr

NeedsCompilation no

Author Alexander McNeil [aut, cre],
Martin Bladt [aut]

Repository CRAN

Date/Publication 2021-07-07 08:20:02 UTC

R topics documented:

`acf2pacf` *Compute partial autocorrelations from autocorrelations*

Description

Compute partial autocorrelations from autocorrelations

Usage

```
acf2pacf(rho)
```

Arguments

`rho`	vector of autocorrelation values (excluding 1).

Value

A vector of partial autocorrelation values with same length as `rho`.

Examples

```
rho <- ARMAacf(ar = -0.9, ma = 0.8, lag.max = 50)[-1]
alpha <- acf2pacf(rho)
```

---

armacopula *Constructor function for ARMA copula process*

Description

Constructor function for ARMA copula process

Usage

```
armacopula(pars = list(ar = 0, ma = 0))
```

Arguments

pars	list consisting of vector of AR parameters named ‘ar‘ and vector of MA parameters named ‘ma‘.

Value

An object of class armacopula.

Examples

```
armacopula(list(ar = 0.5, ma = 0.4))
```

---

armacopula-class *ARMA copula processes*

Description

Class of objects for ARMA copula processes.

Usage

```
## S4 method for signature 'armacopula'
coef(object)

## S4 method for signature 'armacopula'
show(object)

## S4 method for signature 'armacopula'
sim(object, n = 1000)

## S4 method for signature 'armacopula'
kendall(object, lagmax = 20)
```

Arguments

object	an object of the class.
n	length of realization.
lagmax	maximum value of lag.

Methods (by generic)

- coef: Coef method for ARMA copula class
- show: Show method for ARMA copula process
- sim: Simulation method for armacopula class
- kendall: Calculate Kendall's tau values for armacopula model

Slots

name name of ARMA copula process.

modelspec vector containing number of AR and MA parameters.

pars list consisting of vector of AR parameters named 'ar' and vector of MA parameters named 'ma'.

Examples

```
sim(armacopula(list(ar = c(0.5, 0.4), ma = -0.8)), n = 1000)
mod <- armacopula(list(ar = 0.95, ma = -0.85))
kendall(mod)
```

---

bitcoin *Bitcoin price data 2016-19*

Description

Time series of Bitcoin closing prices from 31 December 2015 to 31 December 2019 (1044 values). This permits the calculation of 4 calendar years of returns.

Usage

```
data(bitcoin)
```

Format

An object of class "xts".

Examples

```
data(bitcoin)
plot(bitcoin)
X <- (diff(log(bitcoin))[-1]) * 100
plot(X)
```

`coerce,tscopula,tscm-method`

Convert tscopula object to tscm object

Description

Convert tscopula object to tscm object

Usage

```
## S4 method for signature 'tscopula,tscm'
coerce(from, to = "tsc", strict = TRUE)
```

Arguments

`from`	a tscopula object.
`to`	a tscm object.
`strict`	logical variable stating whether strict coercion should be enforced.

Value

A tscm object.

`coerce,tscopulafit,tscmfit-method`

Convert tscopulafit object to be tscmfit object

Description

Convert tscopulafit object to be tscmfit object

Usage

```
## S4 method for signature 'tscopulafit,tscmfit'
coerce(from, to = "tscmfit", strict = TRUE)
```

Arguments

`from`	a tscopulafit object.
`to`	a tscmfit object.
`strict`	logical variable stating whether strict coercion should be enforced.

Value

A tscmfit object.

cpi *CPI inflation data 1959-2020*

Description

Time series of US quarterly CPI (consumer price index) data Q4 1959 to Q4 2020 (245 values) for studying inflation. These data were sourced from the OECD webpage and represent the total 'perspective' on inflation, including food and energy. They have been based to have a value of 100 in 2015.

Usage

```
data(cpi)
```

Format

An object of class `"xts"`.

Examples

```
data(cpi)
plot(cpi)
X <- (diff(log(cpi))[-1]) * 100
plot(X)
```

dmarg *Compute density of marginal model*

Description

Compute the density function of the marginal model.

Usage

```
dmarg(x, y, log = FALSE)
```

Arguments

x	an object of class margin.
y	vector of values for which density should be computed.
log	logical variable specifying whether log density should be returned.

Value

A vector of values for the density.

Examples

```
margmod <- margin("norm", pars = c(mean = 0, sd = 1))
dmarg(margmod, c(-2, 0, 2), log = TRUE)
```

`doubleweibull` *Double Weibull distribution*

Description

Double Weibull distribution

Usage

```
ddoubleweibull(x, mu = 0.05, shape = 1, scale = 1, log = FALSE)

pdoubleweibull(q, mu = 0.05, shape = 1, scale = 1)

qdoubleweibull(p, mu = 0.05, shape = 1, scale = 1)

rdoubleweibull(n, mu = 0.05, shape = 1, scale = 1)
```

Arguments

`x`	vector of values.
`mu`	location parameter.
`shape`	shape parameter.
`scale`	scale parameter.
`log`	flag for log density.
`q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations.

Value

A vector of density, distribution function, quantile or random values.

dvinecopula *Constructor function for dvinecopula process*

Description

Constructor function for dvinecopula process

Usage

```
dvinecopula(family = "indep", pars = list(NULL), rotation = 0)
```

Arguments

family	a vector of family names
pars	a list containing the parameters of each lag
rotation	a vector of rotations

Value

An object of class dvinecopula.

Examples

```
dvinecopula(family = c("joe", "gauss", "t"), pars = list(3, .5, c(1, 2)), rotation = c(180, 0, 0))
```

dvinecopula-class *D-vine copula processes*

Description

Class of objects for d-vine copula processes.

Usage

```
## S4 method for signature 'dvinecopula'
coef(object)

## S4 method for signature 'dvinecopula'
show(object)

## S4 method for signature 'dvinecopula'
sim(object, n = 1000, innov = NA, start = NA)

## S4 method for signature 'dvinecopula'
kendall(object, lagmax = 20)
```

Arguments

object	an object of the class.
n	length of realization.
innov	vector of innovations of length n.
start	vector of start values with length equal to order of process.
lagmax	maximum value of lag.

Methods (by generic)

- coef: Coef method for dvinecopula class
- show: Show method for dvinecopula class
- sim: Simulation method for dvinecopula class
- kendall: Calculate Kendall's tau values for pair copulas in d-vine copula

Slots

name name of the d-vine copula process.

modelspec list containing the family, number of parameters and rotations

pars list comprising of the parameters.

Examples

```
sim(dvinecopula("gauss", 0.5))
mixmod <- dvinecopula(family = c("gumbel", "gauss"), pars = list(1.5, -0.6))
kendall(mixmod)
```

---

dvinecopula2 *Constructor function for dvinecopula2 process*

Description

Constructor function for dvinecopula2 process

Usage

```
dvinecopula2(
  family = "gauss",
  rotation = 0,
  kpacf = "kpacf_arma",
  pars = list(ar = 0.1, ma = 0.1),
  maxlag = Inf,
  negtau = "none"
)
```

Arguments

family	family name
rotation	rotation
kpacf	character string giving name of Kendal pacf
pars	a list containing the parameters of each lag
maxlag	scalar specifying maximum lag
negtau	character specifiying treatment of negative tau values

Value

An object of class dvinecopula2.

Examples

```
dvinecopula2(family = "joe", kpacf = "kpacf_arma",
pars = list(ar = 0.95, ma = -0.85), maxlag = 30)
```

---

dvinecopula2-class *D-vine copula processes of type 2*

Description

Class of objects for d-vine copula processes.

Usage

```
## S4 method for signature 'dvinecopula2'
coef(object)

## S4 method for signature 'dvinecopula2'
show(object)

## S4 method for signature 'dvinecopula2'
sim(object, n = 1000)

## S4 method for signature 'dvinecopula2'
kendall(object, lagmax = 20)
```

Arguments

object	an object of the class.
n	length of realization.
lagmax	maximum value of lag.

Methods (by generic)

- `coef`: Coef Method for dvinecopula2 class
- `show`: Show method for dvinecopula2 class
- `sim`: Simulation method for dvinecopula2 class
- `kendall`: Calculate Kendall's tau values for pair copulas in type 2 d-vine copula

Slots

`name` name of the d-vine copula process.

`modelspec` list containing the family, rotation, and name of KPACF

`pars` list comprising of the parameters.

Examples

```
copmod <- dvinecopula2(family = "joe", kpacf = "kpacf_arma",
pars = list(ar = 0.95, ma = -0.85), maxlag = 30)
kendall(copmod)
```

`fit` *Generic for estimating time series models*

Description

Methods are available for objects of class tscopulaU, vtscopula, tscopulafit, margin and tscm.

Usage

```
fit(x, y, ...)
```

Arguments

`x`	an object of the model class.
`y`	a vector or time series of data.
`...`	further arguments to be passed on.

Value

An object of the fitted model class.

fit,margin-method *Fit method for margin class*

Description

Fit method for margin class

Usage

```
## S4 method for signature 'margin'
fit(x, y, tsoptions = list(), control = list(maxit = 1000))
```

Arguments

x	an object of class margin.
y	a vector or time series of data.
tsoptions	list of optional arguments: hessian is logical variable specifying whether Hessian matrix should be returned; start is vector od named starting values
control	list of control parameters to be passed to the optim function.

Value

An object of class marginfit.

Examples

```
margmod <- margin("norm", pars = c(mean = 0, sd = 1))
data <- sim(margmod, n = 500)
fit(margmod, data)
```

fit,tscm-method *Fit method for tscm class*

Description

Fit method for tscm class

Usage

```
## S4 method for signature 'tscm'
fit(
  x,
  y,
  tsoptions = list(),
  control = list(warn.1d.NelderMead = FALSE, trace = FALSE, maxit = 5000),
  method = "IFM"
)
```

Arguments

x	an object of class tscm.
y	a vector or time series of data.
tsoptions	a list of parameters passed to fitting.
control	list of control parameters to be passed to the optim function.
method	character string specifying method.

Value

An object of class tscmfit.

Examples

```
mod <- tscm(dvinecopula(family = "gauss", pars = 0.5), margin("doubleweibull"))
y <- sim(mod)
fit(mod, y)
```

---

fit,tscopulafit-method

Fit method for tscopulafit class

Description

Fit method for tscopulafit class

Usage

```
## S4 method for signature 'tscopulafit'
fit(x, y, tsoptions = list(), control = list(warn.1d.NelderMead = FALSE))
```

Arguments

x	an object of class tscopulafit.
y	vector or time series of data to which the copula process is to be fitted.
tsoptions	list of options
control	list of control parameters to be passed to the optim function.

Value

An object of class tscopulafit.

Examples

```
ar1 <- armacopula(list(ar = 0.7))
data <- sim(ar1, 1000)
ar1fit <- fit(fit(ar1, data), sim(ar1, 1000))
```

fit,tscopulaU-method *Fit method for tscopulaU class*

Description

Fit method for tscopulaU class

Usage

```
## S4 method for signature 'tscopulaU'
fit(x, y, tsoptions = list(), control = list(warn.1d.NelderMead = FALSE))
```

Arguments

x	an object of class tscopulaU.
y	vector or time series of data to which the copula process is to be fitted.
tsoptions	list of options
control	list of control parameters to be passed to the optim function.

Value

An object of class tscopulafit.

Examples

```
data <- sim(armacopula(list(ar = 0.5, ma = 0.4)), n = 1000)
fit(armacopula(list(ar = 0.5, ma = 0.4)), data)
```

fit,vtscopula-method *Fit method for vtscopula class*

Description

Fit object of class vtscopula to data using maximum likelihood.

Usage

```
## S4 method for signature 'vtscopula'
fit(
  x,
  y,
  tsoptions = list(),
  control = list(maxit = 2000, warn.1d.NelderMead = FALSE)
)
```

Arguments

x	an object of class vtscopula.
y	a vector or time series of data.
tsoptions	list of optional arguments: hessian is logical variable specifying whether Hessian matrix should be returned; method is choice of optimization method.
control	list of control parameters to be passed to the optim function.

Value

An object of class tscopulafit.

Examples

```
copobject <- armacopula(pars = list(ar = 0.6, ma = 0.2))
vtcop <- vtscopula(copobject, Vtransform = V2p())
y <- sim(vtcop)
fit(vtcop, y)
```

---

glag *Generalized lagging function*

Description

Generalized lagging function

Usage

```
glag(x, lagmax = 20, glagplot = FALSE)
```

Arguments

x	an object of class tscopulafit.
lagmax	maximum value for lag.
glagplot	logical value indicating generalized lag plot.

Value

If glagplot is TRUE a list of generalized lagged datasets of maximum length 9 is returned to facilitate a generalized lagplot. If glagplot is FALSE a vector of length lagmax containing the Kendall rank correlations for the generalized lagged datasets is returned.

kendall *Generic for Kendall correlations*

Description

Methods are available for objects of class armacopula, dvinecopula, dvinecopula2 and vtscopula.

Usage

```
kendall(object, ...)
```

Arguments

object	an object of the model class.
...	further arguments to be passed to Kendall calculation.

Value

A vector of Kendall correlations.

kfilter *Kalman filter for ARMA copula model*

Description

Kalman filter for ARMA copula model

Usage

```
kfilter(x, y)
```

Arguments

x	an object of class armacopula.
y	a vector of data.

Value

A matrix or multivariate time series with columns consisting of conditional mean, standard deviation and residuals.

Examples

```
data <- sim(armacopula(list(ar = c(0.5, 0.4), ma = -0.8)), n = 1000)
kfilter(armacopula(list(ar = c(0.5, 0.4), ma = -0.8)), data)
```

kpacf_arfima *KPACF of ARFIMA process*

Description

KPACF of ARFIMA process

Usage

```
kpacf_arfima(k, theta)
```

Arguments

k	number of lags.
theta	list with components ar, ma and d specifying the ARFIMA parameters

Value

A vector of Kendall partial autocorrelations of length k.

kpacf_arma *KPACF of ARMA process*

Description

KPACF of ARMA process

Usage

```
kpacf_arma(k, theta)
```

Arguments

k	number of lags.
theta	list with components ar and ma specifying the ARMA parameters.

Value

A vector of Kendall partial autocorrelations of length k.

kpacf_fbn *KPACF of fractional Brownian noise*

Description

KPACF of fractional Brownian noise

Usage

```
kpacf_fbn(k, theta)
```

Arguments

k	number of lags
theta	parameter of process

Value

A vector of Kendall partial autocorrelations of length k.

laplace *Laplace distribution*

Description

Laplace distribution

Usage

```
dlaplace(x, mu = 0.05, scale = 1, log = FALSE)

plaplace(q, mu = 0.05, scale = 1)

qlaplace(p, mu = 0.05, scale = 1)

rlaplace(n, mu = 0.05, scale = 1)
```

Arguments

x	vector of values.
mu	location parameter.
scale	scale parameter.
log	flag for log density.
q	vector of quantiles.
p	vector of probabilities.
n	number of observations.

Value

A vector of density, distribution function, quantile or random values.

`margin` *Constructor function for margin*

Description

Constructor function for margin

Usage

```
margin(name, pars = NULL)
```

Arguments

`name`	character string giving name of distribution
`pars`	parameters of the distribution

Value

An object of class margin.

Examples

```
margin("sst")
```

`margin-class` *Marginal model for time series*

Description

Class of objects for marginal models for stationary time series. The object is given a name and there must exist functions pname, qname, dname and rname. For example, the object could be named norm and make use of `pnorm`, `qnorm`, `dnorm` and `rnorm`. As well as the parameters of the distribution, dname must have the logical argument log specifying whether log density should be computed.

Usage

```
## S4 method for signature 'margin'
coef(object)

## S4 method for signature 'margin'
sim(object, n = 1000)

## S4 method for signature 'margin'
show(object)
```

Arguments

object	an object of the class.
n	length of realization.

Methods (by generic)

- coef: Coef method for margin class
- sim: Simulation method for margin class
- show: Show method for margin class

Slots

name name of the marginal model class.

pars a numeric vector containing the named parameters of the distribution which are passed as arguments to pname, qname, dname and rname.

Examples

```
new("margin", name = "norm", pars = c(mu = 0, sigma = 1))
margmod <- margin("norm", pars = c(mean = 0, sd = 1))
sim(margmod, n = 500)
```

---

marginfit-class *Fitted marginal model for time series*

Description

Fitted marginal model for time series

Usage

```
## S4 method for signature 'marginfit'
logLik(object)
```

Arguments

object	an object of the class.

Methods (by generic)

- logLik: logLik method for marginfit class

Slots

margin an object of class margin.

data numeric vector or time series of data.

fit a list containing details of the maximum likelihood fit.

non_invert *Check for invertibility of ARMA process*

Description

Check for invertibility of ARMA process

Usage

```
non_invert(ma)
```

Arguments

ma	vector of moving average parameters.

Value

A logical variable stating whether ARMA process is invertible.

non_stat *Check for causality of ARMA process*

Description

Check for causality of ARMA process

Usage

```
non_stat(ar)
```

Arguments

ar	vector of autoregressive parameters

Value

A logical variable stating whether ARMA process is causal.

pacf2acf *Compute autocorrelations from partial autocorrelations*

Description

Compute autocorrelations from partial autocorrelations

Usage

```
pacf2acf(alpha)
```

Arguments

alpha	vector of partial autocorrelation values.

Value

A vector of autocorrelation values with same length as alpha.

Examples

```
alpha <- ARMAacf(ar = -0.9, ma = 0.8, lag.max = 50, pacf = TRUE)
rho <- pacf2acf(alpha)
```

pcoincide *Compute coincidence probability for v-transform*

Description

Computes the probability that if we v-transform a uniform random variable and then stochastically invert the v-transform, we get back to the original value.

Usage

```
pcoincide(x)
```

Arguments

x	an object of class Vtransform.

Value

The probability of coincidence.

Examples

```
pcoincide(Vlinear(delta = 0.4))
pcoincide(V3p(delta = 0.45, kappa = 0.5, xi = 1.3))
```

`plot,marginfit,missing-method`

Plot method for marginfit class

Description

Plot method for marginfit class

Usage

```
## S4 method for signature 'marginfit,missing'
plot(x, bw = FALSE)
```

Arguments

`x`	an object of class marginfit.
`bw`	logical variable specifying whether black-white options should be chosen.

Value

No return value, generates plot.

`plot,tscmfit,missing-method`

Plot method for tscmfit class

Description

Plot method for tscmfit class

Usage

```
## S4 method for signature 'tscmfit,missing'
plot(x, plottype = "residual", bw = FALSE, lagmax = 30)
```

Arguments

`x`	an object of class tscmfit.
`plottype`	type of plot required.
`bw`	logical variable specifying whether black-white options should be chosen.
`lagmax`	maximum lag value for dvinecopula2 plots

Value

No return value, generates plot.

plot,tscopulafit,missing-method

Plot method for tscopulafit class

Description

Plot method for tscopulafit class

Usage

```
## S4 method for signature 'tscopulafit,missing'
plot(x, plottype = "residual", bw = FALSE, lagmax = 30)
```

Arguments

x	an object of class tscopulafit.
plottype	type of plot required.
bw	logical variable specifying whether black-white options should be chosen.
lagmax	maximum lag value for Kendall plots

Value

No return value, generates plot.

Examples

```
data <- sim(armacopula(list(ar = 0.5, ma = 0.4)), n = 1000)
fit <- fit(armacopula(list(ar = 0.5, ma = 0.4)), data)
plot(fit)
```

plot,Vtransform,missing-method

Plot method for Vtransform class

Description

Plots the v-transform as well as its gradient or inverse. Can also plot the conditional probability that a series PIT falls below the fulcrum for a given volatility PIT value v.

Usage

```
## S4 method for signature 'Vtransform,missing'
plot(
  x,
  type = "transform",
  shading = TRUE,
  npoints = 200,
  lower = 0,
  upper = 1
)
```

Arguments

x	an object of class Vtransform.
type	type of plot: 'transform' for plot of transform, 'inverse' for plot of inverse, 'gradient' for plot of gradient or 'pdown' for plot of conditional probability.
shading	logical variable specifying whether inadmissible zone for v-transform should be shaded
npoints	number of plotting points along x-axis.
lower	the lower x-axis value for plotting.
upper	the upper x-axis value for plotting

Value

No return value, generates plot.

Examples

```
plot(Vsymmetric())
plot(V2p(delta = 0.45, kappa = 0.8), type = "inverse")
plot(V2p(delta = 0.45, kappa = 0.8), type = "gradient")
```

---

pmarg *Compute CDF of marginal model*

Description

Compute the cumulative distribution function of the marginal model.

Usage

```
pmarg(x, q)
```

Arguments

x	an object of class margin.
q	vector of values at which CDF should be computed.

Value

A vector of values for the CDF.

Examples

```
margmod <- margin("norm", pars = c(mean = 0, sd = 1))
pmarg(margmod, c(-2, 0, 2))
```

`profilefulcrum` *Profile likelihood for fulcrum parameter*

Description

Profile likelihood for fulcrum parameter

Usage

```
profilefulcrum(
  data,
  tscopula = dvinecopula(family = 1, pars = list(0.1)),
  locations = seq(0, 1, by = 0.1),
  plot = TRUE
)
```

Arguments

`data`	a vector or time series of data on (0,1).
`tscopula`	an object of class tscopulaU or vtscopula.
`locations`	vector containing locations of different values for fulcrum.
`plot`	logical values specifying whether plot should be created.

Value

A matrix containing fulcrum values and log likelihood values.

Examples

```
copobject <- armacopula(pars = list(ar = 0.6, ma = 0.2))
vtcop <- vtscopula(copobject, Vtransform = V2p())
y <- sim(vtcop)
profilefulcrum(y, vtcop)
```

qmarg *Compute quantiles of marginal model*

Description

Compute the quantile function of the marginal model.

Usage

```
qmarg(x, p)
```

Arguments

x	an object of class margin.
p	vector of probabilities for which quantiles should be computed.

Value

A vector of values for the quantile function.

Examples

```
margmod <- margin("norm", pars = c(mean = 0, sd = 1))
qmarg(margmod, c(0.05, 0.5, 0.95))
```

quantile,tscmfit-method *Quantile calculation method for VT-ARMA models*

Description

Quantile calculation method for VT-ARMA models

Usage

```
## S4 method for signature 'tscmfit'
quantile(x, alpha, last = FALSE)
```

Arguments

x	an object of class tscmfit based on underlying copula of class armacopula.
alpha	a scalar probability value
last	logical value asserting that only the last volatility prediction should be returned

Value

a vector of the same length as the data embedded in the tscmfit object.

safe_ses *Calculate standard errors safely*

Description

Calculate standard errors safely

Usage

```
safe_ses(hess)
```

Arguments

hess	a Hessian matrix from a model fit.

Value

a vector of standard errors.

sdoubleweibull *Skew double Weibull distribution*

Description

Skew double Weibull distribution

Usage

```
dsdoubleweibull(x, mu = 0.05, shape = 1, scale = 1, gamma = 1, log = FALSE)

psdoubleweibull(q, mu = 0.05, shape = 1, scale = 1, gamma = 1)

qsdoubleweibull(p, mu = 0.05, shape = 1, scale = 1, gamma = 1)

rsdoubleweibull(n, mu = 0.05, shape = 1, scale = 1, gamma = 1)
```

Arguments

x	vector of values.
mu	location parameter.
shape	shape parameter.
scale	scale parameter.
gamma	skewness parameter.
log	flag for log density.
q	vector of quantiles.
p	vector of probabilities.
n	number of observations.

Value

A vector of density, distribution function, quantile or random values.

`sigmastarma` *Standard deviation of innovations for armacopula*

Description

Uses the function `tacvfARMA` in the ltsa library.

Usage

```
sigmastarma(x)
```

Arguments

`x`	an object of class armacopula.

Value

The standard deviation of the standardized ARMA innovation distribution.

Examples

```
sigmastarma(armacopula(list(ar = c(0.5, 0.4), ma = -0.8)))
```

`sim` *Generic for simulating time series copula models*

Description

Methods are available for objects of class swncopula, armacopula, dvinecopula, dvinecopula2, margin and tscm.

Usage

```
sim(object, ...)
```

Arguments

`object`	an object of the model class.
`...`	further arguments to be passed to the simulation.

Value

A simulated realization from the time series model.

slaplace *Skew Laplace distribution*

Description

Skew Laplace distribution

Usage

```
dslaplace(x, mu = 0.05, scale = 1, gamma = 1, log = FALSE)

pslaplace(q, mu = 0.05, scale = 1, gamma = 1)

qslaplace(p, mu = 0.05, scale = 1, gamma = 1)

rslaplace(n, mu = 0.05, scale = 1, gamma = 1)
```

Arguments

x	vector of values.
mu	location parameter.
scale	scale parameter.
gamma	skewness parameter.
log	flag for log density.
q	vector of quantiles.
p	vector of probabilities.
n	number of observations.

Value

A vector of density, distribution function, quantile or random values.

sst *Skew Student t distribution*

Description

Skew Student t distribution

Usage

```
psst(q, df = 10, gamma = 1, mu = 0, sigma = 1)

qsst(p, df, gamma, mu, sigma)

dsst(x, df, gamma, mu, sigma, log = FALSE)

rsst(n, df, gamma, mu, sigma)
```

Arguments

q	vector of quantiles.
df	degrees of freedom.
gamma	skewness parameter.
mu	location parameter.
sigma	scale parameter.
p	vector of probabilities.
x	vector of values.
log	flag for log density.
n	number of observations.

Value

A vector of density, distribution function, quantile or random values.

---

st *Student t distribution*

Description

Student t distribution

Usage

```
pst(q, df = 10, mu = 0, sigma = 1)

qst(p, df, mu, sigma)

dst(x, df, mu, sigma, log = FALSE)

rst(n, df, mu, sigma)
```

Arguments

q	vector of quantiles.
df	degrees of freedom.
mu	location parameter.
sigma	scale parameter.
p	vector of probabilities.
x	vector of values.
log	flag for log density.
n	number of observations.

Value

A vector of density, distribution function, quantile or random values.

---

stochinverse *Stochastic inverse of a v-transform*

Description

Stochastic inverse of a v-transform

Usage

```
stochinverse(x, v, tscopula = NULL, tol = .Machine$double.eps^0.75)
```

Arguments

x	an object of class Vtransform.
v	a vector, matrix or time series with values in [0, 1].
tscopula	a time series copula object.
tol	the desired accuracy (convergence tolerance) that is passed to uniroot if numerical inversion is used.

Value

A vector, matrix or time series with values in [0, 1].

Examples

```
stochinverse(Vsymmetric(), c(0, 0.25, 0.5, 0.75, 1))
```

strank *Calculate standardized ranks of data*

Description

Calculate standardized ranks of data

Usage

```
strank(x)
```

Arguments

x	a vector or time series of data.

Value

A vector or time series of standardized ranks in the interval (0,1)

Examples

```
strank(rnorm(100))
```

swncopula *Constructor function for strict white noise copula process*

Description

Constructor function for strict white noise copula process

Usage

```
swncopula()
```

Value

Object of class swncopula.

Examples

```
swncopula()
```

swncopula-class *Strict white noise copula process*

Description

Strict white noise copula process

Usage

```
## S4 method for signature 'swncopula'
sim(object, n = 1000)

## S4 method for signature 'swncopula'
coef(object)

## S4 method for signature 'swncopula'
show(object)
```

Arguments

object	an object of class swncopula.
n	numeric value for length of simulated realisation.

Methods (by generic)

- sim: Simulation method for strict white noise copula
- coef: Coef method for strict white noise copula
- show: Show method for strict white noise copula

Examples

```
sim(swncopula())
```

tscm *Constructor function for time series*

Description

Constructor function for time series

Usage

```
tscm(tscopula, margin = new("margin", name = "unif"))
```

Arguments

tscopula	an object of class tscopula.
margin	an object of class margin.

Value

An object of class tscm.

Examples

```
tscm(dvinecopula(family = "gauss", pars = 0.5), margin("doubleweibull"))
```

---

tscm-class *Full models*

Description

Class of objects for composite time series models consisting of stationary copula processes and marginal distributions.

Usage

```
## S4 method for signature 'tscm'
show(object)

## S4 method for signature 'tscm'
coef(object)

## S4 method for signature 'tscm'
sim(object, n = 1000)
```

Arguments

object	an object of the class.
n	length of realization.

Methods (by generic)

- show: Show method for tscm class
- coef: Coefficient method for tscm class
- sim: Simulation method for tscm class

Slots

tscopula an object of class tscopula.

margin an object of class margin.

Examples

```
mod <- tscm(dvinecopula(family = "gauss", pars = 0.5), margin("doubleweibull"))
sim(mod)
```

---

tscmfit-class *Fitted tscm model*

Description

Class of objects for fitted tscm models.

Usage

```
## S4 method for signature 'tscmfit'
logLik(object)

## S4 method for signature 'tscmfit'
resid(object, trace = FALSE)
```

Arguments

object	an object of the class.
trace	extract trace instead of residuals.

Methods (by generic)

- logLik: method for tscmfit class
- resid: Residual method for tscmfit class

Slots

tscopula an object of class tscopula.

margin an object of class margin.

data a vector or time series of data to which process has been fitted.

fit a list containing details of the fit.

---

tscopula-class *Time series copula processes*

Description

Class of objects for time series copula processes.

`tscopulafit-class` *Fitted time series copula processes*

Description

Class of objects for fitted time series copula processes.

Usage

```
## S4 method for signature 'tscopulafit'
sim(object, n = 1000)

## S4 method for signature 'tscopulafit'
coef(object)

## S4 method for signature 'tscopulafit'
show(object)

## S4 method for signature 'tscopulafit'
logLik(object)

## S4 method for signature 'tscopulafit'
resid(object, trace = FALSE)
```

Arguments

`object`	an object of class tscopulafit.
`n`	length of realization.
`trace`	extract trace instead of residuals.

Methods (by generic)

- `sim`: Simulation method for tscopulafit class
- `coef`: Coef method for tscopulafit class
- `show`: Show method for tscopulafit objects
- `logLik`: logLik method for tscopulafit class
- `resid`: Residual method for tscopulafit class

Slots

`tscopula` an object of class tscopula.

`data` a vector or time series of data.

`fit` a list containing details of the fit.

Examples

```
ar1 <- armacopula(list(ar = 0.7))
data <- sim(ar1, 1000)
ar1fit <- fit(ar1, data)
sim(ar1fit)
```

---

tscopulaU-class *Time series copulas of class tscopulaU*

Description

S4 Class union for basic time series copula types. These are armacopula, dvinecopula and dvinecopula2,

---

V2b *Constructor function for 2-parameter beta v-transform*

Description

Constructor function for 2-parameter beta v-transform

Usage

```
V2b(delta = 0.5, kappa = 1)
```

Arguments

delta	a value in (0, 1) specifying the fulcrum of the v-transform.
kappa	additional positive parameter of v-transform.

Value

An object of class Vtransform.

Examples

```
V2b(delta = 0.45, kappa = 1.2)
```

V2p *Constructor function for 2-parameter v-transform*

Description

Constructor function for 2-parameter v-transform

Usage

```
V2p(delta = 0.5, kappa = 1)
```

Arguments

delta	a value in (0, 1) specifying the fulcrum of the v-transform.
kappa	additional positive parameter of v-transform.

Value

An object of class Vtransform.

Examples

```
V2p(delta = 0.45, kappa = 1.2)
```

V3b *Constructor function for 3-parameter beta v-transform*

Description

Constructor function for 3-parameter beta v-transform

Usage

```
V3b(delta = 0.5, kappa = 1, xi = 1)
```

Arguments

delta	a value in (0, 1) specifying the fulcrum of the v-transform.
kappa	additional positive parameter of v-transform.
xi	additional positive parameter of v-transform.

Value

An object of class Vtransform.

Examples

```
V3b(delta = 0.45, kappa = 1.2, xi = 1.2)
```

V3p *Constructor function for 3-parameter v-transform*

Description

Constructor function for 3-parameter v-transform

Usage

```
V3p(delta = 0.5, kappa = 1, xi = 1)
```

Arguments

delta	a value in (0, 1) specifying the fulcrum of the v-transform.
kappa	additional positive parameter of v-transform.
xi	additional positive parameter of v-transform.

Value

An object of class Vtransform.

Examples

```
V3p(delta = 0.45, kappa = 0.8, xi = 1.1)
```

Vdegenerate *Constructor function for degenerate v-transform*

Description

Constructor function for degenerate v-transform

Usage

```
Vdegenerate()
```

Value

An object of class VtransformI.

Examples

```
Vdegenerate()
```

vdownprob *Calculate conditional down probability of v-transform*

Description

Calculate conditional down probability of v-transform

Usage

```
vdownprob(x, v)
```

Arguments

x	an object of class Vtransform.
v	a vector or time series with values in [0, 1].

Value

A vector or time series of values of gradient.

Examples

```
vdownprob(V2p(delta = 0.55, kapp = 1.2), c(0, 0.25, 0.5, 0.75, 1))
```

vgradient *Calculate gradient of v-transform*

Description

Calculate gradient of v-transform

Usage

```
vgradient(x, u)
```

Arguments

x	an object of class Vtransform.
u	a vector or time series with values in [0, 1].

Value

A vector or time series of values of gradient.

Examples

```
vgradient(Vsymmetric(), c(0, 0.25, 0.5, 0.75, 1))
```

vinverse *Calculate inverse of v-transform*

Description

If the Vtransform object is also a VtransformI object (an invertible v-transform) then the analytical inverse is used. Otherwise an inverse is found by numerical root finding with `uniroot`.

Usage

```
vinverse(x, v, tol = .Machine$double.eps^0.75)
```

Arguments

`x`	an object ofc lass Vtransform.
`v`	a vector or time series with values in [0, 1].
`tol`	the desired accuracy (convergence tolerance) that is passed to `uniroot` if numerical inversion is used.

Value

A vector or time series with values in [0, 1].

Examples

```
vinverse(Vsymmetric(), c(0, 0.25, 0.5, 0.75, 1))
```

Vlinear *Constructor function for linear v-transform*

Description

Constructor function for linear v-transform

Usage

```
Vlinear(delta = 0.5)
```

Arguments

`delta`	a value in (0, 1) specifying the fulcrum of the v-transform.

Value

An object of class VtransformI.

Examples

```
Vlinear(delta = 0.45)
```

Vsymmetric *Constructor function for symmetric v-transform*

Description

Constructor function for symmetric v-transform

Usage

```
Vsymmetric()
```

Value

An object of class VtransformI.

Examples

```
Vsymmetric()
```

---

vtrans *Evaluate a v-transform*

Description

Evaluate a v-transform

Usage

```
vtrans(x, u)
```

Arguments

x	an object of class Vtransform.
u	a vector or time series with values in [0, 1].

Value

A vector or time series with values in [0, 1].

Examples

```
vtrans(Vsymmetric(), c(0, 0.25, 0.5, 0.75, 1))
```

`Vtransform-class` *Class of v-transforms*

Description

This is the class of v-transforms. It contains the VtransformI subclass consisting of v-transforms with an analytical expression for the inverse.

Usage

```
## S4 method for signature 'Vtransform'
show(object)

## S4 method for signature 'Vtransform'
coef(object)
```

Arguments

`object`	an object of the class.

Methods (by generic)

- `show`: Show method for Vtransform class
- `coef`: Coef method for Vtransform class

Slots

`name` a name for the v-transform of class character.

`Vtrans` function to evaluate the v-transform.

`pars` vector containing the named parameters of the v-transform.

`gradient` function to evaluate the gradient of the v-transform.

Examples

```
V2p(delta = 0.5, kappa = 1.2)
```

VtransformI-class *Class of invertible v-transforms*

Description

This class inherits from the Vtransform class and contains v-transforms with an analytical expression for the inverse.

Slots

`name` a name for the v-transform of class character.

`Vtrans` function to evaluate the v-transform.

`pars` vector containing the named parameters of the v-transform.

`gradient` function to evaluate the gradient of the v-transform.

`inverse` function to evaluate the inverse of the v-transform.

Examples

```
Vlinear(delta = 0.55)
```

vtscopula *Constructor function for vtscopula object*

Description

Constructor function for vtscopula object

Usage

```
vtscopula(tscopulaU, Vtransform = Vlinear(), Wcopula = swncopula())
```

Arguments

`tscopulaU`	an object of class armacopula, dvinecopula or dvinecopula2.
`Vtransform`	an object of class Vtransform.
`Wcopula`	an object of class tscopula.

Value

An object of class vtscopula.

Examples

```
copobject <- armacopula(pars = list(ar = 0.6, ma = 0.2))
vtscopula(copobject, Vtransform = V2p())
```

vtscopula-class *Time series copula processes with v-transforms*

Description

Class of objects for v-transformed time series copula processes.

Usage

```
## S4 method for signature 'vtscopula'
show(object)

## S4 method for signature 'vtscopula'
coef(object)

## S4 method for signature 'vtscopula'
sim(object, n = 1000)

## S4 method for signature 'vtscopula'
kendall(object, lagmax = 20)
```

Arguments

object	an object of the class.
n	length of realization.
lagmax	maximum value of lag.

Methods (by generic)

- show: Show method for vtscopula objects
- coef: Coef method for vtscopula class
- sim: Simulation method for vtscopula class
- kendall: Calculate Kendall's tau values for vtscopula model

Slots

Vcopula object of class tscopulaU.

Vtransform object of class Vtransform.

Wcopula object of class tscopula.

Examples

```
copobject <- armacopula(pars = list(ar = 0.6, ma = 0.2))
sim(vtscopula(copobject, Vtransform = V2p()))
mod <- vtscopula(armacopula(list(ar = 0.95, ma = -0.85)))
kendall(mod)
```

Index