Package: esgtoolkit (via r-universe)

September 26, 2024

Type Package

Title Toolkit for Monte Carlo Simulations

Version 1.1.2

Date 2024-05-14

URL https://github.com/Techtonique/esgtoolkit

BugReports https://github.com/Techtonique/esgtoolkit/issues

Maintainer T. Moudiki <thierry.moudiki@gmail.com>

Description A toolkit for Monte Carlo Simulations in Finance, Economics, Insurance, Physics. Multiple simulation models can be created by combining building blocks provided in the package.

License BSD_3_clause Clear + file LICENSE

Depends ggplot2, gridExtra, methods, reshape2, VineCopula, randtoolbox

Imports Rcpp(>= 0.11.0)

Suggests devtools, ggplot2, testthat

LinkingTo Rcpp

NeedsCompilation yes

Date/Publication 2014-06-13 06:17:14

RoxygenNote 7.2.2

Repository https://techtonique.r-universe.dev

RemoteUrl https://github.com/Techtonique/esgtoolkit

RemoteRef HEAD

RemoteSha 9b0c6025cb713abd24c59cb238bfb34857835d97

Contents

`calculatereturns` *Calculate returns or log-returns for multivariate time series*

Description

Calculate returns or log-returns for multivariate time series

Usage

```
calculatereturns(x, type = c("basic", "log"))
```

Arguments

`x`	Multivariate time series
`type`	Type of return: basic return ("basic") or log-return ("log")

Examples

```
kappa <- 1.5
V0 <- theta <- 0.04
sigma_v <- 0.2
theta1 <- kappa*theta
theta2 <- kappa

eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart")
sim_GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
                   model = "GBM",
                x0 = 100, theta1 = 0.03, theta2 = 0.1, eps = eps0)

returns <- calculatereturns(sim_GBM)
log_returns <- calculatereturns(sim_GBM, type = "log")

print(identical(returns, log_returns))
```

```
par(mfrow=c(1, 2))
matplot(returns, type = 'l')
matplot(log_returns, type = 'l')
```

---

esgcortest *Correlation tests for the shocks (if Gaussian copula)*

Description

This function performs correlation tests for the shocks generated by `simshocks`.

Usage

```
esgcortest(
  x,
  alternative = c("two.sided", "less", "greater"),
  method = c("pearson", "kendall", "spearman"),
  conf.level = 0.95
)
```

Arguments

`x`	gaussian (bivariate) shocks, with correlation, generated by `simshocks` (if Gaussian copula).
`alternative`	indicates the alternative hypothesis and must be one of "two.sided", "greater" or "less".
`method`	which correlation coefficient is to be used for the test : "pearson", "kendall", or "spearman".
`conf.level`	confidence level.

Value

a list with 2 components : estimated correlation coefficients, and confidence intervals for the estimated correlations.

Author(s)

T. Moudiki + stats package

References

D. J. Best & D. E. Roberts (1975), Algorithm AS 89: The Upper Tail Probabilities of Spearman's rho. Applied Statistics, 24, 377-379.

Myles Hollander & Douglas A. Wolfe (1973), Nonparametric Statistical Methods. New York: John Wiley & Sons. Pages 185-194 (Kendall and Spearman tests).

See Also

esgplotbands

Examples

```
nb <- 500

s0.par1 <- simshocks(n = nb, horizon = 3, frequency = "semi",
family = 1, par = 0.2)

s0.par2 <- simshocks(n = nb, horizon = 3, frequency = "semi",
family = 1, par = 0.8)

(test1 <- esgcortest(s0.par1))
(test2 <- esgcortest(s0.par2))
#par(mfrow=c(2, 1))
esgplotbands(test1)
esgplotbands(test2)
```

---

esgdiscountfactor *Stochastic discount factors or discounted values*

Description

This function provides calculation of stochastic discount factors or discounted values

Usage

```
esgdiscountfactor(r, X)
```

Arguments

r	the short rate, a `numeric` (constant rate) or a time series object
X	the asset's price, a `numeric` (constant payoff or asset price) or a time series object

Details

The function result is :

$$X_t exp(-\int_0^t r_s ds)$$

where X_t is an asset value at a given maturity t, and $(r_s)_s$ is the risk-free rate.

Author(s)

T. Moudiki

See Also

esgmcprices, esgmccv

Examples

```
kappa <- 1.5
V0 <- theta <- 0.04
sigma_v <- 0.2
theta1 <- kappa*theta
theta2 <- kappa
theta3 <- sigma_v

# OU
r <- simdiff(n = 10, horizon = 5,
               frequency = "quart",
               model = "OU",
               x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = theta3)

# Stochastic discount factors
esgdiscountfactor(r, 1)
```

---

esgfwdrates *Instantaneous forward rates*

Description

This function provides instantaneous forward rates. They can be used in no-arbitrage short rate models, to fit the yield curve exactly.

Usage

```
esgfwdrates(
  in.maturities,
  in.zerorates,
  n,
  horizon,
 out.frequency = c("annual", "semi-annual", "quarterly", "monthly", "weekly", "daily"),
  method = c("fmm", "periodic", "natural", "monoH.FC", "hyman", "HCSPL", "SW"),
  ...
)
```

Arguments

in.maturities	input maturities
in.zerorates	input zero rates
n	number of independent observations
horizon	horizon of projection

out.frequency	either "annual", "semi-annual", "quarterly", "monthly", "weekly", or "daily" (1, 1/2, 1/4, 1/12, 1/52, 1/252)
method	specifies the type of spline to be used for interpolation. Possible values are "fmm", "natural", "periodic", "monoH.FC" and "hyman". See spline; "HCSPL" (Hermite cubic spline) or "SW" (Smith-Wilson)
...	additional parameters provided to splinefun

Author(s)

T. Moudiki

Examples

```
# Yield to maturities
txZC <- c(0.01422,0.01309,0.01380,0.01549,0.01747,0.01940,0.02104,0.02236,0.02348,
         0.02446,0.02535,0.02614,0.02679,0.02727,0.02760,0.02779,0.02787,0.02786,0.02776
      ,0.02762,0.02745,0.02727,0.02707,0.02686,0.02663,0.02640,0.02618,0.02597,0.02578,0.02563)

# Observed maturities
u <- 1:30

par(mfrow=c(2,2))
fwd1 <- esgfwdrates(in.maturities = u, in.zerorates = txZC,
                    n = 10, horizon = 20,
                    out.frequency = "semi-annual", method = "fmm")
matplot(as.vector(time(fwd1)), fwd1, type = 'l')

fwd2 <- esgfwdrates(in.maturities = u, in.zerorates = txZC,
                    n = 10, horizon = 20,
                    out.frequency = "semi-annual", method = "natural")
matplot(as.vector(time(fwd2)), fwd2, type = 'l')

fwd4 <- esgfwdrates(in.maturities = u, in.zerorates = txZC,
                       n = 10, horizon = 20,
                       out.frequency = "semi-annual", method = "hyman")
matplot(as.vector(time(fwd4)), fwd4, type = 'l')
```

---

esgmartingaletest *Martingale and market consistency tests*

Description

This function performs martingale and market consistency (t-)tests.

Usage

```
esgmartingaletest(r, X, p0, alpha = 0.05)
```

Arguments

`r`	a `numeric` or a time series object, the risk-free rate(s).
`X`	a time series object, containing payoffs or projected asset values.
`p0`	a `numeric` or a vector or a univariate time series containing initial price(s) of an asset.
`alpha`	1 - confidence level for the test. Default value is 0.05.

Value

The function result can be just displayed. Otherwise, you can get a list by an assignation, containing (for each maturity) :

- the Student t values
- the p-values
- the estimated mean of the martingale difference
- Monte Carlo prices

Author(s)

T. Moudiki

See Also

`esgplotbands`

Examples

```
r0 <- 0.03
S0 <- 100

set.seed(10)
eps0 <- simshocks(n = 100, horizon = 3, frequency = "quart")
sim.GBM <- simdiff(n = 100, horizon = 3, frequency = "quart",
               model = "GBM",
               x0 = S0, theta1 = r0, theta2 = 0.1,
               eps = eps0, seed=10)

mc.test <- esgmartingaletest(r = r0, X = sim.GBM, p0 = S0,
alpha = 0.05)
esgplotbands(mc.test)
```

esgmccv — *Convergence of Monte Carlo prices*

Description

This function computes and plots confidence intervals around the estimated average price, as functions of the number of simulations.

Usage

```
esgmccv(r, X, maturity, plot = TRUE, ...)
```

Arguments

`r`	a `numeric` or a time series object, the risk-free rate(s).
`X`	asset prices obtained with `simdiff`
`maturity`	the corresponding maturity (optional). If missing, all the maturities available in `X` are used.
`plot`	if `TRUE` (default), a plot of the convergence is displayed.
`...`	additional parameters provided to `matplot`

Details

Studying the convergence of the sample mean of :

$$E[X_T exp(-\int_0^T r_s ds)]$$

towards its true value.

Value

a list with estimated average prices and the confidence intervals around them.

Author(s)

T. Moudiki

Examples

```
r <- 0.03

set.seed(1)
eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart")
sim.GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
               model = "GBM",
               x0 = 100, theta1 = 0.03, theta2 = 0.1,
```

```
                   eps = eps0)

# monte carlo prices
esgmcprices(r, sim.GBM)

# convergence to a specific price
(esgmccv(r, sim.GBM, 2))
```

---

esgmcprices *Estimation of discounted asset prices*

Description

This function computes estimators (sample mean) of

$$E[X_T exp(-\int_0^T r_s ds)]$$

where X_T is an asset value at given maturities T, and $(r_s)_s$ is the risk-free rate.

Usage

```
esgmcprices(r, X, maturity = NULL)
```

Arguments

r	a `numeric` or a time series object, the risk-free rate(s).
X	asset prices obtained with `simdiff`
maturity	the corresponding maturity (optional). If missing, all the maturities available in X are used.

Author(s)

T. Moudiki

See Also

`esgdiscountfactor`, `esgmccv`

Examples

```
# GBM

r <- 0.03

eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart")
sim.GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
```

```
                  model = "GBM",
                  x0 = 100, theta1 = 0.03, theta2 = 0.1,
                  eps = eps0)

# monte carlo prices
esgmcprices(r, sim.GBM)

# monte carlo price for a given maturity
esgmcprices(r, sim.GBM, 2)
```

---

esgplotbands *Plot time series percentiles and confidence intervals*

Description

This function plots colored bands for time series percentiles and confidence intervals. You can use it for outputs from `simdiff`, `esgmartingaletest`, `esgcortest`.

Usage

```
esgplotbands(x, ...)
```

Arguments

`x`	a times series object
`...`	additionnal (optional) parameters provided to `plot`

Author(s)

T. Moudiki

See Also

`esgplotts`

Examples

```
# Times series

kappa <- 1.5
V0 <- theta <- 0.04
sigma <- 0.2
theta1 <- kappa*theta
theta2 <- kappa
theta3 <- sigma
x <- simdiff(n = 100, horizon = 5,
frequency = "quart",
model = "OU",
```

```
x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = theta3)

#par(mfrow=c(2,1))
esgplotbands(x, xlab = "time", ylab = "values")
matplot(as.vector(time(x)), x, type = 'l', xlab = "time", ylab = "series values")

# Martingale test

r0 <- 0.03
S0 <- 100
sigma0 <- 0.1
nbScenarios <- 100
horizon0 <- 10
eps0 <- simshocks(n = nbScenarios, horizon = horizon0, frequency = "quart",
method = "anti")
sim.GBM <- simdiff(n = nbScenarios, horizon = horizon0, frequency = "quart",
                 model = "GBM",
                 x0 = S0, theta1 = r0, theta2 = sigma0,
                 eps = eps0)

mc.test <- esgmartingaletest(r = r0, X = sim.GBM, p0 = S0, alpha = 0.05)
esgplotbands(mc.test)

# Correlation test

nb <- 500

s0.par1 <- simshocks(n = nb, horizon = 3, frequency = "semi",
family = 1, par = 0.2)

s0.par2 <- simshocks(n = nb, horizon = 3, frequency = "semi",
family = 1, par = 0.8)

(test1 <- esgcortest(s0.par1))
(test2 <- esgcortest(s0.par2))
#par(mfrow=c(2, 1))
esgplotbands(test1)
esgplotbands(test2)
```

`esgplotshocks` *Visualize the dependence between 2 gaussian shocks*

Description

This function helps you in visualizing the dependence between 2 gaussian shocks.

Usage

```
esgplotshocks(x, y = NULL)
```

Arguments

x   an output from simshocks, a list with 2 components.

y   an output from simshocks, a list with 2 components (Optional).

Author(s)

T. Moudiki + some nice blogs :)

References

H. Wickham (2009), ggplot2: elegant graphics for data analysis. Springer New York.

See Also

simshocks

Examples

```
# Number of risk factors
d <- 2

# Number of possible combinations of the risk factors
dd <- d*(d-1)/2

# Family : Gaussian copula
fam1 <- rep(1,dd)
# Correlation coefficients between the risk factors (d*(d-1)/2)
par0.1 <- 0.1
par0.2 <- -0.9

# Family : Rotated Clayton (180 degrees)
fam2 <- 13
par0.3 <- 2

# Family : Rotated Clayton (90 degrees)
fam3 <- 23
par0.4 <- -2

# number of simulations
nb <- 500

# Simulation of shocks for the d risk factors
s0.par1 <- simshocks(n = nb, horizon = 4,
family = fam1, par = par0.1)

s0.par2 <- simshocks(n = nb, horizon = 4,
family = fam1, par = par0.2)

s0.par3 <- simshocks(n = nb, horizon = 4,
family = fam2, par = par0.3)
```

```
s0.par4 <- simshocks(n = nb, horizon = 4,
family = fam3, par = par0.4)


esgplotshocks(s0.par1, s0.par2)
esgplotshocks(s0.par2, s0.par3)
esgplotshocks(s0.par2, s0.par4)
esgplotshocks(s0.par1, s0.par4)
```

---

esgplotts *Plot time series objects*

Description

This function plots outputs from `simdiff`.

Usage

```
esgplotts(x)
```

Arguments

x	a time series object, an output from `simdiff`.

Details

For a large number of simulations, it's preferable to use `esgplotbands` for a synthetic view by percentiles.

Author(s)

T. Moudiki

References

H. Wickham (2009), ggplot2: elegant graphics for data analysis. Springer New York.

See Also

`simdiff`, `esgplotbands`

Examples

```
kappa <- 1.5
V0 <- theta <- 0.04
sigma <- 0.2
theta1 <- kappa*theta
theta2 <- kappa
theta3 <- sigma
x <- simdiff(n = 10, horizon = 5, frequency = "quart",
model = "OU",
x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = theta3)

esgplotts(x)
```

---

forwardrates *Forward rates extraction*

Description

This function extracts the forward rates from the results obtained with `ycinter` and `ycextra`

Usage

```
forwardrates(.Object)
```

Arguments

`.Object`	An S4 object created by `ycinter` or `ycextra`.

Value

A time series object giving the instantaneous forward rates for `methods` "NS", "SV" and the forward rates for `methods` "HCSPL", "SW"

Author(s)

Thierry Moudiki

See Also

`ycinter`, `ycextra`

Examples

```
# Prices
 p <- c(0.9859794,0.9744879,0.9602458,0.9416551,0.9196671,0.8957363,0.8716268,0.8482628,
 0.8255457,0.8034710,0.7819525,0.7612204,0.7416912,0.7237042,0.7072136
 ,0.6922140,0.6785227,0.6660095,0.6546902,0.6441639,0.6343366,0.6250234,0.6162910,0.6080358,
 0.6003302,0.5929791,0.5858711,0.5789852,0.5722068,0.5653231)

 # Observed maturities
 u <- 1:30

 # Output maturities
 t <- seq(from = 1, to = 60, by = 0.5)

 # Svensson interpolation
 yc <- ycextra(p = p, matsin = u, matsout = t,
 method="SV", typeres="prices", UFR = 0.018)

 plot(forwardrates(yc))
```

`simdiff` *Simulation of diffusion processes.*

Description

This function makes simulations of diffusion processes, that are building blocks for various risk factors' models.

Usage

```
simdiff(
  n,
  horizon,
  frequency = c("annual", "semi-annual", "quarterly", "monthly", "weekly", "daily"),
  model = c("GBM", "CIR", "OU"),
  x0,
  theta1 = NULL,
  theta2 = NULL,
  theta3 = NULL,
  lambda = NULL,
  mu_z = NULL,
  sigma_z = NULL,
  p = NULL,
  eta_up = NULL,
  eta_down = NULL,
  eps = NULL,
  start = NULL,
  seed = 123
)
```

Arguments

n
: number of independent observations.

horizon
: horizon of projection.

frequency
: either "annual", "semi-annual", "quarterly", "monthly", "weekly", or "daily" (1, 1/2, 1/4, 1/12, 1/52, 1/252).

model
: either Geometric Brownian motion-like ("GBM"), Cox-Ingersoll-Ross ("CIR"), or Ornstein-Uhlenbeck ("OU").

GBM-like (GBM, Merton, Kou, Heston, Bates)

$$dX_t = \theta_1(t)X_t dt + \theta_2(t)X_t dW_t + X_t J dN_t$$

CIR

$$dX_t = (\theta_1 - \theta_2 X_t)dt + \theta_3\sqrt{(}X_t)dW_t$$

Ornstein-Uhlenbeck

$$dX_t = (\theta_1 - \theta_2 X_t)dt + \theta_3 dW_t$$

Where $(W_t)_t$ is a standard brownian motion :

$$dW_t \ \ \epsilon\sqrt{(}dt)$$

and

$$\epsilon \ \ N(0,1)$$

The ϵ is a gaussian increment that can be an output from simshocks.

For 'GBM-like', θ_1 and θ_2 can be held constant, and the jumps part JdN_t is optional. In case the jumps are used, they arise following a Poisson process (N_t), with intensities J drawn either from lognormal or asymmetric double-exponential distribution.

x0
: starting value of the process.

theta1
: a numeric for model = "GBM", model = "CIR", model = "OU". Can also be a time series object (an output from simdiff with the same number of scenarios, horizon and frequency) for model = "GBM", and time-varying parameters.

theta2
: a numeric for model = "GBM", model = "CIR", model = "OU". Can also be a time series object (an output from simdiff with the same number of scenarios, horizon and frequency) for model = "GBM", and time-varying parameters.

theta3
: a numeric, volatility for model = "CIR" and model = "OU".

lambda
: intensity of the Poisson process counting the jumps. Optional.

mu_z
: mean parameter for the lognormal jumps size. Optional.

sigma_z
: standard deviation parameter for the lognormal jumps size. Optional.

p
: probability of positive jumps. Must belong to]0, 1[. Optional.

eta_up
: mean of positive jumps in Kou's model. Must belong to]0, 1[. Optional.

eta_down
: mean of negative jumps. Must belong to]0, 1[. Optional.

eps	gaussian shocks. If not provided, independent shocks are generated internally by the function. Otherwise, for custom shocks, must be an output from simshocks.
start	the time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit. See '?ts'.
seed	reproducibility seed

Value

a time series object.

Author(s)

T. Moudiki

References

Black, F., Scholes, M.S. (1973) The pricing of options and corporate liabilities, Journal of Political Economy, 81, 637-654.

Cox, J.C., Ingersoll, J.E., Ross, S.A. (1985) A theory of the term structure of interest rates, Econometrica, 53, 385-408.

Iacus, S. M. (2009). Simulation and inference for stochastic differential equations: with R examples (Vol. 1). Springer.

Glasserman, P. (2004). Monte Carlo methods in financial engineering (Vol. 53). Springer.

Kou S, (2002), A jump diffusion model for option pricing, Management Sci- ence Vol. 48, 1086-1101.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. Journal of financial economics, 3(1), 125-144.

Uhlenbeck, G. E., Ornstein, L. S. (1930) On the theory of Brownian motion, Phys. Rev., 36, 823-841.

Vasicek, O. (1977) An Equilibrium Characterization of the Term Structure, Journal of Financial Economics, 5, 177-188.

See Also

simshocks, esgplotts

Examples

```
kappa <- 1.5
V0 <- theta <- 0.04
sigma_v <- 0.2
theta1 <- kappa*theta
theta2 <- kappa
theta3 <- sigma_v

# OU
```

```
sim.OU <- simdiff(n = 10, horizon = 5,
               frequency = "quart",
               model = "OU",
               x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = theta3)
head(sim.OU)
#par(mfrow=c(2,1))
esgplotbands(sim.OU, xlab = "time", ylab = "values", main = "with esgplotbands")
matplot(as.vector(time(sim.OU)), sim.OU, type = 'l', main = "with matplot")


# OU with simulated shocks (check the dimensions)

eps0 <- simshocks(n = 50, horizon = 5, frequency = "quart", method = "anti")
sim.OU <- simdiff(n = 50, horizon = 5, frequency = "quart",
               model = "OU",
               x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = theta3,
               eps = eps0)
#par(mfrow=c(2,1))
esgplotbands(sim.OU, xlab = "time", ylab = "values", main = "with esgplotbands")
matplot(as.vector(time(sim.OU)), sim.OU, type = 'l', main = "with matplot")
# a different plot
esgplotts(sim.OU)


# CIR

sim.CIR <- simdiff(n = 50, horizon = 5,
               frequency = "quart",
               model = "CIR",
               x0 = V0, theta1 = theta1, theta2 = theta2, theta3 = 0.05)
esgplotbands(sim.CIR, xlab = "time", ylab = "values", main = "with esgplotbands")
matplot(as.vector(time(sim.CIR)), sim.CIR, type = 'l', main = "with matplot")



# GBM

eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart")
sim.GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
               model = "GBM",
               x0 = 100, theta1 = 0.03, theta2 = 0.1,
               eps = eps0)
esgplotbands(sim.GBM, xlab = "time", ylab = "values", main = "with esgplotbands")
matplot(as.vector(time(sim.GBM)), sim.GBM, type = 'l', main = "with matplot")


eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart")
sim.GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
               model = "GBM",
               x0 = 100, theta1 = 0.03, theta2 = 0.1,
               eps = eps0)
esgplotbands(sim.GBM, xlab = "time", ylab = "values", main = "with esgplotbands")
```

```
matplot(as.vector(time(sim.GBM)), sim.GBM, type = 'l', main = "with matplot")



# GBM log returns (haha) with starting date

eps0 <- simshocks(n = 100, horizon = 5, frequency = "quart", start = c(1995, 1))
sim.GBM <- simdiff(n = 100, horizon = 5, frequency = "quart",
               model = "GBM",
               x0 = 100, theta1 = 0.03, theta2 = 0.1,
               eps = eps0, start = c(1995, 1))
log_returns_GBM <-  calculatereturns(sim.GBM, type = "log")
par(mfrow=c(1, 2))
esgplotbands(log_returns_GBM , xlab = "time", ylab = "values", main = "with esgplotbands")
matplot(as.vector(time(log_returns_GBM)), log_returns_GBM, type = 'l',
main = "with matplot", xlab = "time")
```

`simshocks` *Underlying gaussian shocks for risk factors' simulation.*

Description

This function makes simulations of correlated or dependent gaussian shocks for risk factors.

Usage

```
simshocks(
  n,
  horizon,
  frequency = c("annual", "semi-annual", "quarterly", "monthly", "weekly", "daily"),
  method = c("classic", "antithetic", "mm", "hybridantimm", "TAG"),
  family = NULL,
  par = NULL,
  par2 = rep(0, length(par)),
  RVM = NULL,
  type = c("CVine", "DVine", "RVine"),
  start = NULL,
  seed = 123
)
```

Arguments

`n`	number of independent observations for each risk factor.
`horizon`	horizon of projection.
`frequency`	either "annual", "semi-annual", "quarterly", "monthly", "weekly", or "daily" (1, 1/2, 1/4, 1/12, 1/52, 1/252).

method	either classic monte carlo, antithetic variates, moment matching, hybrid antithetic variates + moment matching, "TAG" (see the 4th reference for the latter. Options: "classic", "antithetic", "mm", "hybridantimm", "TAG".
family	A d*(d-1)/2 integer vector of C-/D-vine pair-copula families with values 0 = independence copula, 1 = Gaussian copula, 2 = Student t copula (t-copula), 3 = Clayton copula, 4 = Gumbel copula, 5 = Frank copula, 6 = Joe copula, 7 = BB1 copula, 8 = BB6 copula, 9 = BB7 copula, 10 = BB8 copula, 13 = rotated Clayton copula (180 degrees; "survival Clayton"), 14 = rotated Gumbel copula (180 degrees; "survival Gumbel"), 16 = rotated Joe copula (180 degrees; "survival Joe"), 17 = rotated BB1 copula (180 degrees; "survival BB1"), 18 = rotated BB6 copula (180 degrees; "survival BB6"), 19 = rotated BB7 copula (180 degrees; "survival BB7"), 20 = rotated BB8 copula (180 degrees; "survival BB8"), 23 = rotated Clayton copula (90 degrees), 24 = rotated Gumbel copula (90 degrees), 26 = rotated Joe copula (90 degrees), 27 = rotated BB1 copula (90 degrees), 28 = rotated BB6 copula (90 degrees), 29 = rotated BB7 copula (90 degrees), 30 = rotated BB8 copula (90 degrees), 33 = rotated Clayton copula (270 degrees), 34 = rotated Gumbel copula (270 degrees), 36 = rotated Joe copula (270 degrees), 37 = rotated BB1 copula (270 degrees), 38 = rotated BB6 copula (270 degrees), 39 = rotated BB7 copula (270 degrees), 40 = rotated BB8 copula (270 degrees)
par	A d*(d-1)/2 vector of pair-copula parameters.
par2	A d*(d-1)/2 vector of second parameters for pair-copula families with two parameters (t, BB1, BB6, BB7, BB8; no default).
RVM	An `RVineMatrix` object containing the information of the R-vine copula model. Optionally, a length-N list of `VineCopula::RVineMatrix` objects sharing the same structure, but possibly different family/parameter can be supplied. Must be not NULL for type == "RVine", not used otherwise. See also VineCopula::RVineMatrix.
type	type of vine model: "CVine", "DVine" or "RVine"
start	the time of the first observation. Either a single number or a vector of two numbers (the second of which is an integer), which specify a natural time unit and a (1-based) number of samples into the time unit. See '?ts'.
seed	reproducibility seed

Details

The function shall be used along with `simdiff`, in order to embed correlated or dependent random gaussian shocks into simulated diffusions. `esgplotshocks` can help in visualizing the type of dependence between the shocks.

Value

If `family` and `par` are not provided, a univariate time series object with simulated gaussian shocks for one risk factor. Otherwise, a list of time series objects, containing gaussian shocks for each risk factor.

Author(s)

T. Moudiki

References

Brechmann, E., Schepsmeier, U. (2013). Modeling Dependence with C- and D-Vine Copulas: The R Package CDVine. Journal of Statistical Software, 52(3), 1-27. URL https://www.jstatsoft.org/v52/i03/.

Genz, A. Bretz, F., Miwa, T. Mi, X., Leisch, F., Scheipl, F., Hothorn, T. (2013). mvtnorm: Multivariate Normal and t Distributions. R package version 0.9-9996.

Genz, A. Bretz, F. (2009), Computation of Multivariate Normal and t Probabilities. Lecture Notes in Statistics, Vol. 195., Springer-Verlag, Heidelberg. ISBN 978-3-642-01688-2.

Thomas Nagler, Ulf Schepsmeier, Jakob Stoeber, Eike Christian Brechmann, Benedikt Graeler and Tobias Erhardt (2020). VineCopula: Statistical Inference of Vine Copulas. R package version 2.4.0. https://CRAN.R-project.org/package=VineCopula

Nteukam T, O., & Planchet, F. (2012). Stochastic evaluation of life insurance contracts: Model point on asset trajectories and measurement of the error related to aggregation. Insurance: Mathematics and Economics, 51(3), 624-631. URL http://www.ressources-actuarielles.net/EXT/ISFA/1226.nsf/0/ab539dcebcc4e77ac12576c6004afa67/$FILE/Article_US_v1.5.pdf

See Also

simdiff, esgplotshocks

Examples

```
# Number of risk factors
d <- 6

# Number of possible combinations of the risk factors
dd <- d*(d-1)/2

# Family : Gaussian copula for all
fam1 <- rep(1,dd)

# Correlation coefficients between the risk factors (d*(d-1)/2)
par1 <- c(0.2,0.69,0.73,0.22,-0.09,0.51,0.32,0.01,0.82,0.01,
        -0.2,-0.32,-0.19,-0.17,-0.06)


# Simulation of shocks for the 6 risk factors
simshocks(n = 10, horizon = 5, family = fam1, par = par1)


# Simulation of shocks for the 6 risk factors
# on a quarterly basis
simshocks(n = 10, frequency = "quarterly", horizon = 2, family = fam1,
par = par1)


# Simulation of shocks for the 6 risk factors simulation
# on a quarterly basis, with antithetic variates and moment matching.
s0 <- simshocks(n = 10, method = "hyb", horizon = 4,
```

```
family = fam1, par = par1)


s0[[2]]
colMeans(s0[[1]])
colMeans(s0[[5]])
apply(s0[[3]], 2, sd)
apply(s0[[4]], 2, sd)
```

---

ycextra *Yield curve or zero-coupon prices extrapolation*

Description

Yield curve or zero-coupon bonds prices curve extrapolation using the Nelson-Siegel, Svensson, Smith-Wilson models.

Usage

```
ycextra(yM = NULL, p = NULL, matsin, matsout,
  method = c("NS", "SV", "SW"),
  typeres = c("rates", "prices"), UFR, T_UFR = NULL)
```

Arguments

`yM`	A vector of non-negative numerical quantities, containing the yield to maturities.
`p`	A vector of non-negative numerical quantities, containing the zero-coupon prices.
`matsin`	A vector containing the observed maturities.
`matsout`	the output maturities needed.
`method`	A character string giving the type of method used fo intepolation and extrapolation. `method` can be either "NS" for Nelson-Siegel, "SV" for Svensson, or "SW" Smith-Wilson.
`typeres`	A character string, giving the type of return. Either "prices" or "rates".
`UFR`	The ultimate forward rate.
`T_UFR`	The number of years after which the yield curve converges to the UFR. `T_UFR` is used only when `method` is "SW".

Details

This function interpolates between observed points of a yield curve, or zero-coupon prices, and extrapolates the curve using the Nelson-Siegel, Svensson, Smith-Wilson models. The result can be either prices or zero rates. For the purpose of extrapolation, an ultimate forward rate (UFR) to which the yield curve converges must be provided. With the Smith-Wilson method, a period of convergence (number of years) to the ultimate forward rate, after the last liquid point, must be provided.

Value

An S4 Object

Author(s)

Thierry Moudiki

Examples

```
# Yield to maturities
 txZC <- c(0.01422,0.01309,0.01380,0.01549,0.01747,0.01940,0.02104,0.02236,0.02348,
 0.02446,0.02535,0.02614,0.02679,0.02727,0.02760,0.02779,0.02787,0.02786,0.02776
 ,0.02762,0.02745,0.02727,0.02707,0.02686,0.02663,0.02640,0.02618,0.02597,0.02578,0.02563)

 # Prices
 p <- c(0.9859794,0.9744879,0.9602458,0.9416551,0.9196671,0.8957363,0.8716268,0.8482628,
 0.8255457,0.8034710,0.7819525,0.7612204,0.7416912,0.7237042,0.7072136
 ,0.6922140,0.6785227,0.6660095,0.6546902,0.6441639,0.6343366,0.6250234,0.6162910,0.6080358,
 0.6003302,0.5929791,0.5858711,0.5789852,0.5722068,0.5653231)

 # Observed maturities
 u <- 1:30

 # Output maturities
 t <- seq(from = 1, to = 60, by = 0.5)

 # Svensson extrapolation
(yc <- ycextra(p = p, matsin = u, matsout = t,
 method="SV", typeres="prices", UFR = 0.018))

 #Smith-Wilson extrapolation
 (yc <- ycextra(p = p, matsin = u, matsout = t,
 method="SW", typeres="rates", UFR = 0.019, T_UFR = 20))

 # Nelson-Siegel extrapolation
 (yc <- ycextra(yM = txZC, matsin = u, matsout = t,
 method="NS", typeres="prices", UFR = 0.029))
```

---

ycinter — *Yield curve or zero-coupon prices interpolation*

Description

Yield curve or zero-coupon bonds prices curve interpolation using the Nelson-Siegel , Svensson, Smith-Wilson models and an Hermite cubic spline.

Usage

```
ycinter(yM = NULL, p = NULL, matsin, matsout,
  method = c("NS", "SV", "SW", "HCSPL"),
  typeres = c("rates", "prices"))
```

Arguments

yM	A vector of non-negative numerical quantities, containing the yield to maturities.
p	A vector of non-negative numerical quantities, containing the zero-coupon prices.
matsin	A vector containing the observed maturities.
matsout	the output maturities needed.
method	A character string giving the type of method used fo intepolation. `method` can be either "NS" for Nelson-Siegel, "SV" for Svensson, "HCSPL" for Hermite cubic spline, or "SW" Smith-Wilson.
typeres	A character string, giving the type of return. Either "prices" or "rates".

Details

This function interpolates between observed points of a yield curve, or zero-coupon prices, using the Nelson-Siegel, Svensson, Smith-Wilson models and an Hermite cubic spline. The result can be either prices or zero rates.

Value

An S4 Object

Author(s)

Thierry Moudiki

Examples

```
## Interpolation of yields to matuities with prices as outputs

 # Yield to maturities
 txZC <- c(0.01422,0.01309,0.01380,0.01549,0.01747,0.01940,0.02104,0.02236,0.02348,
 0.02446,0.02535,0.02614,0.02679,0.02727,0.02760,0.02779,0.02787,0.02786,0.02776
 ,0.02762,0.02745,0.02727,0.02707,0.02686,0.02663,0.02640,0.02618,0.02597,0.02578,0.02563)

 # Zero-coupon prices
 p <- c(0.9859794,0.9744879,0.9602458,0.9416551,0.9196671,0.8957363,0.8716268,0.8482628,
 0.8255457,0.8034710,0.7819525,0.7612204,0.7416912,0.7237042,0.7072136
 ,0.6922140,0.6785227,0.6660095,0.6546902,0.6441639,0.6343366,0.6250234,0.6162910,0.6080358,
 0.6003302,0.5929791,0.5858711,0.5789852,0.5722068,0.5653231)

 # Observed maturities
 u <- 1:30

 # Output maturities
 t <- seq(from = 1, to = 30, by = 0.5)

 # Cubic splines interpolation
 (yc <- ycinter(yM = txZC, matsin = u, matsout = t,
 method="HCSPL", typeres="rates"))
```

```
# Nelson-Siegel interpolation
(yc <- ycinter(yM = txZC, matsin = u, matsout = t,
method="NS", typeres="prices"))

# Svensson interpolation
(yc <- ycinter(p = p, matsin = u, matsout = t,
method="SV", typeres="prices"))

#Smith-Wilson interpolation
(yc <- ycinter(p = p, matsin = u, matsout = t,
method="SW", typeres="rates"))
```

Index