Package ‘fMultivar’

March 7, 2020

Title Rmetrics - Analysing and Modeling Multivariate Financial Return Distributions

Date 2017-11-12

Version 3042.80.1

Author Diethelm Wuertz [aut],
Tobias Setz [cre],
Yohan Chalabi [ctb]

Maintainer Tobias Setz <tobias.setz@live.com>

Description Provides a collection of functions
to manage, to investigate and to analyze bivariate and multivariate
data sets of financial returns.

Depends R (>= 2.15.1), timeDate, timeSeries, fBasics

Imports cubature, mvtnorm, sn, methods, grDevices, graphics, stats

Suggests spatial, RUnit, tcltk, akima

LazyData yes

License GPL (>= 2)

URL https://www.rmetrics.org

NeedsCompilation no

Repository CRAN

Date/Publication 2020-03-07 11:06:19 UTC

R topics documented:

`fMultivar-package` *Modelling Multivariate Return Distributions*

Description

The Rmetrics "fMultivar"" package is a collection of functions to manage, to investigate and to analyze bivariate and multivariate data sets of financial returns.

Details

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

1 Introduction

The package `fMultivar` was written to explore and investigate bivariate and multivariate financial return series. The bivariate modeling allows us the comparison of financial returns from two investments or from one investment and its benchmark. When it comes to the investigation of multiple investment returns from funds or portfolios we are concerned with the multivariate case.

In the case of bivariate distribution functions we provide functions for the 2-dimensional Cauchy, Normal, and Student-t distributions. A generalisation (for the density only) is made for the family of 2-dimensional elliptical distributions. In this case we provide density functions for the Normal, Cauchy, Student-t, Logistic, Laplace, Kotz, e-Power distributions.

In the case of multivariate distribution functions from the skew-normal (SN) family and some related ones we recommend to use the density funtions, probability functions and random number generators provided by Azzalini's contributed package sn. The family of his SN-distributions cover the skew Cauchy, the skew Normal, and the skew Student-t distributions. For parameter fitting we have added three simple wrapper functions for an easy to use approach to estimate the distributional parameters for financial return series.

In the case of multivariate distribution functions from the generalized hyperbolic (GHYP) family and some related ones we recommend to use the density funtions, probability functions and random number generators provided by David Luethi and Wolfgang Breymann's contributed package ghyp. The family of their GHYP-distributions cover beside the General Hyperbolic distribution (GHYP) also the special cases for the Hyperbolic distribution (HYP), for the Normal Inverse Gaussian distribution (NIG), for the Variance Gamma distribution (VG), and for the skewed Student-t distribution (GHST).

2 Bivariate Distributions

This section contains functions to model bivariate density, probability, quantile functions, and to generate random numbers for three standard distributions.

```
[dpr]cauchy2d        Bivariate Cauchy Distribution
[dpr]norm2d          Bivariate Normal Disribution
[dpr]t2d             Bivariate Student-t Disribution
```

The density function

```
delliptical2d        Bivariate Elliptical Densities
```

computes for several bivariate elliptical distributions their densities. Included distributions are the following types: "norm", "cauchy", "t", "logistic", "laplace", "kotz", and "epower".

3 Multivariate Symmetric Distributions

```
[dpr]              Multivariate Cauchy Distribution
[dpr]              Multivariate Normal Distribution
[dpr]              Multivariate Student-t Distribution

[dpr]              Multivariate Truncated Normal Distribution
```

3 Multivariate Skew Distributions

We use the functions from the contributed package "sn" package to model multivariate density and probability functions, and to generate random numbers for the skew Cauchy, Normal and Student-t distributions. Note the symmetric case is also included in these functions. The functions are:

```
[dpr]msc             Multivariate Skew Cauchy Distribution
[dpr]msn             Multivariate Skew Normal Distribution
[dpr]mst             Multivariate Skew Student-t Distribution
```

Note the functions are not part of the `fMultivar` package they depend on the `"sn"` package and are loaded when `fMultivar` is loaded.

NOTE: In the new version of the `fMultivar` package the following two distribution functions `*mvsnorm` (multivariate Normal distribution) and `*mvst` (multivariate Student-t Distribution) will become obsolete together with the `mvFit` parameter estimation function. The functionality is fully covered by the `"sn"` package. (They will be most likely deprecated in the future.)

For parameter estimation please use the simple wrapper functions:

```
mscFit              Multivariate Skew Cauchy Fit
msnFit              Multivariate Skew Normal Fit
mstFit              Multivariate Skew Student-t Fit
```

Thes parameter estimation functions will be in the same style as all the other fitting functions in other Rmetrics packages.

4 Multivariate GHYP Distributions

We refer to the package `"ghyp"` authored by David Luethi and Wolfgang Breymann,

5 Utility Functions

We have also added some very useful utility functions for the bivariate case, these include 2-D grid generation, squared and hexagonal binned histograms, 2-D kernel density estimates, bivariate histogram plots:

```
grid2d              Bivariate Square Grid of Coordinates
binning2d           Bivariate Square/Hexagonal Binning Plot
density2d           Bivariate Kernel Density Plot
hist2d              Bivariate Histogram Plot
gridData            Bivariate gridded data set
```

For integration we have added two quadratur routines a simple one for the bivariate case and an adaptive one for the multivariate case:

```
integrate2d         Bivariate Integration
adapt               Multivariate adaptive Quadratur
```

The function `adapt` is a wrapper to the function `adaptIntegrate` from the new contributed package `cubature` authored by Stephan G. Johnson.

About Rmetrics:

The `fMultivar` Rmetrics package is written for educational support in teaching "Computational Finance and Financial Engineering" and licensed under the GPL.

bvdist-cauchy2d *Bivariate Cauchy Distribution*

Description

Density, distribution function, and random generation for the bivariate Cauchy distribution.

Usage

```
dcauchy2d(x, y, rho = 0)
pcauchy2d(x, y, rho = 0)
rcauchy2d(n, rho = 0)
```

Arguments

x, y	two numeric vectors defining the x and y coordinates.
n	the number of random deviates to be generated, an integer value.
rho	the correlation parameter, a numeric value ranging between minus one and one, by default zero.

Value

pcauchy2d
returns a two column matrix of probabilities for the bivariate Cauchy distribution function.

dcauchy2d
returns a two column matrix of densities for the bivariate Cauchy distribution function.

rcauchy2d
returns a two column matrix of random deviates generated from the bivariate Cauchy distribution function.

Author(s)

Adelchi Azzalini for the underlying pnorm2d function,
Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Examples

```
## Bivariate Cauchy Density:
   x <- (-40:40)/10
   X <- grid2d(x)
   z <- dcauchy2d(X$x, X$y, rho = 0.5)
   Z <- list(x = x, y = x, z = matrix(z, ncol = length(x)))

## Perspective Density Plot:
   persp(Z, theta = -40, phi = 30, col = "steelblue")

## Image Density Plot with Contours:
   image(Z, main="Bivariate Cauchy")
   contour(Z, add=TRUE)
```

---

bvdist-elliptical2d *Bivariate Elliptical Densities*

Description

Density function for bivariate elliptical distributions.

Usage

```
delliptica12d(x, y, rho = 0, param = NULL, type = c("norm", "cauchy", "t",
    "logistic", "laplace", "kotz", "epower"), output = c("vector", "list"))
```

Arguments

x, y	two numeric vectors defining the x and y coordinates.
output	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	additional parameters to specify the bivariate density function. Only effective for the Kotz and Exponential Power distribution. For the Kotz distribution we can specify a numeric value, by default defined as param=c(r=sqrt(2)), and for the Exponential Power distribution a numeric vector, by default defined as param=c(r=sqrt(2)),s=1/2.
rho	the correlation parameter, a numeric value ranging between minus one and one, by default zero.
type	the type of the elliptical copula. A character string selected from: "norm", "cauchy", "t", "laplace", "kotz", or "epower".

Value

`delliptical2d`
returns a two column matrix of densities for the selected bivariate elliptical distribution function.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Examples

```
## delliptical2d -
   # Kotz' Elliptical Density:
   x <- (-40:40)/10
   X <- grid2d(x)
   z <- delliptical2d(X$x, X$y, rho = 0.5, type = "kotz")
   Z <- list(x = x, y = x, z = matrix(z, ncol = length(x)))

## Perspective Plot:
   persp(Z, theta = -40, phi = 30, col = "steelblue")

## Image Plot with Contours:
   image(Z, main = "Bivariate Kotz")
   contour(Z, add=TRUE)

## Internal Density Slider:
   ## Not run:
   .delliptical2dSlider()

## End(Not run)
```

---

bvdist-norm2d *Bivariate Normal Distribution*

Description

Density, distribution function, and random generation for the bivariate normal distribution.

Usage

```
dnorm2d(x, y, rho = 0)
pnorm2d(x, y, rho = 0)
rnorm2d(n, rho = 0)
```

Arguments

`x, y`	two numeric vectors defining the x and y coordinates.
`n`	the number of random deviates to be generated, an integer value.
`rho`	the correlation parameter, a numeric value ranging between minus one and one, by default zero.

Value

`pnorm2d`
returns a two column matrix of probabilities for the bivariate normal distribution function.

`dnorm2d`
returns a two column matrix of densities for the bivariate normal distribution function.

`rnorm2d`
returns a two column matrix of random deviates generated from the bivariate normal distribution function.

Author(s)

Adelchi Azzalini for the underlying `pnorm2d` function,
Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Examples

```
## dnorm2d -
   # Bivariate Normal Density:
   x <- (-40:40)/10
   X <- grid2d(x)
   z <- dnorm2d(X$x, X$y, rho = 0.5)
   ZD <- list(x = x, y = x, z = matrix(z, ncol = length(x)))
   # Perspective Density Plot:
   persp(ZD, theta = -40, phi = 30, col = "steelblue")
   # Contour Density Plot:
   contour(ZD, main="Bivariate Normal Density")

## pnorm2d -
   # Bivariate Normal Probability:
   z <- pnorm2d(X$x, X$y, rho = 0.5)
   ZP <- list(x = x, y = x, z = matrix(z, ncol = length(x)))
   # Perspective Plot:
   persp(ZP, theta = -40, phi = 30, col = "steelblue")
   # Contour Plot:
   contour(ZP)
```

```
## rnorm2d -
   # Bivariate Normal Random Deviates
   r <- rnorm2d(5000, rho=0.5)
   # Scatter Plot:
   plot(r, col="steelblue", pch=19, cex=0.5)
   contour(ZD, add=TRUE, lwd=2, col="red")
   # Hexagonal Binning:
   plot(hexBinning(r))
   contour(ZD, add=TRUE, lwd=2, col="black")
```

---

bvdist-t2d *Bivariate Student-t Distribution*

Description

Density, distribution function, and random generation for the bivariate Student-t distribution.

Usage

```
dt2d(x, y, rho = 0, nu = 4)
pt2d(x, y, rho = 0, nu = 4)
rt2d(n, rho = 0, nu = 4)
```

Arguments

n	the number of random deviates to be generated, an integer value.
nu	the number of degrees of freedom, a numeric value ranging between two and infinity, by default four.
rho	the correlation parameter, a numeric value ranging between minus one and one, by default zero.
x, y	two numeric vectors defining the x and y coordinates.

Value

`pt2d`
returns a two column matrix of probabilities for the bivariate Student-t distribution function.

`dt2d`
returns a two column matrix of densities for the bivariate Student-t distribution function.

`rt2d`
returns a two column matrix of random deviates generated from the bivariate Student-t distribution function.

Author(s)

Adelchi Azzalini for the underlying pnorm2d function,
Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Examples

```
## dt2d -
   # Bivariate Student-t Density:
   x <- (-40:40)/10
   X <- grid2d(x)
   z <- dt2d(X$x, X$y, rho = 0.5, nu = 6)
   Z <- list(x = x, y = x, z = matrix(z, ncol = length(x)))
   # Perspective Plot:
   persp(Z, theta = -40, phi = 30, col = "steelblue")
   # Contour Plot:
   contour(Z)

## pt2d -
   # Bivariate Student-t Probability:
   x <- (-40:40)/10
   X <- grid2d(x)
   z <- pt2d(X$x, X$y, rho = 0.5, nu = 6)
   Z <- list(x = x, y = x, z = matrix(z, ncol = length(x)))
   # Image Plot with Contours:
   image(Z)
   contour(Z, add=TRUE)
```

---

mvdist-msc *Multivariate Skew Cauchy Distribution*

Description

Density, distribution function, and random number generation for the multivariate Cauchy distribution.

Details

The functions to compute densities dmsc, probabilities pmsc, and to generate random numbers rmsc for the multivariate skew Cauchy distribution are available in the contributed R package sn (note, they are no longer builtin in fMultivar). The reason is that the performance for these functions in package sn has superseeded those used before in the package fMultivar.

The usage of the sn functions is:

```
dmsc(x,xi,Omega,alpha,dp = NULL,log = FALSE)
pmsc(x,xi,Omega,alpha,dp = NULL,...)
rmsc(n,xi,Omega,alpha,dp = NULL)
```

NOTE: The multivariate skew-normal distribution is discussed by Azzalini and Dalla Valle (1996). The `(Omega,alpha)` parametrization adopted here is the one of Azzalini and Capitanio (1999). Chapter 5 of Azzalini and Capitanio (2014) provides an extensive account, including subsequent developments. Be aware that the location vector `xi` does not represent the mean vector of the distribution. Similarly,`Omega` is not the covariance matrix of the distribution, although it is a covariance matrix.

For further details we refer to the help page in the package `sn`.

References

Azzalini, A. and Dalla Valle, A. (1996), The multivariate skew-normal distribution, Biometrika 83, 715-726.

Azzalini, A. and Capitanio, A. (1999), Statistical applications of the multivariate skew normal distribution, Journal Roy.Statist.Soc. B 61, 579-602, Full-length version available at http://arXiv.org/abs/0911.2093

Azzalini, A. with the collaboration of Capitanio, A. (2014), The Skew-Normal and Related Families, Cambridge University Press, IMS Monographs Series.

Examples

```
## Not run:
## grid2d -
   # Make 2-D Grid Coordinates:
   N <- 101
   x <- y <- seq(-3, 3, l=N)
   X <- cbind(u=grid2d(x)$x, v=grid2d(x)$y)

## Set Parameters:
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
   alpha <- c(2, -6)

## dmsc -
   # Compute skew Cauchy Density:
   z <- sn::dmsc(X, xi, Omega, alpha)
   Z <- list(x=x, y=x, z=matrix(z, ncol = length(x)))
   # Plot:
   image(Z, main ="Skew Cauchy Density")
   contour(Z, add=TRUE)
   grid(col="red")

## pmsc -
   # Compute skew Cauchy Probability:
   z <- NULL
   for (i in 1:nrow(X)) z <- c(z, sn::pmsc(X[i, ], xi, Omega, alpha)[[1]])
   Z <- list(x=x, y=x, z=matrix(z, ncol = length(x)))
   # Plot:
   image(Z, main ="Skew Cauchy Probability")
```

```
   contour(Z, add=TRUE)
   grid(col="red")

## rmsc -
   # Skew Cauchy Random Deviates:
   set.seed(4711)
   r <- sn::rmsc(10000, xi, Omega, alpha)
   plot(hexBinning(r[, 1], r[, 2]))
   # Note, we have fat tails ...

## End(Not run)
```

`mvdist-mscFit` *Multivariate Skew Cauchy Parameter Estimation*

Description

Fitting the parameters for the Multivariate Skew Cauchy Distribution.

Usage

```
mscFit(x, trace=FALSE, title = NULL, description = NULL)
```

Arguments

`x`	a matrix with "d" columns, giving the coordinates of the point(s) where the density must be evaluated.
`trace`	a logical value, should the estimation be traced? By default FALSE.
`title`	an optional project title.
`description`	an option project desctiption.

Details

This is an easy to use wrapper function using default function settings for fitting the distributional parameters in the framework of the contributed package `"sn"` written by Adelchi Azzalini.

Starting values for the estimation have not to be provided, they are automatically created.

Examples

```
## Not run:
## Load Library:
   require(sn)

## mscFit -
   # Fit Example:
   N <- 1000
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
```

```
    alpha <- c(2, -6)
    set.seed(4711)
    X <- rmsc(n=N, xi, Omega, alpha)
    ans <- mscFit(X)
    # Show fitted Parameters:
    print(ans)

    # 2-D Density Plot:
    plot(hexBinning(X[,1], X[, 2], bins = 30), main="Skew Cauchy")
    # Add Contours:
    N <- 101
    x <- seq(min(X[, 1]), max(X[, 1]), l=N)
    y <- seq(min(X[, 2]), max(X[, 2]), l=N)
    u <- grid2d(x, y)$x
    v <- grid2d(x, y)$y
    XY <- cbind(u, v)
    param <- ans@fit$dp
    Z <- matrix(dmsc(XY, param[[1]][1,], param[[2]], param[[3]]), ncol=N)
    contour(x, y, Z, add=TRUE, col="green", lwd=2)
    grid(col="brown", lty=3)

## Cut the Tails:
    CUT <- 25
    X <- X[abs(X[, 1]) <= CUT, ]
    X <- X[abs(X[, 2]) <= CUT, ]
    plot(hexBinning(X[,1], X[, 2], bins = 30), main="Skew Cauchy")
    x <- y <- seq(-CUT, CUT, l=N)
    u <- grid2d(x, y)$x
    v <- grid2d(x, y)$y
    XY <- cbind(u, v)
    param <- ans@fit$dp
    Z <- matrix(dmsc(XY, param[[1]][1,], param[[2]], param[[3]]), ncol=N)
    contour(x, y, Z, add=TRUE, col="green", lwd=2)
    grid(col="brown", lty=3)
    # Try larger cuts ...

## End(Not run)
```

`mvdist-msn` *Multivariate Skew-Normal Distribution*

Description

Density, distribution function, and random number generation for the multivariate Skew-Normal distribution.

Details

The functions to compute densities `dmsc`, probabilities `pmsc`, and to generate random numbers `rmsc` for the multivariate skew Normal distribution are available in the contributed R package `sn` (note, they are no longer builtin in fMultivar). The reason is that the performance for these functions in package sn has superseeded those used before in the package fMultivar.

The usage of the sn functions is:

dmsn(x,xi,Omega,alpha,tau = 0,dp = NULL,log = FALSE)
pmsn(x,xi,Omega,alpha,tau = 0,dp = NULL,...)
rmsn(n,xi,Omega,alpha,tau = 0,dp = NULL)

NOTE: The multivariate skew-normal distribution is discussed by Azzalini and Dalla Valle (1996). The (Omega,alpha) parametrization adopted here is the one of Azzalini and Capitanio (1999). Chapter 5 of Azzalini and Capitanio (2014) provides an extensive account, including subsequent developments. Be aware that the location vector xi does not represent the mean vector of the distribution. Similarly,Omega is not the covariance matrix of the distribution, although it is a covariance matrix.

For further details we refer to the help page in the package sn.

References

Azzalini, A. and Dalla Valle, A. (1996), The multivariate skew-normal distribution, Biometrika 83, 715-726.

Azzalini, A. and Capitanio, A. (1999), Statistical applications of the multivariate skew normal distribution, Journal Roy.Statist.Soc. B 61, 579-602, Full-length version available at http://arXiv.org/abs/0911.2093

Azzalini, A. with the collaboration of Capitanio, A. (2014), The Skew-Normal and Related Families, Cambridge University Press, IMS Monographs Series.

Examples

```
## Not run:
## Make 2-D Grid Coordinates:
   N <- 101
   x <- y <- seq(-3, 3, l=N)
   X <- cbind(u=grid2d(x)$x, v=grid2d(x)$y)

## dmsn
   # Set Parameters:
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
   alpha <- c(2, -6)
   # Compute skew Normal Density:
   z <- sn::dmsn(X, xi, Omega, alpha)
   Z <- list(x=x, y=x, z=matrix(z, ncol = length(x)))
   # Plot:
   image(Z)
   contour(Z)
   grid(col="red")

## rmsn -
   set.seed(4711)
   r <- sn::rmsn(n=5000, xi, Omega, alpha)
   plot(hexBinning(r))
   contour(Z, add=TRUE, col="darkblue", lwd=2)
```

```
   grid(col="red")

## End(Not run)
```

---

mvdist-msnFit *Multivariate Skew Normal Parameter Estimation*

Description

Fitting the parameters for the multivariate skew Normal distribution.

Usage

```
msnFit(x, trace = FALSE, title = NULL, description = NULL)
```

Arguments

x	a matrix with "d" columns, giving the coordinates of the point(s) where the density must be evaluated.
trace	a logical value, should the estimation be traced? By default FALSE.
title	an optional project title.
description	an option project desctiption.

Details

This is an easy to use wrapper function using default function settings for fitting the distributional parameters in the framework of the contributed package "sn" written by Adelchi Azzalini.

Starting values for the estimation have not to be provided, they are automatically created.

Examples

```
## Not run:
## Load Library:
   require(sn)

## msnFit -
   # Fit Example:
   N <- 1000
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
   alpha <- c(2, -6)
   set.seed(4711)
   X <- rmsn(n=N, xi, Omega, alpha)
   ans <- msnFit(X)
   print(ans)

   # 2-D Density Plot:
   plot(hexBinning(X[,1], X[, 2], bins = 30), main="Skew Normal")
```

```
# Add Contours:
N <- 101
x <- seq(min(X[, 1]), max(X[, 1]), l=N)
y <- seq(min(X[, 2]), max(X[, 2]), l=N)
u <- grid2d(x, y)$x
v <- grid2d(x, y)$y
XY <- cbind(u, v)
param <- ans@fit$estimate
Z <- matrix(dmsn(XY, param[[1]][1,], param[[2]], param[[3]]), ncol=N)
contour(x, y, Z, add=TRUE, col="green", lwd=2)
grid(col="brown", lty=3)

## End(Not run)
```

`mvdist-mst` *Multivariate Skew Student-t Distribution*

Description

Density, distribution function, and random number generation for the multivariate Skew-Student-t distribution.

Details

The functions to compute densities `dmsc`, probabilities `pmsc`, and to generate random numbers `rmsc` for the multivariate skew Student-t distribution are available in the contributed R package `sn` (note, they are no longer builtin in fMultivar). The reason is that the performance for these functions in package `sn` has superseeded those used before in the package fMultivar.

The usage of the `sn` functions is:

```
dmst(x,xi,Omega,alpha,nu = Inf,dp = NULL,log = FALSE)
pmst(x,xi,Omega,alpha,nu = Inf,dp = NULL,...)
rmst(n,xi,Omega,alpha,nu = Inf,dp = NULL)
```

NOTE: The multivariate skew-normal distribution is discussed by Azzalini and Dalla Valle (1996). The `(Omega,alpha)` parametrization adopted here is the one of Azzalini and Capitanio (1999). Chapter 5 of Azzalini and Capitanio (2014) provides an extensive account, including subsequent developments. Be aware that the location vector `xi` does not represent the mean vector of the distribution. Similarly,`Omega` is not the covariance matrix of the distribution, although it is a covariance matrix.

For further details we refer to the help page in the package `sn`.

References

Azzalini, A. and Dalla Valle, A. (1996), The multivariate skew-normal distribution, Biometrika 83, 715-726.

Azzalini, A. and Capitanio, A. (1999), Statistical applications of the multivariate skew normal distribution, Journal Roy.Statist.Soc. B 61, 579-602, Full-length version available at http://arXiv.org/abs/0911.2093

Azzalini, A. with the collaboration of Capitanio, A. (2014), The Skew-Normal and Related Families, Cambridge University Press, IMS Monographs Series.

Examples

```
## Not run:
## Make 2-D Grid Coordinates:
   N <- 101
   x <- y <- seq(-3, 3, l=N)
   X <- cbind(u=grid2d(x)$x, v=grid2d(x)$y)

## dmst -
   # Set Parameters:
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
   alpha <- c(2, -6)
   nu <- 4
   # Compute skew Student-t Density:
   z <- dmst(X, xi, Omega, alpha, nu)
   Z <- list(x=x, y=x, z=matrix(z, ncol = length(x)))
   # Plot:
   image(Z)
   contour(Z)
   grid(col="red")

## rmst -
   set.seed(4711)
   r <- rmst(n=5000, xi, Omega, alpha, nu)
   plot(hexBinning(r))
   contour(Z, add=TRUE, col="darkblue", lwd=2)
   grid(col="red")

## End(Not run)
```

`mvdist-mstFit` *Multivariate Skew Student-t Parameter Estimation*

Description

Fitting the parameters for the Multivariate Skew Student-t Distribution

Usage

```
mstFit(x, fixed.nu=NULL, trace=FALSE, title=NULL, description=NULL)
```

Arguments

x	a matrix with "d" columns, giving the coordinates of the point(s) where the density must be evaluated.

`fixed.nu`	a positive value to keep fixed the parameter nu of the Student-t distribution in the optimization process; with default value NULL, nu is estimated like the other parameters.
`trace`	a logical value, should the estimation be traced? By default FALSE.
`title`	an optional project title.
`description`	an option project desctiption.

Details

This is an easy to use wrapper function using default function settings for fitting the distributional parameters in the framework of the contributed package "sn" written by Adelchi Azzalini.

Starting values for the estimation have not to be provided, they are automatically created.

Examples

```
## Not run:
## Load Library:
   require(sn)

## mstFit -
   # Fit Example:
   N <- 1000
   xi <- c(0, 0)
   Omega <- diag(2); Omega[2,1] <- Omega[1,2] <- 0.5
   alpha <- c(2, -2)
   nu <- 4
   set.seed(4711)
   X <- rmst(n=N, xi, Omega, alpha, nu=4)
   ans <- mstFit(X)
   # Show fitted Parameters:
   print(ans)

   # 2-D Density Plot:
   plot(hexBinning(X[,1], X[, 2], bins = 30), main="Skew Student-t")
   # Add Contours:
   N <- 101
   x <- seq(min(X[, 1]), max(X[, 1]), l=N)
   y <- seq(min(X[, 2]), max(X[, 2]), l=N)
   u <- grid2d(x, y)$x
   v <- grid2d(x, y)$y
   XY <- cbind(u, v)
   param <- ans@fit$dp
   Z <- matrix(dmst(
     XY, param[[1]][1,], param[[2]], param[[3]], param[[4]]), ncol=N)
   contour(x, y, Z, add=TRUE, col="green", lwd=2)
   grid(col="brown", lty=3)

## mstFit -
   # Fit Example with fixed nu=4:
   ans <- mstFit(X, fixed.nu=4)
   # Show fitted Parameters:
```

```
    print(ans)

    # 2-D Density Plot:
    plot(hexBinning(X[,1], X[, 2], bins = 30), main="Student-t | fixed nu")
    # Add Contours:
    param <- ans@fit$dp
    Z <- matrix(dmst(
      XY, param[[1]][1,], param[[2]], param[[3]], nu=4), ncol=N)
    contour(x, y, Z, add=TRUE, col="green", lwd=2)
    grid(col="brown", lty=3)

## End(Not run)
```

`utils-adapt` *Integrator for multivariate distributions*

Description

The function is for adaptive quadrature.

Usage

```
adapt(ndim, lower, upper, functn, ...)
```

Arguments

`ndim`	the dimension of the integral. By default NUll, no longer used.
`lower`	vector of at least length `ndim` of the lower bounds on the integral.
`upper`	vector of at least length `ndim` of the upper bounds on the integral.
`functn`	an R function which should take a single vector argument and possibly some parameters and return the function value at that point. `functn` must return a single numeric value.
`...`	other parameters to be passed to the underlying function.

Value

The returned value is a list of three items:

`integral`	the value of the integral.
`error`	the estimated relative error.
`functionEvaluations`	the number of times the function was evaluated.
`returnCode`	the actual integer return code of the C routine.

Note

In 2007 the package adapt was removed from the CRAN repository, due to unclear license conditions. Nevertheless, formerly available versions can still be obtained from the CRAN archive. Package adapt used FORTRAN code from Professor Genz.

From 2007 until 2013 the package fMultivar used an builtin licensed by Professor Genz to Rmetrics. This version is still available in the current package, have a look into the folder deprecated.

2013 the contributed package cubature was added to the CRAN repository. This provides an alternative n-dimensional integration routine. We recommend to use the function adaptIntegrate directly from the package cubature which allows adaptive multivariate integration over hypercubes. It is a wrapper around the pure C, GPLed implementation by Steven G. Johnson.

Since 2014 fMultivar uses also the C Version based implementation of Johnson. The former function adapt has been replaced by a wrapper function calling adaptIntegrate. The arguments ndim, lower, upper, and functn have been remeined the same, control parameters have been adapted to the function cubature::adaptIntegrate.

Author(s)

Balasubramanian Narasimhan

References

See: http://ab-initio.mit.edu/wiki/index.php/Cubature.

Examples

```
## Check that dnorm2d is normalized:

   # Normal Density:
   density <- function(x) dnorm2d(x=x[1], y = x[2])

   # Calling Cubature:
   BIG <- c(99, 99)
   cubature::adaptIntegrate(f=density, lowerLimit=-BIG, upperLimit=BIG)
   cubature::adaptIntegrate(f=density, low=-BIG, upp=BIG, tol=1e-7)

   # Using the Wrapper:
   adapt(lower=-BIG, upper=BIG, functn=density)
   adapt(lower=-BIG, upper=BIG, functn=density, tol=1e-7)$integral
```

---

utils-binning2 *Square and Hexagonal Data Binning*

Description

Two functions which allow to create histograms due to sqaure and hexagonal binning.

Usage

```
squareBinning(x, y = NULL, bins = 30)
hexBinning(x, y = NULL, bins = 30)

## S3 method for class 'squareBinning'
plot(x, col = heat.colors(12), addPoints = TRUE,
    addRug = TRUE, ...)
## S3 method for class 'hexBinning'
plot(x, col = heat.colors(12), addPoints = TRUE,
    addRug = TRUE, ...)
```

Arguments

`addPoints`	a logical flag, should the center of mass points added to the plot?
`addRug`	a logical flag, should a rug representation be added to the plot, for details see the function `rug`.
`bins`	an integer specifying the number of bins.
`col`	color map like for the `image` function.
`x, y`	[squareBinning][hexBinning] - either two numeric vectors of equal length or if `y` is NULL, a list with entries `x`, `y`, or named data frame with `x` in the first and `y` in the second column. Note, `timeSeries` objects are also allowed as input. [plot] - an object of class `squareBinning` or `hexBinning`.
`...`	arguments to be passed.

Details

`squareBinning` does a square binning of data points, and `hexBinning` does a hexagonal binning of data points.

Value

A list with three entries, `x`, `y` and `z`, specified by an oject of class `squareBinning` or `hexBinning`.

Note, the returned value, can be directly used by the `persp()` and `contour` 3D plotting functions.

Author(s)

Diethelm Wuertz for the Rmetrics R-port.

Examples

```
## squareBinning -
   sB <- squareBinning(x = rnorm(1000), y = rnorm(1000))
   plot(sB)

## hexBinning -
```

```
hB <- hexBinning(x = rnorm(1000), y = rnorm(1000))
plot(hB)
```

---

utils-density2d *Bivariate Density Tools*

Description

Kernel density estimator and histogram counter for bivariate distributions

Usage

```
density2d(x, y = NULL, n = 20, h = NULL, limits = c(range(x), range(y)))

hist2d(x, y = NULL, n = c(20, 20))
```

Arguments

`x, y`	two vectors of coordinates of data. If `y` is NULL then `x` is assumed to be a two column matrix, where the first column contains the `x` data, and the second column the `y` data.
`n`	n - an integer specifying the number of grid points in each direction. The default value is 20. [hist2D] - In this case `n` may be a scalar or a two element vector. The default value is 20.
`h`	a vector of bandwidths for `x` and `y` directions. Defaults to normal reference bandwidth.
`limits`	the limits of the rectangle covered by the grid.

Value

`density2d` and `hist2d` return a list with three elements `$x`, `$y`, and `$z`. `x` and `y` are vectors spanning the two dimensional grid and `z` the corresponding matrix. The output can directly serve as input to the plotting functions `image`, `contour` and `persp`.

Author(s)

W.N. Venables and B.D. Ripley for the underlying kde2d function,
Gregory R. Warnes for the underlying hist2d function,
Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Warnes G.R., (2004); *The gregmisc Package*; R Reference Guide available from www.r-project.org.

Examples

```
## hist2d -
   # Normal Random Numbers:
   set.seed(4711)
   X <- rnorm2d(40000)
   # 2D Histogram Plot:
   Z <- hist2d(X)
   image(Z)
   contour(Z, add=TRUE)
```

---

utils-grid2d *Bivariate Density Tools*

Description

Grid generator for bivariate distributions.

Usage

```
grid2d(x = (0:10)/10, y = x)
```

Arguments

`x, y`	two numeric vectors defining the `x` and `y` coordinates.

Value

`grid2d` returns a list with two vectors named `$x` and `$y` spanning the grid defined by the coordinate vectors `x` and `y`.

Author(s)

Diethelm Wuertz.

Examples

```
## grid2d -
   # Create a square grid:
   x <- seq(0, 10, length = 6)
   X <- grid2d(x = x, y = x)
   cbind(X$x, X$y)
```

utils-gridding2d *Bivariate Gridded Data Sets*

Description

Functions which allow to generate bivariate gridded data sets.

Grid Data Functions:

`gridData`	generates a grid data set of class 'gridData',
`persp`	generates a perspective plot from a grid data set,
`contour`	generates a contour plot from a grid data set.

Usage

```
gridData(x = (-10:10)/10, y = x, z = outer(x, y, function(x, y) (x^2+y^2) ) )

## S3 method for class 'gridData'
persp(x, theta = -40, phi = 30, col = "steelblue",
    ticktype = "detailed", ...)
## S3 method for class 'gridData'
contour(x, addImage = TRUE, ...)
```

Arguments

`addImage` [contour] -
a logical flag indicating if an image plot should be underlayed to the contour level plot.

`x, y, z` [gridData] -
`x` and `y` are two numeric vectors of grid pounts and `z` is a numeric matrix or any other rectangular object which can be transformed by the function `as.matrix` into a matrix object.

`theta, phi, col, ticktype`
[persp] -
tailored parameters passed the perspective plot function `persp`.

`...` [contour][persp] -
additional arguments to be passed to the perspectice and countour plot functions.

Value

`gridData` -
A list with at least three entries, `x`, `y` and `z`.

The returned values, can be directly used by the `persp.gridData()` and `contour.gridData` 3D plotting methods.

Author(s)

Diethelm Wuertz for the Rmetrics R-port,
H. Akima for the Fortran Code of the Akima spline interpolation routine.

Examples

```
## gridData -
   # Grid Data Set:
   gD = gridData()
   persp(gD)
   contour(gD)
```

`utils-integrate2d` *Bivariate Integration Tools*

Description

Integrates over the unit square.

Usage

```
integrate2d(fun, error = 1.0e-5, ...)
```

Arguments

`fun`	the function to be integrated. The first argument requests the x values, the second the y values, and the remaining are reserved for additional parameters. The integration is over the unit square "[0,1]\^2".
`error`	the error bound to be achieved by the integration formula. A numeric value.
`...`	parameters passed to the function to be integrated.

Value

`integrate2d` returns a list with the `$value` of the integral over the unit square [0,1]\^2, an `$error` estimate and the number of grid `$points` used by the integration function.

Author(s)

W.N. Venables and B.D. Ripley for the underlying kde2d function,
Gregory R. Warnes for the underlying hist2d function,
Diethelm Wuertz for the Rmetrics R-port.

References

Azzalini A., (2004); *The sn Package*; R Reference Guide available from www.r-project.org.

Venables W.N., Ripley B.D., (2002); *Modern Applied Statistics with S*, Fourth Edition, Springer.

Warnes G.R., (2004); *The gregmisc Package*; R Reference Guide available from www.r-project.org.

`zzz-mvnorm` *Multivariate Normal Distribution*

Description

Alternative density, distribution function, and random generation for the multivariate Normal distribution.

Details

The multivariate distribution functions to compute densities `dmvnorm`, probabilities `pmvnorm`, and to generate random numbers `rmvnorm` are available from the contributed R package `mvtnorm`. The function `qmvnorm` computes the equicoordinate quantile function of the multivariate normal distribution for arbitrary correlation matrices based on inversion of `pmvnorm`.

```
dmvnorm(x,mean,sigma,<<...>>
pmvnorm(<<...>>)
qmvnorm(p,<<...>>)
rmvnorm(n,mean,sigma,<<...>>
```

NOTE: The function are not builtin in the package `fMultivar`. Fur details we refer to the help page of `mvnorm`.

Author(s)

Friedrich Leisch and Fabian Scheipl.

Examples

```
## Not run:
## Load Libray:
   require(mvtnorm)

## dmvnorm -
   # Multivariate Normal Density Function:
   mean <- c(1, 1)
   sigma <- matrix(c(1, 0.5, 0.5, 1), ncol=2)
   dmvnorm(x = c(0, 0),mean, sigma)

## dmvnorm -
   # Across a Grid:
   x <- seq(-4, 4, length=90)
   X <- grid2d(x)
   X <- cbind(X$x, X$y)
   # Write Density Function:
   dmvnorm. <- function(X, mean, sigma)
     matrix(apply(X, 1, dmvnorm, mean=mean, sigma=sigma), ncol=sqrt(dim(X)[1]))
   z <- dmvnorm.(X, mean, sigma)
   contour(list(x = x, y = x, z = z))
```

```
## qmvnorm -
   # Equicoordinate Quantile Function:
   qmvnorm(p = 0.95, sigma = diag(2), tail = "both")

## rmvnorm -
   # Random Numbers:
   sigma <- matrix(c(4, 2, 2, 3), ncol=2)
   x <- rmvnorm(n = 500, mean = c(1, 2), sigma = sigma)
   colMeans(x)
   var(x)
   # Next Generation:
   x <- rmvnorm(n = 500, mean = c(1, 2), sigma = sigma, method = "chol")
   colMeans(x)
   var(x)
   plot(x, cex=0.5, pch=19, col="steelblue")

## End(Not run)
```

`zzz-mvstnorm` *Obsolete Functions*

Description

Obsolete Functions: Alternative multivariate distribution and parameter estimation functions for the skew normal and skew Student-t distribution functions.

Usage

```
dmvsnorm(x, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim))
pmvsnorm(q, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim))
rmvsnorm(n, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim))

dmvst(x, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim), df=4)
pmvst(q, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim), df=4)
rmvst(n, dim=2, mu=rep(0, dim), Omega=diag(dim), alpha=rep(0, dim), df=4)

mvFit(x, method = c("snorm", "st"), fixed.df = NA,
    title = NULL, description = NULL, trace = FALSE)
```

Arguments

`x, q` the vector of quantiles, a matrix with "dim" columns.

`n` the number of desired observations.

`dim` the dimension, by default the bivariate case is considered where `dim=2`

`mu, Omega, alpha, df`
: `mu` is a numeric vector of length "dim" representing the location parameter of the distribution, `Omega` is a symmetric positive-definite matrix of dimension "d" timesd "d", `alpha` is a numeric vector which regulates the the slant of the density, `df` a positive value representing the degrees of freedom.

`method`	selects the type of distribution function, either `"snorm"` which is the default, or `"st"`.
`fixed.df`	set to a positive value to keep fixed the parameter `nu` of the skew student-t distribution in the optimization process; with default value NULL, i.e. `nu` is estimated like the other parameters.
`title`	an optional project title.
`description`	an option project desctiption.
`trace`	a logical, should the estimation be traced?

Details

The former implementations have been replaced by wrpper functions calling functions from the package `"sn"`.

Value

`dm*` gives the density, `pm*` gives the distribution function, and `rm*` generates `n` random deviates of dimension `dim`

`mvFit` returns an object of class `fDISTFEED`, see package `fBasics`.

Examples

```
## Not run:
## Load Libray:
   require(mvtnorm)

## [dr]mvsnorm -
   dmvsnorm(rnorm2d(100))
   rmvsnorm(100)

## [dr]mvst -
   dmvst(rt2d(100))
   rmvst(100)

## End(Not run)
```

zzz-mvt *Multivariate Student-t Distribution*

Description

Alternative density, distribution function, and random generation for the multivariate Student-t distribution.

Details

The functions to compute densities `dmvt`, probabilities `pmvt`, and to generate random numbers `rmvt` are available from the contributed R package `mvtnorm`. The function `qmvt` computes the equicoordinate quantile function of the multivariate normal distribution for arbitrary correlation matrices based on inversion of `pmvt`.

```
dmvt(x,delta,sigma,df,<<...>>)
pmvt(<<...>>)
rmvt(n,sigma,df,delta,<<...>>
```

NOTE: The function are not builtin in the package `fMultivar`. Fur details we refer to the help page of `mvnorm`.

Author(s)

Alan Genz, Frank Bretz, Tetsuhisa Miwa, Xuefei Mi, Friedrich Leisch, Fabian Scheipl, Bjoern Bornkamp, Torsten Hothorn.

References

McNeil, A. J., Frey, R., and Embrechts, P. (2005), *Quantitative Risk Management: Concepts, Techniques, Tools*, Princeton University Press.

Examples

```
## Not run:
## Load Libray:
   require(mvtnorm)

## dmvt -
   # basic evaluation
   dmvt(x = c(0,0), sigma = diag(2))

## dmvt | dmvnorm -
   # check behavior for df=0 and df=Inf
   x <- c(1.23, 4.56)
   mu <- 1:2
   Sigma <- diag(2)
   x0 <- dmvt(x, delta = mu, sigma = Sigma, df = 0) # default log = TRUE!
   x8 <- dmvt(x, delta = mu, sigma = Sigma, df = Inf) # default log = TRUE!
   xn <- dmvnorm(x, mean = mu, sigma = Sigma, log = TRUE)
   stopifnot(identical(x0, x8), identical(x0, xn))

## rmvt -
   # X ~ t_3(0, diag(2))
   x <- rmvt(100, sigma = diag(2), df = 3) # t_3(0, diag(2)) sample
   plot(x)

## rmvt -
   # X ~ t_3(mu, Sigma)
   n <- 1000
```

```
   mu <- 1:2
   Sigma <- matrix(c(4, 2, 2, 3), ncol=2)
   set.seed(271)
   x <- rep(mu, each=n) + rmvt(n, sigma=Sigma, df=3)
   plot(x)

## rmvt -
   # Note that the call rmvt(n, mean=mu, sigma=Sigma, df=3) does *not*
   # give a valid sample from t_3(mu, Sigma)! [and thus throws an error]
   try(rmvt(n, mean=mu, sigma=Sigma, df=3))

## rmvnorm -
   # df=Inf correctly samples from a multivariate normal distribution
   set.seed(271)
   x <- rep(mu, each=n) + rmvt(n, sigma=Sigma, df=Inf)
   set.seed(271)
   x. <- rmvnorm(n, mean=mu, sigma=Sigma)
   stopifnot(identical(x, x.))

## End(Not run)
```

Index