Package 'mnormt'

July 23, 2025

Version 2.1.1

Date 2022-09-26

Title The Multivariate Normal and t Distributions, and Their Truncated Versions

Author Adelchi Azzalini [aut, cre],
Alan Genz [aut] (most Fortran code),
Alan Miller [ctb] (Fortran routine PHI),
Michael J. Wichura [ctb] (Fortran routine PHINV),
G. W. Hill [ctb] (Fortran routine STDINV),
Yihong Ge [ctb] (Fortran routines BNVU and MVBVU).

Maintainer Adelchi Azzalini <adelchi.azzalini@unipd.it>

Depends R (>= 2.2.0)

Description Functions are provided for computing the density and the distribution function of d-dimensional normal and ``t'' random variables, possibly truncated (on one side or two sides), and for generating random vectors sampled from these distributions, except sampling from the truncated ``t''. Moments of arbitrary order of a multivariate truncated normal are computed, and converted to cumulants up to order 4.
Probabilities are computed via non-Monte Carlo methods; different routines are used in the case d=1, d=2, d=3, d>3, if d denotes the dimensionality.

License GPL-2 | GPL-3

URL http://azzalini.stat.unipd.it/SW/Pkg-mnormt/

NeedsCompilation yes

Encoding UTF-8

Repository CRAN

Date/Publication 2022-09-26 10:10:06 UTC

Contents

`mnormt-package` *The 'mnormt' package: summary information*

Description

Functions are provided for computing the density and the distribution function of d-dimensional normal and *t* random variables, possibly truncated (on one side or two sides, with componentwise choice), and for generating random vectors sampled from these distributions, except sampling from the truncated *t*. Moments of arbitrary order of a truncated normal are computed, and converted to cumulants up to order 4.

Details

Probabilities are computed via non-Monte Carlo methods; different routines are used in the case `d=1`, `d=2`, `d=3`, `d>2`, if `d` denotes the dimensionality.

Licence

This package and its documentation are usable under the terms of the "GNU General Public License" version 3 or version 2, as you prefer; a copy of them is available from https://www.R-project.org/Licenses/.

Author(s)

Adelchi Azzalini (R code and package creation) and Alan Genz (Fortran code, see the references below; this code incorporates routines of other authors).

References

Genz, A. (1992). Numerical Computation of Multivariate Normal Probabilities. *J. Computational and Graphical Statist.* **1**, 141-149.

Genz, A. (1993). Comparison of methods for the computation of multivariate normal probabilities. *Computing Science and Statistics*, **25**, 400-405.

Genz, A.: Fortran code downloaded in 2006 from the author web page, located at https://www.math.wsu.edu/faculty/genz/software/software.html, as of 2020-06-01.

mnorm *The multivariate normal distribution*

Description

The probability density function, the distribution function and random number generation for a d-dimensional multivariate normal (Gaussian) random variable.

Usage

```
dmnorm(x, mean = rep(0, d), varcov, log = FALSE)
pmnorm(x, mean = rep(0, d), varcov, ...)
rmnorm(n = 1, mean = rep(0, d), varcov, sqrt=NULL)
sadmvn(lower, upper, mean, varcov, maxpts = 2000*d, abseps = 1e-06, releps = 0)
```

Arguments

x	either a vector of length d or a matrix with d columns representing the coordinates of the point(s) where the density must be evaluated; see also 'Details' for restrictions on d.
mean	either a vector of length d, representing the mean value, or (except for rmnorm) a matrix whose rows represent different mean vectors; in the matrix case, only allowed for dmnorm and pmnorm, its dimensions must match those of x.
varcov	a symmetric positive-definite matrix representing the variance-covariance matrix of the distribution; a vector of length 1 is also allowed (in this case, d=1 is set).
sqrt	if not NULL (default value is NULL), a square root of the intended varcov matrix; see 'Details' for a full description.
log	a logical value (default value is FALSE); if TRUE, the logarithm of the density is computed.
...	arguments passed to sadmvn, among maxpts, abseps, releps.
n	the number of (pseudo) random vectors to be generated.
lower	a numeric vector of lower integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed.
upper	a numeric vector of upper integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed.
maxpts	the maximum number of function evaluations (default value: 2000*d).
abseps	absolute error tolerance (default value: 1e-6).
releps	relative error tolerance (default value: 0).

Details

The dimension d cannot exceed 20 for pmnorm and sadmvn. If this threshold is exceeded, NA is returned.

The function pmnorm works by making a suitable call to sadmvn if d>3, or to ptriv.nt if d=3, or to biv.nt.prob if d=2, or to pnorm if d=1. The R functions sadmvn, ptriv.nt and biv.nt.prob are, in essence, interfaces to underlying FORTRAN 77 routines by Alan Genz; see the references below. These routines use adaptive numerical quadrature and other non-random type techniques.

If sqrt=NULL (default value), the working of rmnorm involves computation of a square root of varcov via the Cholesky decomposition. If a non-NULL value of sqrt is supplied, it is assumed that it represents a matrix, R say, such that $R'R$ represents the required variance-covariance matrix of the distribution; in this case, the argument varcov is ignored. This mechanism is intended primarily for use in a sequence of calls to rmnorm, all sampling from a distribution with fixed variance matrix; a suitable matrix sqrt can then be computed only once beforehand, avoiding that the same operation is repeated multiple times along the sequence of calls; see the examples below. Another use of sqrt is to supply a different form of square root of the variance-covariance matrix, in place of the Cholesky factor.

For efficiency reasons, rmnorm does not perform checks on the supplied arguments.

If, after setting the same seed value to set.seed, two calls to rmnorm are made with the same arguments except that one generates n1 vectors and the other n2 vectors, with n1<n2, then the n1 vectors of the first call coincide with the initial n2 vectors of the second call.

Value

dmnorm returns a vector of density values (possibly log-transformed); pmnorm returns a vector of probabilities, possibly with attributes on the accuracy in case x is a vector; sadmvn return a single probability with attributes giving details on the achieved accuracy; rmnorm returns a matrix of n rows of random vectors, or a vector in case n=1 or d=1.

Note

The attributes error and status of the probability returned by pmnorm and sadmvn indicate whether the function had a normal termination, achieving the required accuracy. If this is not the case, re-run the function with a higher value of maxpts

Author(s)

FORTRAN 77 code of SADMVN, package mvtdstpack.f, package tvpack and most auxiliary functions by Alan Genz; some additional auxiliary functions by people referred to within his programs; interface to R and additional R code (for dmnormt, rmnormt, etc.) by Adelchi Azzalini.

References

Genz, A. (1992). Numerical Computation of multivariate normal probabilities. *J. Computational and Graphical Statist.*, **1**, 141-149.

Genz, A. (1993). Comparison of methods for the computation of multivariate normal probabilities. *Computing Science and Statistics*, **25**, 400-405.

Genz, A.: Fortran 77 code downloaded in 2005 and again in 2007 from his web-page, whose URL as of 2020-04-28 was https://www.math.wsu.edu/faculty/genz/software/software.html

See Also

dnorm, dmt, biv.nt.prob, ptriv.nt, plot_fxy for plotting examples

Examples

```
x <- seq(-2, 4, length=21)
y <- cos(2*x) + 10
z <- x + sin(3*y)
mu <- c(1,12,2)
Sigma <- matrix(c(1,2,0,2,5,0.5,0,0.5,3), 3, 3)
f <- dmnorm(cbind(x,y,z), mu, Sigma)
f0 <- dmnorm(mu, mu, Sigma)
p1 <- pmnorm(c(2,11,3), mu, Sigma)
p2 <- pmnorm(c(2,11,3), mu, Sigma, maxpts=10000, abseps=1e-10)
p <- pmnorm(cbind(x,y,z), mu, Sigma)
#
set.seed(123)
x1 <- rmnorm(5, mu, Sigma)
set.seed(123)
x2 <- rmnorm(5, mu, sqrt=chol(Sigma)) # x1=x2
eig <- eigen(Sigma, symmetric = TRUE)
R <- t(eig$vectors %*% diag(sqrt(eig$values)))
for(i in 1:50) x <- rmnorm(5, mu, sqrt=R)
#
p <- sadmvn(lower=c(2,11,3), upper=rep(Inf,3), mu, Sigma) # upper tail
#
p0 <- pmnorm(c(2,11), mu[1:2], Sigma[1:2,1:2])
p1 <- biv.nt.prob(0, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
p2 <- sadmvn(lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
c(p0, p1, p2, p0-p1, p0-p2)
#
p1 <- pnorm(0, 1, 3)
p2 <- pmnorm(0, 1, 3^2)
```

`mom.mtruncnorm` *Moments and other quantities of a (possibly) truncated multivariate normal distribution*

Description

Moments up to the specified orders of a possibly truncated d-dimensional normal distribution; the distribution must be non-degenerate. Each component variable can be truncated on one side (to the left or to the right) or on two sides or not truncated. After the initial stage, cumulants up to the fourth order and other quantities are computed, provided all moments of the required order had been computed in the first stage.

Usage

```
mom.mtruncnorm(powers=4, mean, varcov, lower, upper, cum = TRUE, ...)
```

Arguments

powers	a vector of non-negative integer values representing the required order of moments for each component variable, or a single such value, in which case this value is repeated for all component variables.
mean	a d-length vector representing the mean value of the pre-truncation normal random variable. If d=length(mean), see 'Details' for restrictions on d.
varcov	a symmetric positive definite matrix with dimensions (d,d) representing the variance matrix of the pre-truncation normal random variable.
lower	a d-vector representing the lower truncation values of the component variables; -Inf values are allowed. If missing, it is set equal to rep(-Inf, d).
upper	a d-vector representing the upper truncation values of the component variables; Inf values are allowed. If missing, it is set equal to rep(Inf, d).
cum	a logical value; if codeTRUE (default value), cumulants are other quantities are computed up to the minimum between the fourth order and the maximum possible order, given the available moments.
...	additional arguments passed to sadmvn; see 'Details' for a description.

Details

The maximal value of d is 20. If this threshold is exceeded, NAs are returned. The constraint originates from the underlying function sadmvn.

This function makes use of two workhorses, recintab and mom2cum, providing a user-friendly interface to these more basic tools. The first function computes an array of raw moments of the truncated normal; the second function translates them into cumulants and other quantities such as the Mardia's measures of skewness and kurtosis, unless cum=FALSE. See the documentation of these two underlying functions for additional information about the arguments and the returned quantities. The argument ... is passed, via recintab, to sadmvn for regulation of its working.

Not all d component variables need to be truncated. In fact, the function works also with no truncated components (just omit lower and upper), although for this case there exist known formulae to do the job.

Value

A list with the following components:

mom	an array with raw moments as produced by recintab, followed by normalization; see its documentation for a description.
cum1	the vector of first-order cumulants, AKA the expected value or the mean value, which will be there provided cum=TRUE and all elements of powers are not less than 1.
order2, ...	additional lists with higher order terms up to order 4; these lists only exist when the available moments are of sufficiently high order. See mom2cum for a more detailed description.

Author(s)

Adelchi Azzalini

See Also

recintab, mom2cum, sadmvn

Examples

```
mu <- c(1, -0.5, 0)
Sigma <- toeplitz(1/(1:3))
lower <- c(-Inf, -3, -4)
upper <- c(1.5, Inf, 2)
m <- mom.mtruncnorm(1, mu, Sigma, lower, upper)
print(m$cum1)
#
m <- mom.mtruncnorm(3, mu, Sigma, lower, upper)
print(m$order3$gamma1.marginal)
print(m$order3$gamma1.Mardia)
#
#--
# Example 2 of Leppard & Tallis (1989, Appl.Stat. vol.38, p.547)
truncp <- c(0, 1, 2)
U <- c(0, 0, 0)
V <- 0.5*(diag(3) + matrix(1, 3, 3))
m <- mom.mtruncnorm(2, U, V, truncp)
print(m$cum1, digits=9)
print(m$order2$cum2, digits=9)
```

---

mom2cum *Conversion of an array of moments to cumulants*

Description

Given an array of moments of a multivariate distribution, the corresponding cumulants up to the 4th order and other connected quantities are computed, notably the Mardia's measures of multivariate skewness and kurtosis

Usage

```
mom2cum(mom)
```

Arguments

mom	an array whose entries are assumed to represent moments of a multivariate distribution; see 'Details' for an extended description.

Details

The structure of the input array mom is of type M/M[1] where M represents the output from function recintab. For a d-dimensional random variable, mom is a k-fold d-dimensional array, where k is the highest order of moments being considered; see the documentation of recintab for a more detailed description. However, it is not necessary that mom originates from recintab; the moments can refer to any distribution, as long as mom has the appropriate structure and content.

Also, it is not necessary that all entries of mom are there; values not required for the processing can be left as NA. For computing cumulants of order k, say, we only need cross moments whose exponents add up to k or less.

Conversion from moments to cumulants is performed by using formulae (2.7) of McCullagh (1987). See also ρ_{23}^2 in his (2.15) and ρ_4 in (2.16) for computing the Mardia's (1970, 1974) measures of multivariate skewness and kurtosis.

In some cases, the function may report inconsistencies detected in the argument mom. A typical origin of this situation is in numerical inaccuracies of the returned value of recintab, as explained in more detail in its documentation. When detected, cases of these sort are flagged in the returned $message string, and a warning message is issued. The absence of such string does not represent a guarantee of perfect input.

Value

In the multivariate case, a list with the following elements, provided moments of the required order are available, up to the maximal order 4.

cum1	the d-vector of first-order cumulants, AKA the expected value or the mean value; this will be there if mom contains all moments of order 1.
order2	a list with the following components: m2, the (d,d) matrix of second order moments; cum2, the (d,d) matrix of second order cumulants, AKA the variance-covariance matrix, the variance matrix, the covariance matrix, the dispersion matrix; conc.matrix, the concentration matrix, that is, the inverse of cum2; log.det.cum2, the logarithm of the determinant of cum2.
order3	a list with the following components: m3, array of third order moments, having dimension (d,d,d); cum3, array of third order cumulants, having dimension (d,d,d); m3.marginal, vector of third order marginal moments; centr.mom3.marginal, vector of third order marginal central moments; gamma1.marginal, vector of third order marginal standardized cumulants; gamma1.Mardia, the Mardia measure of multivariate skewness; beta1.Mardia, the Mardia measure of multivariate skewness, again.
order4	a list with the following components: m4, array of fourth order moments, with dimension (d,d,d,d); cum4, array of fourth order cumulants, with dimension (d,d,d,d); m4.marginal, vector of fourth order marginal moments; centr.mom4.marginal, vector of fourth order marginal central moments; gamma2.marginal, vector of fourth order marginal standardized cumulants; gamma2.Mardia, the Mardia measure of multivariate kurtosis, $\gamma_{2,d}$; beta2.Mardia, the Mardia measure of multivariate kurtosis, $\beta_{2,d}$.
message	possibly, a character string indicating that some inconsistency has been detected in the argument mom; see 'Details'.

In the univariate case a list with elements:

`cum`	a vector of cumulants,
`centr.mom`	a vector of central moments,
`std.cum`	a vector with the third and the fourth standardized cumulants (when enough moments are available), representing common measures of skewness and kurtosis.
`message`	possibly, a character string indicating that some inconsistency has been detected in the argument `mom`; see 'Details'.

Note

In the case of a multivariate truncated normal distribution, a user does not need to call this function; `mom.mtruncnorm` provides a more convenient interface for the same computations. The present function needs to be called only if the array `mom` represents the moments of some other distribution.

Author(s)

Adelchi Azzalini

References

Mardia, K. V. (1970). Measures of multivariate skewness and kurtosis with applications *Biometrika*, 57, 519-530.

Mardia, K. V. (1974). Applications of some measures of multivariate skewness and kurtosis in testing normality and robustness studies. *Sankhya ser.B*, 36, 115-128.

McCullagh, Peter (1987). *Tensor Methods in Statistics*. Chapman & Hall, London.

See Also

`recintab`

Examples

```
mu <- c(1, -0.5)
Sigma <- toeplitz(1/(1:2))
low <- c(-Inf, -3)
hi <- c(1.5, Inf)
mom <- recintab(c(3,3), low, hi, mu, Sigma)
cum <- mom2cum(mom)
print(cum$order3$gamma1.marginal)
print(cum$order3$gamma1.Mardia)
```

mt *The multivariate Student's* t *distribution*

Description

The probability density function, the distribution function and random number generation for a d-dimensional Student's *t* random variable.

Usage

```
dmt(x, mean = rep(0, d), S, df=Inf, log = FALSE)
pmt(x, mean = rep(0, d), S, df=Inf, ...)
rmt(n = 1, mean = rep(0, d), S, df=Inf, sqrt=NULL)
sadmvt(df, lower, upper, mean, S, maxpts = 2000*d, abseps = 1e-06, releps = 0)
biv.nt.prob(df, lower, upper, mean, S)
ptriv.nt(df, x, mean, S)
```

Arguments

x	either a vector of length d or (for dmt and pmt) a matrix with d columns representing the coordinates of the point(s) where the density must be evaluated; see also 'Details'.
mean	either a vector of length d, representing the location parameter (equal to the mean vector when df>1), or (for dmt and pmt) a matrix whose rows represent different mean vectors; in the matrix case, its dimensions must match those of x.
S	a symmetric positive definite matrix with dimensions (d,d) representing the scale matrix of the distribution, such that S*df/(df-2) is the variance-covariance matrix when df>2; a vector of length 1 is also allowed (in this case, d=1 is set).
df	the degrees of freedom. For rmt, it must be a positive real value or Inf. For all other functions, it must be a positive integer or Inf. A value df=Inf is translated to a call to a suitable function for the the multivariate normal distribution. See 'Details' for its effect for the evaluation of distribution functions and other probabilities.
log	a logical value(default value is FALSE); if TRUE, the logarithm of the density is computed.
sqrt	if not NULL (default value is NULL), a square root of the intended scale matrix S; see 'Details' for a full description.
...	arguments passed to sadmvt, among maxpts, absrel, releps.
n	the number of random vectors to be generated
lower	a numeric vector of lower integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed.
upper	a numeric vector of upper integration limits of the density function; must be of maximal length 20; +Inf and -Inf entries are allowed

maxpts	the maximum number of function evaluations (default value: 2000*d)
abseps	absolute error tolerance (default value: 1e-6).
releps	relative error tolerance (default value: 0).

Details

The dimension d cannot exceed 20 for pmt and sadmvt. If this threshold is exceeded, NA is returned.

The functions sadmvt, ptriv.mt and biv.nt.prob are interfaces to Fortran 77 routines by Alan Genz, available from his web page; they makes use of some auxiliary functions whose authors are indicated in the Fortran code itself. The routine sadmvt uses an adaptive integration method. If df=3, a call to pmt activates a call to ptriv.nt which is specific for the trivariate case, and uses Genz's Fortran code tvpack.f; see Genz (2004) for the background methodology. A similar fact takes place when df=2 with function biv.nt.prob; note however that the underlying Fortran code is taken from mvtdstpack.f, not from tvpack.f. If pmt is called with d>3, this is converted into a suitable call to sadmvt.

If sqrt=NULL (default value), the working of rmt involves computation of a square root of S via the Cholesky decomposition. If a non-NULL value of sqrt is supplied, it is assumed that it represents a square root of the scale matrix, otherwise represented by S, whose value is ignored in this case. This mechanism is intended primarily for use in a sequence of calls to rmt, all sampling from a distribution with fixed scale matrix; a suitable matrix sqrt can then be computed only once beforehand, avoiding that the same operation is repeated multiple times along the sequence of calls. For examples of use of this argument, see those in the documentation of rmnorm. Another use of sqrt is to supply a different form of square root of the scale matrix, in place of the Cholesky factor.

For efficiency reasons, rmt does not perform checks on the supplied arguments.

Value

dmt returns a vector of density values (possibly log-transformed); pmt and sadmvt return a single probability with attributes giving details on the achieved accuracy, provided x of pmnorm is a vector; rmt returns a matrix of n rows of random vectors, or a vector in case n=1 or d=1.

Note

The attributes error and status of the probability returned by sadmvt and by pmt (the latter only if x is a vector and d>2) indicate whether the function had a normal termination, achieving the required accuracy. If this is not the case, re-run the function with a higher value of maxpts.

Author(s)

FORTRAN 77 code of SADMVT, MVTDSTPACK, TVPACK and many auxiliary functions by Alan Genz; some additional auxiliary functions by people referred to within his programs; interface to R and additional R code (for dmt, rmt etc.) by Adelchi Azzalini.

References

Genz, A.: Fortran 77 code in files mvt.f, mvtdstpack.f and codetvpack, downloaded in 2005 and again in 2007 from his webpage, whose URL as of 2020-06-01 is https://www.math.wsu.edu/faculty/genz/software/software.html

Genz, A. (2004). Numerical computation of rectangular bivariate and trivariate normal and *t* probabilities. *Statistics and Computing* 14, 251-260.

Dunnett, C.W. and Sobel, M. (1954). A bivariate generalization of Student's *t*-distribution with tables for certain special cases. *Biometrika* 41, 153–169.

See Also

dt, rmnorm for use of argument sqrt, plot_fxy for plotting examples

Examples

```
x <- seq(-2,4,length=21)
y <- 2*x+10
z <- x+cos(y)
mu <- c(1,12,2)
Sigma <- matrix(c(1,2,0,2,5,0.5,0,0.5,3), 3, 3)
df <- 4
f  <- dmt(cbind(x,y,z), mu, Sigma,df)
p1 <- pmt(c(2,11,3), mu, Sigma, df)
p2 <- pmt(c(2,11,3), mu, Sigma, df, maxpts=10000, abseps=1e-8)
x  <- rmt(10, mu, Sigma, df)
p  <- sadmvt(df, lower=c(2,11,3), upper=rep(Inf,3), mu, Sigma) # upper tail
#
p0 <- pmt(c(2,11), mu[1:2], Sigma[1:2,1:2], df=5)
p1 <- biv.nt.prob(5, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
p2 <- sadmvt(5, lower=rep(-Inf,2), upper=c(2, 11), mu[1:2], Sigma[1:2,1:2])
c(p0, p1, p2, p0-p1, p0-p2)
```

---

mtruncnorm *The multivariate truncated normal distribution*

Description

The probability density function, the distribution function and random number generation for the d-dimensional truncated normal (Gaussian) random variable.

Usage

```
dmtruncnorm(x, mean, varcov, lower, upper, log = FALSE, ...)
pmtruncnorm(x, mean, varcov, lower, upper, ...)
rmtruncnorm(n, mean, varcov, lower, upper, start, burnin=5, thinning=1)
```

Arguments

x	either a vector of length d or a matrix with d columns,representing the coordinates of the point(s) where the density must be evaluated. Here we denote d=ncol(varcov); see 'Details' for restrictions.
mean	a d-vector representing the mean value of the pre-truncation normal distribution.

varcov	a symmetric positive definite matrix with dimensions (d,d) representing the variance matrix of the pre-truncation normal distribution.
lower	a d-vector representing the lower truncation values of the component variables; -Inf values are allowed. If missing, it is set equal to rep(-Inf, d).
upper	a d-vector representing the upper truncation values of the component variables; Inf values are allowed. If missing, it is set equal to rep(Inf, d).
log	a logical value (default value is FALSE); if TRUE, the logarithm of the density is computed.
...	arguments passed to sadmvn, among maxpts, abseps, releps.
n	the number of (pseudo) random vectors to be generated.
start	an optional vector of initial values; see 'Details'.
burnin	the number of burnin iterations of the Gibbs sampler (default: 5); see 'Details'.
thinning	a positive integer representing the thinning factor of the internally generated Gibbs sequence (default: 1); see 'Details'.

Details

For dmtruncnorm and pmtruncnorm, the dimension d cannot exceed 20. If this threshold is exceeded, NAs are returned. The constraint originates from the underlying function sadmvn.

If d>1, rmtruncnorm uses a Gibbs sampling scheme as described by Breslaw (1994) and by Kotecha & Djurić (1999), Detailed algebraic expressions are provided by Wilhelm (2022). After some initial settings in R, the core iteration is performed by a compiled FORTRAN 77 subroutine, for numerical efficiency.

If the start vector is not supplied, the mean value of the truncated distribution is used. This choice should provide a good starting point for the Gibbs iteration, which explains why the default value for the burnin stage is so small. Since successive random vectors generated by a Gibbs sampler are not independent, which can be a problem in certain applications. This dependence is typically ameliorated by generating a larger-than-required number of random vectors, followed by a 'thinning' stage; this can be obtained by setting the thinning argument larger that 1. The overall number of generated points is burnin+n*thinning, and the returned object is formed by those with index in burnin+(1:n)*thinning.

If d=1, the values are sampled using a non-iterative procedure, essentially as in equation (4) of Breslaw (1994), except that in this case the mean and the variance do not refer to a conditional distribution, but are the arguments supplied in the calling statement.

Value

dmtruncnorm and pmtruncnorm return a numeric vector; rmtruncnorm returns a matrix, unless either n=1 or d=1, in which case it returns a vector.

Author(s)

Adelchi Azzalini

References

Breslaw, J.A. (1994) Random sampling from a truncated multivariate normal distribution. *Appl. Math. Lett.* vol.7, pp.1-6.

Kotecha, J.H. and Djurić, P.M. (1999). Gibbs sampling approach for generation of truncated multivariate Gaussian random variables. In *ICASSP'99: Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol.3, pp.1757-1760. doi:10.1109/ICASSP.1999.756335.

Wilhelm, S. (2022). Gibbs sampler for the truncated multivariate normal distribution. Vignette of R package https://cran.r-project.org/package=tmvtnorm, version 1.5.

See Also

plot_fxy for additional plotting examples, sadmvn for regulating accuracy via ...

Examples

```
# example with d=2
m2 <- c(0.5, -1)
V2 <- matrix(c(3, 3, 3, 6), 2, 2)
low <- c(-1, -2.8)
up <- c(1.5, 1.5)
# plotting truncated normal density using 'dmtruncnorm' and 'contour' functions
plot_fxy(dmtruncnorm, xlim=c(-2, 2), ylim=c(-3, 2), mean=m2, varcov=V2,
      lower=low, upper=up,  npt=101)
set.seed(1)
x <-  rmtruncnorm(n=500, mean=m2, varcov=V2, lower=low, upper=up)
points(x, cex=0.2, col="red")
#------
# example with d=1
set.seed(1)
low <- -4
hi <- 3
x <- rmtruncnorm(1e5, mean=2, varcov=5, lower=low, upper=hi)
hist(x, prob=TRUE, xlim=c(-8, 12), main="Truncated univariate N(2, sqrt(5))")
rug(c(low, hi), col=2)
x0 <- seq(-8, 12, length=251)
pdf <- dnorm(x0, 2, sqrt(5))
p <- pnorm(c(low, hi), 2, sqrt(5))
lines(x0, pdf/diff(p), col=4, lty=2)
lines(x0, dmtruncnorm(x0, 2, 5, low, hi), col=2, lwd=2)
```

---

mtrunct *The multivariate truncated Student's* t *distribution*

Description

The probability density function and the distribution function of the multivariate truncated Student's *t* distribution

Usage

```
dmtrunct(x, mean, S, df, lower, upper, log = FALSE, ...)
pmtrunct(x, mean, S, df, lower, upper, ...)
```

Arguments

x	either a vector of length d or a matrix with d columns, where d=ncol(S), giving the coordinates of the point(s) where the density must be evaluated.
mean	either a vector of length d, representing the location parameter (equal to the mean vector when df>1) of the pre-truncation distribution or a matrix whose rows represent different mean vectors; in the matrix case, its dimensions must match those of x.
S	a symmetric positive-definite matrix representing the scale matrix, such that S*df/(df-2) is the variance-covariance matrix of the pre-truncation distribution when df>2.
df	degrees of freedom; it must be a positive integer
lower	a vector representing the lower truncation values of the component variables; -Inf values are allowed. If missing, it is set equal to rep(-Inf, d).
upper	a vector representing the upper truncation values of the component variables; Inf values are allowed. If missing, it is set equal to rep(Inf, d).
log	a logical value (default value is FALSE); if TRUE, the logarithm of the density is computed.
...	arguments passed to sadmvt, among maxpts, absrel, releps.

Details

The dimension d cannot exceed 20.

Value

a numeric vector

Author(s)

Adelchi Azzalini

See Also

sadmvt for regulating accuracy

Examples

```
m2 <- c(0.5, -1)
V2 <- matrix(c(1.5, -1.75, -1.75, 3), 2, 2)
lower <- a <- c(-1, -2.5)
upper <- b <- c(2, 1)
set.seed(1)
points <- matrix(runif(10, -3, 3), nrow=5, ncol=2)
```

```
pdf <- dmtrunct(points, mean=m2, S=V2, df=4, lower, upper)
cdf <- pmtrunct(points, mean=m2, S=V2, df=4, lower, upper)
```

---

pd.solve *Inverse of a symmetric positive-definite matrix*

Description

The inverse of a symmetric positive-definite matrix and its log-determinant

Usage

```
pd.solve(x, silent = FALSE, log.det=FALSE)
```

Arguments

x	a symmetric positive-definite matrix.
silent	a logical value which indicates the action to take in case of an error. If silent==TRUE and an error occurs, the function silently returns a NULL value; if silent==FALSE (default), an error generates a stop with an error message.
log.det	a logical value to indicate whether the log-determinant of x is required (default is FALSE).

Details

The function checks that x is a symmetric positive-definite matrix. If an error is detected, an action is taken which depends on the value of the argument silent.

Value

the inverse matrix of x; if log.det=TRUE, this inverse has an attribute which contains the logarithm of the determinant of x.

Author(s)

Adelchi Azzalini

Examples

```
x <- toeplitz(rev(1:4))
x.inv <- pd.solve(x)
print(x.inv %*% x)
x.inv <- pd.solve(x, log.det=TRUE)
logDet <- attr(x.inv, "log.det")
print(abs(logDet - determinant(x, logarithm=TRUE)$modulus))
```

plot_fxy *Plotting a function of two variables*

Description

Plot a real-valued function `f` evaluated on a grid of points of the Cartesian plane, possibly with parameters specified by `...`. The type of graphical display can be regulated by selecting the plotting function among a set of available options.

Usage

```
plot_fxy(f, xlim, ylim, ..., npt=51, grf, grpar)
```

Arguments

`f`	either a function or a character string with the name of a real-valued function whose first argument represents the coordinates of points where `f` is evaluated; see 'Details' for additional information.
`xlim`	either a vector of abscissae where the `f` must be evaluated, or a length-two vector with the endpoints of such an interval, in which case `npt[1]` equally spaced points will be considered.
`ylim`	either a vector of ordinates where the `f` must be evaluated, or a length-two vector with the endpoints of such an interval, in which case `npt[2]` equally spaced points will be considered.
`...`	additional parameters to be supplied to `f`; these must be named as expected by the specification of `f`.
`npt`	either an integer value or a two-element integer vector with the number of equally-spaced points, within the endpoints of `xlim` and `ylim`, used to set up the grid of points where `f` is evaluated; default value: `51`. When a single value is supplied, this is expanded into a length-2 vector. If `length(xlim)>2` and `length(ylim)>2`, `npt` is ignored.
`grf`	an optional character string with the name of the function which produces the graphical display, selectable among `"contour"`, `"filled.contour"`, `"persp"`, `"image"` of package `graphics`; if `grf` is unset, `"contour"` is used.
`grpar`	an optional character string with arguments supplied to the selected `grf` function, with items separated by `,` as in a regular call.

Details

Function `f` will be called with the first argument represented by a two-column matrix, where each row represents a point of the grid on the Cartesian plane identified by `xlim` and `ylim`; this set of coordinates is stored in matrix `pts` of the returned list. If present, arguments supplied as `...` are also passed to `f`. It is assumed that `f` accepts this type of call.

The original motivation of `plot_fxy` was to plot instances of bivariate probability density functions specified by package `mnormt`, but it can be used for plotting any function fulfilling the above requirements, as illustrated by some of the examples below.

Value

an invisible list with the following components:

`x`	a vector of coordinates on the x axis
`y`	a vector of coordinates on the y axis
`pts`	a matrix of dimension `(npt[1]*npt[2],2)` with the coordinates of the evaluation points (x, y)
`f.values`	the vector of `f` values at the `pts` points.

See Also

`contour`, `filled.contour`, `persp`, `image`

Examples

```
Sigma <- matrix(c(1,1,1,2), 2, 2)
mean <- c(0, -1)
xlim <- c(-3, 5)
ylim <- c(-5, 3)
#
# multivariate normal density, contour-level plot
gp <- 'col="blue", nlevels=6, main="bivariate normal density"'
u <- plot_fxy(dmnorm, xlim, ylim, mean=mean, varcov=Sigma,  grpar=gp)
cat(str(u))
#---
# multivariate normal density, filled-contour plot
plot_fxy(dmnorm,  xlim, ylim, mean=mean, varcov=Sigma,grf="filled.contour")
#---
# multivariate normal density, perspective plot
gp <- "theta = 10, phi = 25, r = 2.5"
plot_fxy(dmnorm, xlim, ylim, mean=mean, varcov=Sigma, grf="persp", grpar=gp)
#---
# multivariate Student's "t" density;
# the xlim argument passed to function 'grf' overrides the earlier xlim;
# xlim and ylim can be placed after the arguments of 'f', if one prefers so
grp <- 'xlim=c(-1, 3)'
plot_fxy(dmt, mean=mean, S=Sigma, df=8, xlim, ylim, npt=101,
        grf="filled.contour", grpar=grp)
#---
# multivariate truncated normal density, 'image' plot
low <- c(-3, -5)
hi <- c(1, 0)
plot_fxy(dmtruncnorm, mean=mean, varcov=Sigma, lower=low, upper=hi,
        xlim, ylim, npt=81, grf="image")
#---
# multivariate truncated normal distribution function, 'image' plot;
# hence not a density function
low <- c(-3, -5)
hi <- c(1, 0)
v <- plot_fxy(pmtruncnorm, mean=mean, varcov=Sigma, lower=low, upper=hi,
              xlim, ylim, npt=c(61, 81), grf="image")
#---
# a different sort of 'f' function (lbeta), not a component of this package
```

```
funct <- function(z) lbeta(a=z[,1], b=z[,2])
plot_fxy(funct, xlim=c(0.1, 2), ylim=c(0.1, 2), npt=41,
         grpar='main="function log-beta(a,b)", xlab="a", ylab="b"')
```

`recintab` *Moments of arbitrary order of a (possibly) truncated multivariate normal variable*

Description

Produces an array with the moments up to specified orders of a (possibly) truncated multivariate normal distribution. Each component variable can be truncated on one side (to the left or to the right) or on two sides or not truncated.

Usage

```
recintab(kappa, a, b, mu, S, ...)
```

Arguments

`kappa`	a vector of non-negative integer values representing the required order of moments for each component variable.
`a`	a vector representing the lower truncation values of the component variables; `-Inf` values are allowed.
`b`	a vector representing the upper truncation values of the component variables; `Inf` values are allowed.
`mu`	a vector representing the mean value of the pre-truncation normal random variable.
`S`	a symmetric positive-definite matrix representing the variance matrix of the pre-truncation normal random variable.
`...`	parameters passed to `sadmvn`; see the 'Details'.

Details

The maximal dimension of the multivariate normal variable is 20. If this threshold is exceeded `NA`s are returned.

This function is the R translation of the Matlab function with the same name belonging to the package `ftnorm`, which is associated to the paper of Kan and Robotti (2017). The Matlab package `ftnorm` has been downloaded from http://www-2.rotman.utoronto.ca/~kan/research.htm, on 2020-04-23.

The function returns an array, `M` say, whose entries represent integrals of type $\int_a^b x^\kappa f(x)dx$, where $f(x)$ denotes the d-dimensional normal density function. Typically, interest is in the scaled array `M/M[1]` whose entries represent the moments of the truncated distribution.

The algorithm is based on a recursion starting from the integral of the normal distribution over the specified hyper-rectangle. This integral is evaluated by `sadmvn`, whose tuning parameters `maxpts`, `abseps`, `releps` can be regulated via the `...` argument.

Value

In the multivariate case, for an input vector `kappa=c(k1,...,kd)`, the functions returns an array of dimension `c((k1+1),...,(kd+1))` whose entries represent integrals described in section 'Details'. In other words, the array element `M[i+1, j+1, k+1,...]` contains the *unnormalized* cross moment of order `(i, j, k,...)`; this must be divided by `M[1]` to obtain the regular cross moment.

In the univariate case, a vector is returned with similar meaning.

Warning

Although the underlying algorithm is exact in principle, the actual computation hinges crucially on the initial integration of the multivariate normal density over the truncation hyper-cube. This integration may result in numerical inaccuracies, whose amount depends on the supplied arguments. Moreover, the recursion employed by the algorithm propagates the initial error to other terms.

When problematic cases have been processed by the original Matlab function, the same issues have occurred, up to minor variations.

Instances of such errors may be detected when the array `M/M[1]` is passed to `mom2cum`, but there is no guarantee that all such problems are detected.

Note

This function is not intended for direct call by a user, at least in commonly encountered situations. Function `mom.mtruncnorm` represents a more user-friendly tool.

Author(s)

Original Matlab code by Raymond Kan and Cesare Robotti, porting to R by Adelchi Azzalini.

References

Kan, Raymond and Robotti, Cesare (2017). On moments of folded and truncated multivariate normal distributions. *Journal of Computational and Graphical Statistics*, 26, 930-934, DOI: 10.1080/10618600.2017.1322092

Leppard, P. and Tallis, G. M. (1989). Algorithm AS249: Evaluation of the mean and covariance of the truncated multinormal distribution *Applied Statistics* 38, 543-553)

See Also

`mom.mtruncnorm` for a more user-friendly function, `mom2cum` for transformation to cumulants, `sadmvn` for regulating accuracy if `d>2`

Examples

```
mu <- c(1, -0.5, 0)
Sigma <- toeplitz(1/(1:3))
low <- c(-Inf, -3, -4)
hi <- c(1.5, Inf, 2)
M <- recintab(c(2,3,1), low, hi, mu, Sigma)
M/M[1]
# cross-moments up to order 2 for X1, up to the 3 for X2, up to 1 for X3,
# if the components of the trivariate variable are denoted (X1,X2,X3)
```

```
#--
# Example 2 of Leppard & Tallis (1989, Appl.Stat. vol.38, p.547)
truncp <- c(0, 1, 2)
U <- c(0, 0, 0)
V <- 0.5*(diag(3) + matrix(1, 3, 3))
M <- recintab(c(2,2,2), truncp, rep(Inf,3), U, V)
mom <- M/M[1]
EX <-  c(mom[2,1,1], mom[1,2,1], mom[1,1,2])
print(EX, digits=9)
EX2 <- matrix(c(
          mom[3,1,1], mom[2,2,1], mom[2,1,2],
          mom[2,2,1], mom[1,3,1], mom[1,2,2],
          mom[2,1,2], mom[1,2,2], mom[1,1,3]),
          3, 3, byrow=TRUE)
varX <- EX2 - outer(EX ,EX)
print(varX, digits=9)
```

`sample_Mardia_measures` *The Mardia measures of multivariate skewness and kurtosis for a given sample*

Description

Given a multivariate sample, the Mardia measures of skewness and kurtosis are computed, along with their *p*-values for testing normality

Usage

```
sample_Mardia_measures(data, correct = FALSE)
```

Arguments

`data`	a data matrix
`correct`	(logical) if `correct=TRUE`, the 'corrected' sample variance matrix is used, otherwise the 'uncorrected' version is used (default)

Details

For a given a data matrix, the multivariate measures of skewness and kurtosis introduced by Mardia (1970, 1974) are computed, along with some associated quantities. We follow the notation of the 1974 paper.

If `n` denotes the number of complete cases, the condition `n>3` is required for numerical computation. Clearly, a much larger `n` is required for meaningful statistical work.

The sample variance matrix S appearing in (2.2) and (2.4) is computed here (in the dafault setting) with the n denominator, at variance from the commonly employed `n-1` denominator. With this definition of S, one obtains the same numerical outcome of the example on p.127 of Mardia (1974).

The approximate observed significance levels for testing normality, `p.b1` and `p.b2`, are computed using expressions (5.5) and (5.6) in Section 5 of Mardia (1974). For `p.b2`, the condition `(n-d-1)>0` is required, where `d` denotes the number of variables.

Value

A named vector with the following components:

`b1`	the measure of asymmetry as given in (2.2)
`b2`	the measure of kurtosis as given in (2.4)
`g1`	the measure of asymmetry as given in (2.10)
`g2`	the measure of kurtosis as given in (2.11)
`p.b1`	observed significance level of `b1`
`p.b2`	observed significance level of `b2`
`n`	The number of complete cases in the input data matrix

where the quoted formulae are those of Mardia (1974).

Author(s)

Adelchi Azzalini

References

Mardia, K. V. (1970). Measures of multivariate skewness and kurtosis with applications *Biometrika*, 57, 519-530.

Mardia, K. V. (1974). Applications of some measures of multivariate skewness and kurtosis in testing normality and robustness studies. *Sankhya ser.B*, 36, 115-128.

Examples

```
set.seed(1)
x <- rmnorm(100, mean=1:3, varcov=toeplitz(1/(1:3)))
sample_Mardia_measures(x)
```

Index