Package: cmvnorm (via r-universe)

September 10, 2024

Type Package

Title The Complex Multivariate Gaussian Distribution

Version 1.0-9

Depends quadform (>= 0.0-2), emulator (>= 1.2-21)

Suggests knitr

Enhances mvtnorm

Imports elliptic

Maintainer Robin K. S. Hankin <hankin.robin@gmail.com>

Description Various utilities for the complex multivariate Gaussian distribution and complex Gaussian processes.

VignetteBuilder knitr

License GPL-2

URL https://github.com/RobinHankin/cmvnorm

BugReports https://github.com/RobinHankin/cmvnorm/issues

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

RemoteUrl https://github.com/robinhankin/cmvnorm

RemoteRef HEAD

RemoteSha 5aad5eec08e887985fd3d18a982b6aaa718f1994

Contents

`cmvnorm-package` *The Complex Multivariate Gaussian Distribution*

Description

Various utilities for the complex multivariate Gaussian distribution and complex Gaussian processes.

Details

The DESCRIPTION file: This package was not yet installed at build time.

Index: This package was not yet installed at build time.

Generalizing the real multivariate Gaussian distribution to the complex case is not straightforward but one common approach is to replace the real symmetric variance matrix with a Hermitian positive-definite matrix. The **cmvnorm** package provides some functionality for the resulting density function.

Author(s)

Robin K. S. Hankin [aut, cre] (<https://orcid.org/0000-0001-5982-0415>)

Maintainer: Robin K. S. Hankin <hankin.robin@gmail.com>

References

- N. R. Goodman 1963. "Statistical analysis based on a certain multivariate complex Gaussian distribution". *The Annals of Mathematical Statistics*. 34(1): 152–177
- R. K. S. Hankin 2015. "The complex multivariate Gaussian distribution". *R News*, volume 7, number 1.

Examples

```
S1 <- 4+diag(5)
S2 <- S1
S2[1,5] <- 4+1i
S2[5,1] <- 4-1i   # Hermitian


rcmvnorm(10,sigma=S1)
rcmvnorm(10,mean=rep(1i,5),sigma=S2)

dcmvnorm(rep(1,5),sigma=S2)
```

`corr_complex` *Complex Gaussian processes*

Description

Various utilities for investigating complex Gaussian processes

Usage

```
corr_complex(z1, z2 = NULL, distance.function = complex_CF, means =
NULL, scales = NULL, pos.def.matrix = NULL)
complex_CF(z1,z2, means, pos.def.matrix)
scales.likelihood.complex(pos.def.matrix, scales, means,  zold, z,
               give_log = TRUE, func = regressor.basis)
interpolant.quick.complex(x, d, zold, Ainv, scales = NULL, pos.def.matrix = NULL,
               means=NULL, func = regressor.basis, give.Z = FALSE,
               distance.function = corr_complex, ...)
```

Arguments

`z, z1, z2` Points in C^n

`distance.function`
Function giving the (complex) covariance between two points in C^n

`means, pos.def.matrix, scales`
In function `complex_CF()`, the mean and covariance matrix of the distribution whose characteristic function is used to give the covariance matrix; `scales` is used to specify the diagonal of `pos.def.matrix` if the off-diagonal elements are zero

`zold, d, give_log, func, x, Ainv, give.Z, ...`
Direct analogues of the arguments in `interpolant()` and `scales.likelihood()` in the **emulator** package

Details

- Function `complex_CF()` returns a (slightly reparameterized) characteristic function of a complex Gaussian distribution. The covariance is given by

$$c(\mathbf{t}) = \exp(i\mathrm{Re}(\mathbf{t}^*\mu) - \mathbf{t}^*B\mathbf{t})$$

  where $\mathbf{t} = \mathbf{x} - \mathbf{x}'$ is interpreted as the distance between two observations, μ is the mean of the distribution (which is in general a complex vector), and B a positive-definite matrix.

- Function `corr_complex()` is the complex analogue of `corr.matrix()`. It returns a matrix with entry (i, j) equal to the covariance of the process at observation i and observation j, or $\mathrm{cov}(\eta(\mathbf{x}_i), \eta(\mathbf{x}_j))$. The elements are calculated by `complex_CF()`.

  This function includes only a single method, that of nested calls to `apply()`. I could not figure out how to generalize method 1 of `corr.matrix()` to the complex case.

- Function `scales.likelihood.complex()` is a complex version of `scales.likelihood()` which takes a positive definite matrix and a mean. The formula used is

$$(\sigma^2)^{-(n-q)}|A|^{-1}|H^*A^{-1}H|^{-1}$$

. Here and elsewhere, A^* means the complex conjugate of the transpose.

- Function `interpolant.quick.complex()` is a complex version of `interpolant.quick()`.

$$\mathbf{h}(\mathbf{x})^*\hat{\beta} + \mathbf{t}(\mathbf{x})^*A^{-1}(\mathbf{y} - H\hat{\beta})$$

This is the complex version of Oakley's equation 2.30 or Hankin's equation 5.

More details are given in the package vignette.

Author(s)

Robin K. S. Hankin

References

- Hankin, R. K. S. 2005. "Introducing BACCO, an R bundle for Bayesian Analysis of Computer Code Output", *Journal of Statistical Software*, 14(15)
- J. Oakley 1999. *Bayesian uncertainty analysis for complex computer codes*, PhD thesis, University of Sheffield.

Examples

```
complex_CF(c(1,1i),c(1,-1i),means=c(1i,1i),pos.def.matrix=diag(2))

V <- latin.hypercube(7,2,complex=TRUE)

cm <- c(1,1+1i)                      # "complex mean"
cs <- matrix(c(2,1i,-1i,1),2,2)   # "complex scales"
tb <- c(1,1i,1-1i)                     # "true beta"

A <- corr_complex(V,means=cm,pos.def.matrix=cs)
Ainv <- solve(A)
z <- drop(rcmvnorm(n=1,mean=regressor.multi(V) %*% tb, sigma=A))


betahat.fun(V,Ainv,z)    # should be close to 'tb'

#scales.likelihood.complex(cs,cm,V,z)   # log-likelihood evaluated true parameters



interpolant.quick.complex(x=0.1i+V[1:3,],d=z,zold=V,Ainv=Ainv,pos.def.matrix=cs,means=cm)
```

isHermitian — *Is a Matrix Hermitian?*

Description

Returns TRUE if a matrix is Hermitian or Hermitian positive-definite

Usage

```
isHermitian(x, tol = 100 * .Machine$double.eps)
ishpd(x,tol= 100 * .Machine$double.eps)
zapim(x,tol= 100 * .Machine$double.eps)
```

Arguments

x	A square matrix
tol	Tolerance for numerical scruff

Details

Functions `isHermitian()` and `ishpd()` return a Boolean, indicating whether the argument is Hermitian or Hermitian positive definite respectively. Function `zapim()` zaps small imaginary parts of a vector, returning real if all elements are so zapped.

Author(s)

Robin K. S. Hankin

Examples

```
v <- 2^(1:30)
zapim(v+1i*exp(-v))


ishpd(matrix(c(1,0.1i,-0.1i,1),2,2))   # should be TRUE
isHermitian(matrix(c(1,3i,-3i,1),2,2)) # should be TRUE
ishpd(rcwis(6,2))                      # should be TRUE
```

`Mvcnorm` *Multivariate complex Gaussian density and random deviates*

Description

Density function and a random number generator for the multivariate complex Gaussian distribution.

Usage

```
rcnorm(n)
dcmvnorm(z, mean, sigma, log = FALSE)
rcmvnorm(n, mean = rep(0, nrow(sigma)), sigma = diag(length(mean)),
    method = c("svd", "eigen", "chol"),
    tol= 100 * .Machine$double.eps)
```

Arguments

`z`	Complex vector or matrix of quantiles. If a matrix, each row is taken to be a quantile
`n`	Number of observations
`mean`	Mean vector
`sigma`	Covariance matrix, Hermitian positive-definite
`tol`	numerical tolerance term for verifying positive definiteness
`log`	In `dcmvnorm()`, Boolean with default `FALSE` meaning to return the Gaussian density function, and `TRUE` meaning to return the logarithm
`method`	Specifies the decomposition used to determine the positive-definite matrix square root of `sigma`. Possible methods are eigenvalue decomposition ("`eigen`", default), and singular value decomposition ("`svd`")

Details

Function `dcmvnorm()` is the density function of the complex multivariate normal (Gaussian) distribution:

$$p(\mathbf{z}) = \frac{\exp\left(-\mathbf{z}^*\Gamma\mathbf{z}\right)}{|\pi\Gamma|}$$

Function `rcnorm()` is a low-level function designed to generate observations drawn from a standard complex Gaussian. Function `rcmvnorm()` is a user-friendly wrapper for this.

Author(s)

Robin K. S. Hankin

References

N. R. Goodman 1963. "Statistical analysis based on a certain multivariate complex Gaussian distribution". *The Annals of Mathematical Statistics*. 34(1): 152–177

Examples

```
S <- quadform::cprod(rcmvnorm(3,mean=c(1,1i),sigma=diag(2)))

rcmvnorm(10,sigma=S)
rcmvnorm(10,mean=c(0,1+10i),sigma=S)


# Now try and estimate the mean (viz 1,1i) and variance (S) from a
#  random sample:


n <- 101
z <- rcmvnorm(n,mean=c(0,1+10i),sigma=S)
xbar <- colMeans(z)
Sbar <- cprod(sweep(z,2,xbar))/n
```

`setreal` *Manipulate real or imaginary components of an object*

Description

Manipulate real or imaginary components of an object

Usage

```
Im(x) <- value
Re(x) <- value
```

Arguments

`x`	Complex-valued object
`value`	Real-valued object

Author(s)

Robin K. S. Hankin

Examples

```
A <- matrix(c(1,0.1i,-0.1i,1),2,2)
Im(A) <- Im(A)*3
Re(A) <- matrix(c(5,2,2,5),2,2)
```

---

var *Variance and standard deviation of complex vectors*

Description

Complex generalizations of `stats::sd()` and `stats::var()`

Usage

```
var(x, y=NULL, na.rm=FALSE,use)
sd(x, na.rm=FALSE)
```

Arguments

`x, y`	Complex vector or matrix
`na.rm`	Boolean with default `FALSE` meaning to leave `NA` values present and `TRUE` meaning to remove them
`use`	Ignored

Details

Intended to be broadly compatible with `stats::sd()` and `stats::var()`.

If given real values, `var()` and `sd()` return the variance and standard deviation as per ordinary real analysis. If given complex values, return the complex generalization in which Hermitian transposes are used.

If `z` is a complex matrix, `var(z)` returns the variance of the rows.

These functions use $n-1$ on the denominator purely for consistency with `stats::var()` (for the record, I disagree with the rationale for $n-1$).

Author(s)

Robin K. S. Hankin

Examples

```
sd(rcnorm(10)) # imaginary component suppressed by zapim()

var(rcmvnorm(1e5,mean=c(0,0)))
```

`wishart` *The complex Wishart distribution*

Description

Returns an observation drawn from the complex Wishart distribution. To sample from the inverse complex Wishart distribution (or indeed the complex inverse Wishart distribution), use `solve(rcwis(...))`.

Usage

```
rcwis(n, S)
```

Arguments

`n`	Integer; degrees of freedom
`S`	Variance matrix. If an integer, use `diag(nrow=S)`

Value

Returns a (semi-) positive definite Hermitian matrix the same size as argument `S`

Note

The first argument of `rcwis()` is `n`, by universal statistics convention. But in the R world, functions returning random observations (such as `runif()`) generally reserve argument `n` for the number of observations to return. Although `rchisq()` uses `df` for the number of degrees of freedom.

Author(s)

Robin K. S. Hankin

Examples

```
rcwis(10,2)
eigen(rcwis(7,3),TRUE,TRUE)   # all positive
eigen(rcwis(3,7),TRUE,TRUE)   # 4 positive, 3 zero

rcwis(10,rcwis(10,3))
```

Index