Package 'matrixsampling'

August 25, 2019

Type Package

Title Simulations of Matrix Variate Distributions

Version 2.0.0

Date 2019-08-24

Author Stéphane Laurent

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Provides samplers for various matrix variate distributions: Wishart, inverse-Wishart, normal, t, inverted-t, Beta type I, Beta type II, Gamma, confluent hypergeometric. Allows to simulate the noncentral Wishart distribution without the integer restriction on the degrees of freedom.

License GPL-3

URL https://github.com/stla/matrixsampling

BugReports https://github.com/stla/matrixsampling/issues

Encoding UTF-8

LazyData true

Imports stats, keep

ByteCompile true

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-08-24 22:00:02 UTC

R topics documented:

`rinvwishart` *Inverse-Wishart sampler*

Description

Samples the inverse-Wishart distribution.

Usage

```
rinvwishart(n, nu, Omega, epsilon = 0, checkSymmetry = TRUE)
```

Arguments

`n`	sample size, a positive integer
`nu`	degrees of freedom, must satisfy `nu > p-1`, where `p` is the dimension (the order of `Omega`)
`Omega`	scale matrix, a positive definite real matrix
`epsilon`	threshold to force invertibility in the algorithm; see Details
`checkSymmetry`	logical, whether to check the symmetry of `Omega`; if `FALSE`, only the upper triangular part of `Omega` is used

Details

The inverse-Wishart distribution with scale matrix Ω is defined as the inverse of the Wishart distribution with scale matrix Ω^{-1} and same number of degrees of freedom. The argument `epsilon` is a threshold whose role is to guarantee the invertibility of the sampled Wishart distributions. See Details in `rwishart`.

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Examples

```
nu <- 6
p <- 3
Omega <- toeplitz(p:1)
IWsims <- rinvwishart(10000, nu, Omega)
dim(IWsims) # 3 3 10000
apply(IWsims, 1:2, mean) # approximately Omega/(nu-p-1)
# the epsilon argument:
IWsims <- tryCatch(rinvwishart(10000, nu=p+0.001, Omega),
                   error=function(e) e)
IWsims <- tryCatch(rinvwishart(10000, nu=p+0.001, Omega, epsilon=1e-8),
                   error=function(e) e)
```

`rmatrixbeta` *Matrix Beta sampler*

Description

Samples a matrix Beta (type I) distribution.

Usage

```
rmatrixbeta(n, p, a, b, Theta1 = NULL, Theta2 = NULL, def = 1,
  checkSymmetry = TRUE)
```

Arguments

`n`	sample size, a positive integer
`p`	dimension, a positive integer
`a, b`	parameters of the distribution, positive numbers with constraints given in Details
`Theta1`	numerator noncentrality parameter, a positive semidefinite real matrix of order `p`; setting it to `NULL` (default) is equivalent to setting it to the zero matrix
`Theta2`	denominator noncentrality parameter, a positive semidefinite real matrix of order `p`; setting it to `NULL` (default) is equivalent to setting it to the zero matrix
`def`	1 or 2, the definition used; see Details
`checkSymmetry`	logical, whether to check the symmetry of `Theta1` and `Theta2`

Details

A Beta random matrix U is defined as follows. Take two independent Wishart random matrices $S_1 \sim \mathcal{W}_p(2a, I_p, \Theta_1)$ and $S_2 \sim \mathcal{W}_p(2b, I_p, \Theta_2)$.

- **definition 1**: $U = (S_1 + S_2)^{-\frac{1}{2}} S_1 (S_1 + S_2)^{-\frac{1}{2}}$
- **definition 2**: $U = S_1^{\frac{1}{2}} (S_1 + S_2)^{-1} S_1^{\frac{1}{2}}$

In the central case, the two definitions yield the same distribution. Under definition 2, the Beta distribution is related to the Beta type II distribution by $U \sim V(I+V)^{-1}$.

Parameters a and b are positive numbers that satisfy the following constraints:

- if both Theta1 and Theta2 are the null matrix, a+b > (p-1)/2; if a < (p-1)/2, it must be half an integer; if b < (p-1)/2, it must be half an integer
- if Theta1 is not the null matrix, a >= (p-1)/2; if b < (p-1)/2, it must be half an integer
- if Theta2 is not the null matrix, b >= (p-1)/2; if a < (p-1)/2, it must be half an integer

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Warning

Definition 2 requires the calculation of the square root of $S_1 \sim \mathcal{W}_p(2a, I_p, \Theta_1)$ (see Details). While S_1 is always positive semidefinite in theory, it could happen that the simulation of S_1 is not positive semidefinite, especially when a is small. In this case the calculation of the square root will return NaN.

Note

The matrix variate Beta distribution is usually defined only for $a > (p-1)/2$ and $b > (p-1)/2$. In this case, a random matrix U generated from this distribution satisfies $0 < U < I$. For an half integer $a \leq (p-1)/2$, it satisfies $0 \leq U < I$ and $rank(U) = 2a$. For an half integer $b \leq (p-1)/2$, it satisfies $0 < U \leq I$ and $rank(I-U) = 2b$.

Examples

```
Bsims <- rmatrixbeta(10000, 3, 1, 1)
dim(Bsims) # 3 3 10000
```

---

rmatrixbetaII *Matrix Beta II sampler*

Description

Samples a matrix Beta type II distribution.

Usage

```
rmatrixbetaII(n, p, a, b, Theta1 = NULL, Theta2 = NULL, def = 1,
  checkSymmetry = TRUE)
```

Arguments

n	sample size, a positive integer
p	dimension, a positive integer
a, b	parameters of the distribution, positive numbers with constraints given in Details
Theta1	numerator noncentrality parameter, a positive semidefinite real matrix of order p; setting it to NULL (default) is equivalent to setting it to the zero matrix
Theta2	denominator noncentrality parameter, a positive semidefinite real matrix of order p; setting it to NULL (default) is equivalent to setting it to the zero matrix
def	1 or 2, the definition used; see Details
checkSymmetry	logical, whether to check the symmetry of Theta1 and Theta2

Details

A Beta type II random matrix V is defined as follows. Take two independent Wishart random matrices $S_1 \sim \mathcal{W}_p(2a, I_p, \Theta_1)$ and $S_2 \sim \mathcal{W}_p(2b, I_p, \Theta_2)$.

- **definition 1**: $V = S_2^{-\frac{1}{2}} S_1 S_2^{-\frac{1}{2}}$
- **definition 2**: $V = S_1^{\frac{1}{2}} S_2^{-1} S_1^{\frac{1}{2}}$

In the central case, the two definitions yield the same distribution. Under definition 2, the Beta type II distribution is related to the Beta distribution by $V \sim U(I-U)^{-1}$.

Parameters a and b are positive numbers that satisfy the following constraints:

- in any case, b > (p-1)/2
- if Theta1 is the null matrix and a < (p-1)/2, then a must be half an integer
- if Theta1 is not the null matrix, a >= (p-1)/2

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Warning

The issue described in the **Warning** section of rmatrixbeta also concerns rmatrixbetaII.

Note

The matrix variate Beta distribution of type II is usually defined only for $a > (p-1)/2$ and $b > (p-1)/2$. In this case, a random matrix V generated from this distribution satisfies $V > 0$. For an half integer $a \leq (p-1)/2$, it satisfies $V \geq 0$ and $rank(V) = 2a$.

Examples

```
Bsims <- rmatrixbetaII(10000, 3, 1, 1.5)
dim(Bsims) # 3 3 10000
```

`rmatrixCHIIkind2` *Sampler of the matrix variate type II confluent hypergeometric kind two distribution*

Description

Samples the matrix variate type II confluent hypergeometric kind two distribution.

Usage

```
rmatrixCHIIkind2(n, nu, alpha, beta, theta = 1, p)
```

Arguments

`n`	sample size, a positive integer
`nu`	shape parameter, a positive number; if `nu < (p-1)/2`, then `nu` must be a half integer
`alpha, beta`	shape parameters; `alpha > (p-1)/2`, `beta < nu + 1`
`theta`	scale parameter, a positive number
`p`	dimension (order of the sampled matrices), an integer greater than or equal to one

Value

A numeric three-dimensional array; simulations are stacked along the third dimension.

References

A. K. Gupta & D. K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, Boca Raton (2000).

Examples

```
nu <- 5; alpha <- 13; beta <- 4; theta <- 2; p <- 2
sims <- rmatrixCHIIkind2(50000, nu, alpha, beta, theta, p)
# simulations of the trace
trsims <- apply(sims, 3, function(X) sum(diag(X)))
mean(trsims)
p * theta * nu * (nu+(p+1)/2-beta) / (alpha+nu+(p+1)/2-beta)
```

`rmatrixCHIkind2` *Sampler of the matrix variate type I confluent hypergeometric kind two distribution*

Description

Samples the matrix variate type I confluent hypergeometric kind two distribution.

Usage

```
rmatrixCHIkind2(n, nu, alpha, beta, theta = 1, p)
```

Arguments

`n`	sample size, a positive integer
`nu`	shape parameter, a positive number; if `nu < (p-1)/2`, then `nu` must be a half integer
`alpha, beta`	shape parameters; `alpha > nu + (p-1)/2`, `beta < nu + 1`
`theta`	scale parameter, a positive number
`p`	dimension (order of the sampled matrices), an integer greater than or equal to one

Value

A numeric three-dimensional array; simulations are stacked along the third dimension.

References

A. K. Gupta & D. K. Nagar. *Matrix Variate Distributions*. Chapman & Hall/CRC, Boca Raton (2000).

Examples

```
nu <- 5; alpha <- 13; beta <- 4; theta <- 2; p <- 2
sims <- rmatrixCHIkind2(50000, nu, alpha, beta, theta, p)
# simulations of the trace
trsims <- apply(sims, 3, function(X) sum(diag(X)))
mean(trsims)
p * theta * nu * (nu+(p+1)/2-beta) / (alpha-nu-(p+1)/2)
```

`rmatrixCHkind1` *Sampler of the matrix variate confluent hypergometric kind one distribution*

Description

Samples the matrix variate confluent hypergometric kind one distribution.

Usage

```
rmatrixCHkind1(n, nu, alpha, beta, theta = 1, Sigma = NULL, p,
  checkSymmetry = TRUE)
```

Arguments

`n`	sample size, a positive integer
`nu`	shape parameter, a positive number; if `nu < (p-1)/2`, where `p` is the dimension (the order of `Sigma`), then `nu` must be a half integer
`alpha, beta`	shape parameters with the following constraints: `b = a` or `b > a > nu + (p-1)/2`
`theta`	scale parameter, a positive number
`Sigma`	scale matrix, a symmetric positive definite matrix, or `NULL` for the identity matrix of order `p`
`p`	if `Sigma` is `NULL`, this sets `Sigma` to the identity matrix of order `p`; ignored if `Sigma` is not `NULL`
`checkSymmetry`	logical, whether to check that `Sigma` is a symmetric positive definite matrix

Value

A numeric three-dimensional array; simulations are stacked along the third dimension.

Note

For `alpha = beta`, this is the matrix variate Gamma distribution with parameters `nu`, `theta`, `Sigma`.

References

Gupta & al. Properties of Matrix Variate Confluent Hypergeometric Function Distribution. *Journal of Probability and Statistics* vol. 2016, Article ID 2374907, 12 pages, 2016.

Examples

```
nu <- 5; alpha <- 10; beta <- 12; theta <- 2; p <- 3; Sigma <- toeplitz(3:1)
CHsims <- rmatrixCHkind1(10000, nu, alpha, beta, theta, Sigma)
# simulations of the trace
sims <- apply(CHsims, 3, function(X) sum(diag(X)))
mean(sims)
theta * nu * (nu-beta+(p+1)/2) / (nu-alpha+(p+1)/2) * sum(diag(Sigma))
```

`rmatrixgamma` *Matrix Gamma sampler*

Description

Samples a matrix Gamma distribution.

Usage

```
rmatrixgamma(n, nu, theta, Sigma = NULL, p, checkSymmetry = TRUE)
```

Arguments

`n`	sample size, a positive integer
`nu`	shape parameter, a positive number; if `nu < (p-1)/2`, where `p` is the dimension (the order of `Sigma`), then `nu` must be a half integer
`theta`	scale parameter, a positive number
`Sigma`	scale matrix, a symmetric positive definite matrix, or `NULL` for the identity matrix of order `p`
`p`	if `Sigma` is `NULL`, this sets `Sigma` to the identity matrix of order `p`; ignored if `Sigma` is not `NULL`
`checkSymmetry`	logical, whether to check that `Sigma` is a symmetric positive definite matrix

Details

This is the distribution of $\frac{\theta}{2}S$ where $S \sim \mathcal{W}_p(2\nu, \Sigma)$.

Value

A numeric three-dimensional array; simulations are stacked along the third dimension.

References

Gupta & al. Properties of Matrix Variate Confluent Hypergeometric Function Distribution. *Journal of Probability and Statistics* vol. 2016, Article ID 2374907, 12 pages, 2016.

Examples

```
nu <- 3; theta <- 4; Sigma <- toeplitz(2:1)
Gsims <- rmatrixgamma(10000, nu, theta, Sigma)
apply(Gsims, c(1,2), mean) # should be nu * theta * Sigma
nu * theta * Sigma
```

`rmatrixit` *Matrix inverted-t sampler*

Description

Samples the matrix inverted-t distribution.

Usage

```
rmatrixit(n, nu, M, U, V, checkSymmetry = TRUE, keep = TRUE)
```

Arguments

`n`	sample size, a positive integer
`nu`	degrees of freedom, any positive number or an integer strictly greater than `1-nrow(M)`
`M`	mean matrix, without constraints
`U`	columns covariance matrix, a positive semidefinite matrix of order equal to `nrow(M)`
`V`	rows covariance matrix, a positive semidefinite matrix of order equal to `ncol(M)`
`checkSymmetry`	logical, whether to check the symmetry of `U` and `V`
`keep`	logical, whether to return an array with class **keep**

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Examples

```
nu <- 0
m <- 2
p <- 3
M <- matrix(1, m, p)
U <- toeplitz(m:1)
V <- toeplitz(p:1)
ITsims <- rmatrixit(10000, nu, M, U, V)
dim(ITsims) # 2 3 10000
apply(ITsims, 1:2, mean) # approximates M
vecITsims <- t(apply(ITsims, 3, function(X) c(t(X))))
round(cov(vecITsims),2) # approximates 1/(nu+m+p-1) * kronecker(U,V)
```

`rmatrixnormal` *Matrix normal sampler*

Description

Samples the matrix normal distribution.

Usage

```
rmatrixnormal(n, M, U, V, checkSymmetry = TRUE, keep = TRUE)
```

Arguments

`n`	sample size, a positive integer
`M`	mean matrix, without constraints
`U`	columns covariance matrix, a positive semidefinite matrix of order equal to `nrow(M)`
`V`	rows covariance matrix, a positive semidefinite matrix of order equal to `ncol(M)`
`checkSymmetry`	logical, whether to check the symmetry of `U` and `V`
`keep`	logical, whether to return an array with class **keep**

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Examples

```
m <- 3
p <- 2
M <- matrix(1, m, p)
U <- toeplitz(m:1)
V <- toeplitz(p:1)
MNsims <- rmatrixnormal(10000, M, U, V)
dim(MNsims) # 3 2 10000
apply(MNsims, 1:2, mean) # approximates M
vecMNsims <- t(apply(MNsims, 3, function(X) c(t(X))))
round(cov(vecMNsims)) # approximates kronecker(U,V)
```

`rmatrixt` *Matrix t sampler*

Description

Samples the matrix t-distribution.

Usage

```
rmatrixt(n, nu, M, U, V, checkSymmetry = TRUE, keep = TRUE)
```

Arguments

`n`	sample size, a positive integer
`nu`	degrees of freedom, a positive number
`M`	mean matrix, without constraints
`U`	columns covariance matrix, a positive semidefinite matrix of order equal to `nrow(M)`
`V`	rows covariance matrix, a positive semidefinite matrix of order equal to `ncol(M)`
`checkSymmetry`	logical, whether to check the symmetry of `U` and `V`
`keep`	logical, whether to return an array with class **keep**

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Note

When `p=1` and `V=nu` or when `m=1` and `U=nu`, the distribution is the multivariate t-distribution.

Examples

```
nu <- 4
m <- 2
p <- 3
M <- matrix(1, m, p)
U <- toeplitz(m:1)
V <- toeplitz(p:1)
Tsims <- rmatrixt(10000, nu, M, U, V)
dim(Tsims) # 2 3 10000
apply(Tsims, 1:2, mean) # approximates M
vecTsims <- t(apply(Tsims, 3, function(X) c(t(X))))
round(cov(vecTsims), 1) # approximates 1/(nu-2) * kronecker(U,V)
## the `keep` class is nice when m=1 or p=1:
Tsims <- rmatrixt(2, nu, M=1:3, U=diag(3), V=1)
Tsims[,,1] # dimensions 3 1
```

```
# without `keep`, dimensions are lost:
rmatrixt(2, nu, M=1:3, U=diag(3), V=1, keep=FALSE)[,,1]
```

---

rwishart *Wishart sampler*

Description

Samples a Wishart distribution.

Usage

```
rwishart(n, nu, Sigma, Theta = NULL, epsilon = 0,
  checkSymmetry = TRUE)
```

Arguments

n	sample size, a positive integer
nu	degrees of freedom, a positive number; if nu < p-1 where p is the dimension (the order of Sigma), must be an integer; in the noncentral case (i.e. when Theta is not the null matrix), nu must satisfy nu >= p-1
Sigma	scale matrix, a positive semidefinite real matrix
Theta	noncentrality parameter, a positive semidefinite real matrix of same order as Sigma; setting it to NULL (default) is equivalent to setting it to the zero matrix
epsilon	a number involved in the algorithm only if it positive; its role is to guarantee the invertibility of the sampled matrices; see Details
checkSymmetry	logical, whether to check the symmetry of Sigma and Theta

Details

The argument epsilon is a threshold whose role is to guarantee that the algorithm samples invertible matrices when nu > p-1 and Sigma is positive definite. The sampled matrices are theoretically invertible under these conditions, but due to numerical issues, they are not always invertible when nu is close to p-1, i.e. when nu-p+1 is small. In this case, the chi-squared distributions involved in the algorithm can generate zero values or values close to zero, yielding the non-invertibility of the sampled matrices. These values are replaced with epsilon if they are smaller than epsilon.

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Note

A sampled Wishart matrix is always positive semidefinite. It is positive definite if nu > p-1 and Sigma is positive definite, in theory (see Details).

In the noncentral case, i.e. when Theta is not null, the Ahdida & Alfonsi algorithm is used if nu is not an integer and p-1 < nu < 2p-1, or if nu = p-1. The simulations are slower in this case.

References

A. Ahdida & A. Alfonsi. Exact and high-order discretization schemes for Wishart processes and their affine extensions. *The Annals of Applied Probability* **23**, 2013, 1025-1073.

Examples

```
nu <- 4
p <- 3
Sigma <- toeplitz(p:1)
Theta <- diag(p)
Wsims <- rwishart(10000, nu, Sigma, Theta)
dim(Wsims) # 3 3 10000
apply(Wsims, 1:2, mean) # approximately nu*Sigma+Theta
# the epsilon argument:
Wsims_det <- apply(rwishart(10000, nu=p-1+0.001, Sigma), 3, det)
length(which(Wsims_det < .Machine$double.eps))
Wsims_det <- apply(rwishart(10000, nu=p-1+0.001, Sigma, epsilon=1e-8), 3, det)
length(which(Wsims_det < .Machine$double.eps))
```

`rwishart_chol` *Sampling Cholesky factor of a Wishart matrix*

Description

Samples the lower triangular Cholesky factor of a Wishart random matrix.

Usage

```
rwishart_chol(n, nu, Sigma, epsilon = 0)
```

Arguments

`n`	sample size, a positive integer
`nu`	degrees of freedom, a number strictly greater than `p-1`, where `p` is the dimension (the order of `Sigma`)
`Sigma`	scale matrix, a positive definite real matrix
`epsilon`	a number involved in the algorithm only if it positive; its role is to guarantee the invertibility of the sampled matrices; see Details

Details

The argument `epsilon` is a threshold whose role is to guarantee that the algorithm samples invertible matrices. The matrices sampled by the algorithm are theoretically invertible. However, because of numerical precision, they are not always invertible when `nu` is close to `p-1`, i.e. when `nu-p+1` is small. In this case, the simulations of chi-squared distributions involved in the algorithm can generate zero values or values close to zero, yielding the non-invertibility of the sampled matrices. These values are replaced with `epsilon` if they are smaller than `epsilon`.

Value

A numeric three-dimensional array; simulations are stacked along the third dimension (see example).

Examples

```
nu <- 4
p <- 3
Sigma <- diag(p)
Wsims <- rwishart_chol(10000, nu, Sigma)
dim(Wsims) # 3 3 10000
Wsims[,,1]
```

Index