Package ‘HypergeoMat’

January 30, 2023

Type Package

Title Hypergeometric Function of a Matrix Argument

Version 4.0.2

Author Stéphane Laurent

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Evaluates the hypergeometric functions of a matrix argument, which appear in random matrix theory. This is an implementation of Koev & Edelman's algorithm (2006) <doi:10.1090/S0025-5718-06-01824-2>.

License GPL-3

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

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

Imports EigenR, gsl, JuliaConnectoR, Rcpp (>= 1.0.2)

Suggests Bessel, jack, knitr, rmarkdown, testthat

LinkingTo Rcpp, RcppEigen

VignetteBuilder knitr

Encoding UTF-8

RoxygenNote 7.2.3

SystemRequirements C++17

NeedsCompilation yes

Repository CRAN

Date/Publication 2023-01-30 13:10:02 UTC

R topics documented:

BesselA *Type one Bessel function of Herz*

Description

Evaluates the type one Bessel function of Herz.

Usage

```
BesselA(m, x, nu)
```

Arguments

m	truncation weight of the summation, a positive integer
x	either a real or complex square matrix, or a numeric or complex vector, the eigenvalues of the matrix
nu	the order parameter, real or complex number with `Re(nu)>-1`

Value

A real or complex number.

Note

This function is usually defined for a symmetric real matrix or a Hermitian complex matrix.

References

A. K. Gupta and D. K. Nagar. *Matrix variate distributions*. Chapman and Hall, 1999.

Examples

```
# for a scalar x, the relation with the Bessel J-function:
t <- 2
nu <- 3
besselJ(t, nu)
BesselA(m=15, t^2/4, nu) * (t/2)^nu
# it also holds for a complex variable:
if(require("Bessel")) {
  t <- 1 + 2i
  Bessel::BesselJ(t, nu)
  BesselA(m=15, t^2/4, nu) * (t/2)^nu
}
```

`hypergeomPFQ` *Hypergeometric function of a matrix argument*

Description

Evaluates a truncated hypergeometric function of a matrix argument.

Usage

```
hypergeomPFQ(m, a, b, x, alpha = 2)
```

Arguments

`m`	truncation weight of the summation, a positive integer
`a`	the "upper" parameters, a numeric or complex vector, possibly empty (or `NULL`)
`b`	the "lower" parameters, a numeric or complex vector, possibly empty (or `NULL`)
`x`	either a real or complex square matrix, or a numeric or complex vector, the eigenvalues of the matrix
`alpha`	the alpha parameter, a positive number

Details

This is an implementation of Koev & Edelman's algorithm (see the reference). This algorithm is split into two parts: the case of a scalar matrix (multiple of an identity matrix) and the general case. The case of a scalar matrix is much faster (try e.g. `x = c(1,1,1)` vs `x = c(1,1,0.999)`).

Value

A real or a complex number.

Note

The hypergeometric function of a matrix argument is usually defined for a symmetric real matrix or a Hermitian complex matrix.

References

Plamen Koev and Alan Edelman. *The Efficient Evaluation of the Hypergeometric Function of a Matrix Argument*. Mathematics of Computation, 75, 833-846, 2006.

Examples

```
# a scalar x example, the Gauss hypergeometric function
hypergeomPFQ(m = 10, a = c(1,2), b = c(3), x = 0.2)
gsl::hyperg_2F1(1, 2, 3, 0.2)
# 0F0 is the exponential of the trace
X <- toeplitz(c(3,2,1))/10
hypergeomPFQ(m = 10, a = NULL, b = NULL, x = X)
exp(sum(diag(X)))
# 1F0 is det(I-X)^(-a)
X <- toeplitz(c(3,2,1))/100
hypergeomPFQ(m = 10, a = 3, b = NULL, x = X)
det(diag(3)-X)^(-3)
# Herz's relation for 1F1
hypergeomPFQ(m = 10, a = 2, b = 3, x = X)
exp(sum(diag(X))) * hypergeomPFQ(m = 10, a = 3-2, b = 3, x = -X)
# Herz's relation for 2F1
hypergeomPFQ(10, a = c(1,2), b = 3, x = X)
det(diag(3)-X)^(-2) *
  hypergeomPFQ(10, a = c(3-1,2), b = 3, -X %*% solve(diag(3)-X))
```

---

hypergeomPFQ_julia *Evaluation with Julia*

Description

Evaluate the hypergeometric function of a matrix argument with Julia. This is highly faster.

Usage

```
hypergeomPFQ_julia()
```

Value

A function with the same arguments as hypergeomPFQ.

Note

See JuliaConnectoR-package for information about setting up Julia. If you want to directly use Julia, you can use my package.

Examples

```
library(HypergeoMat)
if(JuliaConnectoR::juliaSetupOk()){
  jhpq <- hypergeomPFQ_julia()
  jhpq(30, c(1+1i, 2, 3), c(4, 5), c(0.1, 0.2, 0.3+0.3i))
  JuliaConnectoR::stopJulia()
}
```

IncBeta — *Incomplete Beta function of a matrix argument*

Description

Evaluates the incomplete Beta function of a matrix argument.

Usage

```
IncBeta(m, a, b, x)
```

Arguments

`m`	truncation weight of the summation, a positive integer
`a, b`	real or complex parameters with `Re(a)>(p-1)/2` and `Re(b)>(p-1)/2`, where `p` is the dimension (the order of the matrix)
`x`	either a real positive symmetric matrix or a complex positive Hermitian matrix "smaller" than the identity matrix (i.e. `I-x` is positive), or a numeric or complex vector, the eigenvalues of the matrix

Value

A real or a complex number.

Note

The eigenvalues of a real symmetric matrix or a complex Hermitian matrix are always real numbers, and moreover they are positive under the constraints on `x`. However we allow to input a numeric or complex vector `x` because the definition of the function makes sense for such a `x`.

References

A. K. Gupta and D. K. Nagar. *Matrix variate distributions*. Chapman and Hall, 1999.

Examples

```
# for a scalar x, this is the incomplete Beta function:
a <- 2; b <- 3
x <- 0.75
IncBeta(m = 15, a, b, x)
gsl::beta_inc(a, b, x)
pbeta(x, a, b)
```

IncGamma — *Incomplete Gamma function of a matrix argument*

Description

Evaluates the incomplete Gamma function of a matrix argument.

Usage

```
IncGamma(m, a, x)
```

Arguments

`m`	truncation weight of the summation, a positive integer
`a`	real or complex parameter with `Re(a)>(p-1)/2`, where `p` is the dimension (the order of the matrix)
`x`	either a real or complex square matrix, or a numeric or complex vector, the eigenvalues of the matrix

Value

A real or complex number.

Note

This function is usually defined for a symmetric real matrix or a Hermitian complex matrix.

References

A. K. Gupta and D. K. Nagar. *Matrix variate distributions*. Chapman and Hall, 1999.

Examples

```
# for a scalar x, this is the incomplete Gamma function:
a <- 2
x <- 1.5
IncGamma(m = 15, a, x)
gsl::gamma_inc_P(a, x)
pgamma(x, shape = a, rate = 1)
```

mvbeta — *Multivariate Beta function (of complex variable)*

Description

The multivariate Beta function (`mvbeta`) and its logarithm (`lmvbeta`).

Usage

```
lmvbeta(a, b, p)

mvbeta(a, b, p)
```

Arguments

`a, b`	real or complex numbers with `Re(a)>0` and `Re(b)>0`
`p`	a positive integer, the dimension

Value

A real or a complex number.

Examples

```
a <- 5; b <- 4; p <- 3
mvbeta(a, b, p)
mvgamma(a, p) * mvgamma(b, p) / mvgamma(a+b, p)
```

mvgamma — *Multivariate Gamma function (of complex variable)*

Description

The multivariate Gamma function (`mvgamma`) and its logarithm (`lmvgamma`).

Usage

```
lmvgamma(x, p)

mvgamma(x, p)
```

Arguments

`x`	a real or a complex number; `Re(x)>0` for `lmvgamma` and `x` must not be a negative integer for `mvgamma`
`p`	a positive integer, the dimension

Value

A real or a complex number.

Examples

```
x <- 5
mvgamma(x, p = 2)
sqrt(pi)*gamma(x)*gamma(x-1/2)
```

Index