Package 'powerplus'

May 18, 2017

Type Package

Title Exponentiation Operations

Version 3.1

Date 2017-05-18

Author Albert Dorador

Maintainer Albert Dorador <albert.dorador@estudiant.upc.edu>

Description Computation of matrix and scalar exponentiation.

Depends R (>= 3.1.0)

Imports complexplus (>= 2.0), phonTools (>= 0.2-2.1), Matrix (>= 1.2-6), expm (>= 0.999-2), MASS (>= 7.3-45)

License GPL-2

ByteCompile true

RoxygenNote 6.0.1

NeedsCompilation no

Repository CRAN

Date/Publication 2017-05-18 16:02:19 UTC

R topics documented:

explus *Scalar Exponentiation*

Description

Raises any base (real or complex) to any power (even complex).

Usage

```
explus(a, numer, denom = 1, n.cycles = 10, tol = 1e-12)
```

Arguments

`a`	any base (real or complex). It can be a vector.
`numer`	numerator of exponent. Can be a decimal or complex number.
`denom`	denominator of exponent (1 by default). Can be a decimal or complex number.
`n.cycles`	The maximum number of steps to be used in the continued fraction approximation process done internally. Default is 10. Increasing this default may increase precision, but it may also produce unexpected results (even a runtime error).
`tol`	a tolerance, 10^{-12} by default. Prevents possible numerical problems. Can be set to 0 if desired.

Details

Method based on numerical treatment of complex exponents using Euler Formula and angles measured in radians, which is the default method built-in in R.

Value

The solution to the exponentiation operation supplied. Returns a real-valued root whenever possible. Otherwise, the principal complex root.

Author(s)

Albert Dorador

References

For more on Euler Formula, visit http://mathworld.wolfram.com/EulerFormula.html

For more on complex exponents, visit http://mathworld.wolfram.com/ComplexExponentiation.html

Examples

```
explus(-3, 4, 2)
explus(-3, 2, 4)
explus(-3, 2, 3)
explus(-3, 5, 3)
explus(-3, 5, 2)
explus(-3, -2, 4)
explus(0-0.5773503i, 2)
explus(-0.4, pi)
explus(-0.37, 0.2)
explus(-0.37, 1, 5)
explus(5, 7i)
explus(2+3i, 1+2i)
explus(2+3i, 1+2i, -4+1i)
explus(2+3i, 1+2i, 8)
```

Matpow — *Matrix Power*

Description

Raises a valid Matrix to any power (even complex). Valid matrices are square matrices that are diagonalizable or whose real eigenvalues are positive.

Usage

```
Matpow(M, numer, denom = 1, expmethod = "Higham08.b",
  logmethod = "Higham08", tol = 1e-12)
```

Arguments

`M`	a square matrix
`numer`	numerator of exponent. Can be a decimal or complex number.
`denom`	denominator of exponent (1 by default). Can be a decimal or complex number.
`expmethod`	method chosen to compute the matrix exponential if matrix is known to be non-diagonalizable. The default method is the same as in function `expm` from package **expm**.
`logmethod`	method chosen to compute the matrix logarithm if matrix is known to be non-diagonalizable. The default method is the same as in function `logm` from package **expm**.
`tol`	a tolerance, 10^{-12} by default. Prevents possible numerical problems. Can be set to 0 if desired.

Details

If the matrix is diagonalizable, the method used is based on spectral decomposition; if the matrix is not diagonalizable, the method used is based on matrix exponentials and logarithms, calling functions `matexp` and `matlog`, both from package **complexplus**. The particular method used to compute the matrix exponential and logarithm may be chosen from the options available in functions `expm` and `logm` respectively, both from package **expm**. Note that `Matpow`, by extension, allows one to compute roots and the matrix inverse (if invertible).

Value

The solution to the exponentiation operation supplied. For diagonalizable matrices, `Matpow` returns a real-valued root whenever possible (otherwise, the principal complex root).

Author(s)

Albert Dorador

References

For more on spectral decomposition (also known as eigendecomposition), visit http://mathworld.wolfram.com/EigenDecomposition.html

See Also

`matexp` `matlog` `expm` `logm`

Examples

```
A <- matrix(1:4, ncol = 2)

Matpow(A, 3)
Matpow(A, 0.5)
Matpow(A, 0.2)
Matpow(A, 1, 5)
Matpow(A, 2, 4, expmethod = "Pade", logmethod = "Eigen") #inocuous, as A is diagonalizable
Matpow(A, -1)
Matpow(A, 2+5i)
Matpow(A, 3i)
Matpow(A, 1+2i)
Matpow(A, 3i, 2+7i)

B <- matrix(sample(1:100, 81), ncol = 9)
Matpow(B, 2)
Matpow(B, 0.5)
Matpow(B, 7+2i)
Matpow(B, 4i, 1+3i)

C <- matrix(c(1, 0, 1, 1), ncol = 2) # A non-diagonalizable matrix
Matpow(C, 3)
Matpow(C, 0.5)
Matpow(C, 4, 8, expmethod = "Taylor", logmethod = "Eigen")
```

```
Matpow(C, 0.5*pi)
Matpow(C, 0.24)
Matpow(C, -2)
Matpow(C, 3+5i)
Matpow(C, 2i, 1+9i)
```

Index