matexpsym( ) — Exponentiation and logarithms of symmetric matrices

Description   Syntax   Remarks and examples   Conformability   Diagnostics   Also see

Description

`matexpsym(`*A*`)` returns the matrix exponential of the symmetric (Hermitian) matrix *A*.

`matlogsym(`*A*`)` returns the matrix natural logarithm of the symmetric (Hermitian) matrix *A*.

`_matexpsym(`*A*`)` and `_matlogsym(`*A*`)` do the same thing as `matexpsym()` and `matlogsym()`, but instead of returning the result, they store the result in *A*.

Syntax

numeric matrix	`matexpsym(`*numeric matrix A*`)`
numeric matrix	`matlogsym(`*numeric matrix A*`)`
void	`_matexpsym(`*numeric matrix A*`)`
void	`_matlogsym(`*numeric matrix A*`)`

Remarks and examples

Do not confuse `matexpsym(`*A*`)` with `exp(`*A*`)`, nor `matlogsym(`*A*`)` with `log(`*A*`)`.

`matexpsym(2*matlogsym(`*A*`))` produces the same result as *A*`*`*A*. `exp()` and `log()` return elementwise results.

Exponentiated results and logarithms are obtained by extracting the eigenvectors and eigenvalues of *A*, performing the operation on the eigenvalues, and then rebuilding the matrix. That is, first *X* and *L* are found such that

$$AX = X \times \text{diag}(L) \tag{1}$$

For symmetric (Hermitian) matrix *A*, *X* is orthogonal, meaning $X'X = XX' = I$. Thus

$$A = X \times \text{diag}(L) \times X' \tag{2}$$

`matexpsym(`*A*`)` is then defined

$$A = X \times \text{diag}(\exp(L)) \times X' \tag{3}$$

and `matlogsym(`*A*`)` is defined

$$A = X \times \text{diag}(\log(L)) \times X' \tag{4}$$

(1) is obtained via `symeigensystem()`; see [M-5] **eigensystem( )**.

Conformability

matexpsym(*A*),   matlogsym(*A*):

A:	$n \times n$
result:	$n \times n$

_matexpsym(*A*), _matlogsym(*A*):

input:

A:	$n \times n$

output:

A:	$n \times n$

Diagnostics

matexpsym(*A*), matlogsym(*A*), _matexpsym(*A*), and _matlogsym(*A*) return missing results if *A* contains missing values.

Also:

1. These functions do not check that *A* is symmetric or Hermitian. If *A* is a real matrix, only the lower triangle, including the diagonal, is used. If *A* is a complex matrix, only the lower triangle and the real parts of the diagonal elements are used.

2. These functions return a matrix of the same storage type as *A*.

   For symatlog(*A*), this means that if *A* is real and the result cannot be expressed as a real, a matrix of missing values is returned. If you want the generalized solution, code matlogsym(C(*A*)). This is the same rule as with scalars: log(-1) evaluates to missing, but log(C(-1)) is 3.14159265i.

3. These functions are guaranteed to return a matrix that is numerically symmetric, Hermitian, or symmetriconly if theory states that the matrix should be symmetric, Hermitian, or symmetriconly. See [M-5] **matpowersym()** for a discussion of this issue.

   For the functions here, real function exp(x) is defined for all real values of x (ignoring overflow), and thus the matrix returned by matexpsym() will be symmetric (Hermitian).

   The same is not true for matlogsym(). log(x) is not defined for $x = 0$, so if any of the eigenvalues of *A* are 0 or very small, a matrix of missing values will result. Also, log(x) is complex for $x < 0$, and thus if any of the eigenvalues are negative, the resulting matrix will be (1) missing if *A* is real stored as real, (2) symmetriconly if *A* contains reals stored as complex, and (3) general if *A* is complex.

Also see

[M-5] **eigensystem()** — Eigenvectors and eigenvalues

[M-5] **matpowersym()** — Powers of a symmetric matrix

[M-4] **Matrix** — Matrix functions

For suggested citations, see the FAQ on citing Stata documentation.