Title

cholinv() — Symmetric, positive-definite matrix inversion

Syntax

numeric matrix   `cholinv(`*numeric matrix A*`)`

numeric matrix   `cholinv(`*numeric matrix A*`,` *real scalar tol*`)`

void   `_cholinv(`*numeric matrix A*`)`

void   `_cholinv(`*numeric matrix A*`,` *real scalar tol*`)`

Description

`cholinv(`*A*`)` and `cholinv(`*A*`,` *tol*`)` return the inverse of real or complex, symmetric (Hermitian), positive-definite, square matrix *A*.

`_cholinv(`*A*`)` and `_cholinv(`*A*`,` *tol*`)` do the same thing except that, rather than returning the inverse matrix, they overwrite the original matrix *A* with the inverse.

In all cases, optional argument *tol* specifies the tolerance for determining singularity; see *Remarks and examples* below.

Remarks and examples

stata.com

These routines calculate the inverse of a symmetric, positive-definite square matrix *A*. See [M-5] **luinv()** for the inverse of a general square matrix.

A is required to be square and positive definite. See [M-5] **qrinv()** and [M-5] **pinv()** for generalized inverses of nonsquare or rank-deficient matrices. See [M-5] **invsym()** for generalized inverses of real, symmetric matrices.

`cholinv(`*A*`)` is logically equivalent to `cholsolve(`*A*`, I(rows(`*A*`)))`; see [M-5] **cholsolve()** for details and for use of the optional *tol* argument.

Conformability

`cholinv(`*A*`,` *tol*`)`:

A:	$n \times n$	
tol:	1×1	(optional)
result:	$n \times n$	

`_cholinv(`*A*`,` *tol*`)`:

input:

A:	$n \times n$	
tol:	1×1	(optional)

output:

A:	$n \times n$

Diagnostics

The inverse returned by these functions is real if *A* is real and is complex if *A* is complex. If you use these functions with a non–positive-definite matrix, or a matrix that is too close to singularity, returned will be a matrix of missing values. The determination of singularity is made relative to *tol*. See *Tolerance* under *Remarks and examples* in [M-5] **cholsolve( )** for details.

`cholinv(`*A*`)` and `_cholinv(`*A*`)` return a result containing all missing values if *A* is not positive definite or if *A* contains missing values.

`_cholinv(`*A*`)` aborts with error if *A* is a view.

See [M-5] **cholsolve( )** and [M-1] **tolerance** for information on the optional *tol* argument.

Both functions use the elements from the lower triangle of *A* without checking whether *A* is symmetric or, in the complex case, Hermitian.

Also see

[M-5] **invsym( )** — Symmetric real matrix inversion

[M-5] **luinv( )** — Square matrix inversion

[M-5] **qrinv( )** — Generalized inverse of matrix via QR decomposition

[M-5] **pinv( )** — Moore–Penrose pseudoinverse

[M-5] **cholsolve( )** — Solve AX=B for X using Cholesky decomposition

[M-5] **solve_tol( )** — Tolerance used by solvers and inverters

[M-4] **matrix** — Matrix functions

[M-4] **solvers** — Functions to solve AX=B and to obtain A inverse