svsolve() — Solve AX=B for X using singular value decomposition

Description

svsolve(A, B, ...), uses singular value decomposition to solve AX = B and return X. When A is singular, svsolve() computes the minimum-norm least-squares generalized solution. When rank is specified, it is placed the rank of A.

svsolve(A, B, ...) does the same thing, except that it destroys the contents of A and it overwrites B with the solution. Returned is the rank of A.

In both cases, tol specifies the tolerance for determining whether A is of full rank. tol is interpreted in the standard way—as a multiplier for the default if tol > 0 is specified and as an absolute quantity to use in place of the default if tol ≤ 0 is specified.

Syntax

numeric matrix    svsolve(A, B)
numeric matrix    svsolve(A, B, rank)
numeric matrix    svsolve(A, B, rank, tol)
real scalar       _svsolve(A, B)
real scalar       _svsolve(A, B, tol)

where

A:    numeric matrix
B:    numeric matrix
rank: irrelevant; real scalar returned
tol:  real scalar

Remarks and examples

svsolve(A, B, ...) is suitable for use with square or nonsquare, full-rank or rank-deficient matrix A. When A is of full rank, svsolve() returns the same solution as lusolve() (see [M-5] lusolve()), ignoring roundoff error. When A is singular, svsolve() returns the minimum-norm least-squares generalized solution. qrsolve() (see [M-5] qrsolve()), an alternative, returns a generalized least-squares solution that amounts to dropping rows of A.

Remarks are presented under the following headings:

Derivation
Relationship to inversion
Tolerance
Derivation

We wish to solve for $X$

$$AX = B \tag{1}$$

Perform singular value decomposition on $A$ so that we have $A = USV'$. Then (1) can be rewritten as

$$USV'X = B$$

Premultiplying by $U'$ and remembering that $U'U = I$, we have

$$SV'X = U'B$$

Matrix $S$ is diagonal and thus its inverse is easily calculated, and we have

$$V'X = S^{-1}U'B$$

When we premultiply by $V$, remembering that $VV' = I$, the solution is

$$X = VS^{-1}U'B \tag{2}$$

See \[ M-5 \] svd() for more information on the SVD.

Relationship to inversion

For a general discussion, see Relationship to inversion in \[ M-5 \] lusolve().

For an inverse based on the SVD, see \[ M-5 \] pinv(). pinv($A$) amounts to svsolve($A$, I(rows($A$))), although pinv() has separate code that uses less memory.

Tolerance

In (2) above, we are required to calculate the inverse of diagonal matrix $S$. The generalized solution is obtained by substituting zero for the $i$th diagonal element of $S^{-1}$, where the $i$th diagonal element of $S$ is less than or equal to $eta$ in absolute value. The default value of $eta$ is

$$eta = \text{epsilon}(1) \times \text{rows}(A) \times \text{max}(S)$$

If you specify $tol > 0$, the value you specify is used to multiply $eta$. You may instead specify $tol \leq 0$ and then the negative of the value you specify is used in place of $eta$; see \[ M-1 \] tolerance.
Conformability

svsolve(A, B, rank, tol):

input:
A: m × n
B: m × k
tol: 1 × 1 (optional)

output:
rank: 1 × 1 (optional)
result: n × k

svsolve(A, B, tol):

input:
A: m × n
B: m × k
tol: 1 × 1 (optional)

output:
A: 0 × 0
B: m × k
result: 1 × 1

Diagnostics

svsolve(A, B, . . .) and _svsolve(A, B, . . .) return missing results if A or B contain missing.

svsolve(A, B, . . .) aborts with error if A (but not B) is a view.

Also see

[M-5] solvelower() — Solve AX=B for X, A triangular
[M-5] cholsolve() — Solve AX=B for X using Cholesky decomposition
[M-5] lusolve() — Solve AX=B for X using LU decomposition
[M-5] qrsolve() — Solve AX=B for X using QR decomposition
[M-4] matrix — Matrix functions
[M-4] solvers — Functions to solve AX=B and to obtain A inverse