Linalg v1.0: Supplemental Linear Algebra functions not found on the HP48-GX

Scott Hyde
Department of Mathematical Sciences
Montana State University

January 30, 2002

Contents

1 Introduction

1.1 Overview

Linalg provides some added functions for numerical matrix. These commands include generalized inverses, projection operators, matrix factorizations, basis sets for vector spaces, and other commands. In addition, it also provides some symbolic matrix building commands.

1.2 Disclaimer

Linalg and its attached documentation are provided "as is", and are subject to change without prior notice. The author gives no warranty of any kind with regard to the software or documentation, either expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. The author shall not be held liable for any damages, including any general, special, incidental, or consequential damages arising out of the use or inability to use any or all of the included programs.

Permission to copy the Linalg library as a whole, unmodified package is granted provided that the copies are not made or distributed for resale (excepting nominal copying fees).

1.3 Installing and Deleting Linalg

Size	5950.5 bytes
Checksum	#2162h [#8546d]
Version	1.0
Library Number	420

You can check these numbers by putting the library on the stack and pressing LS-[VAR] [BYTES]. The above statistics correspond to the distributed version, and if they do not match your results, your copy of the library may have been modified.

Linalg should only work in a G/GX Calculator. No testing has been done other than Revision R (mine). However, as always, you should BACKUP YOUR MEMORY before running this application.

To install Linalg:

1. Download the file `linalg420.lib` into your calculator in binary mode.
2. Put the content of the created variable `linalg420.lib` on the stack.
3. Delete the variable `linalg420.lib`.
4. Store it in the port of your choice (for example with 0 STO.)
5. Power-cycle the calculator (OFF ON or ON-C).

To delete Linalg:

1. Detach the library with {:0:420} DETACH (1,2 for the cards).
2. Purge the library with {:0:420} PURGE (1,2 for the cards). If you get an error message 'Object in use' then do ON-C and start again from 3.

The source code is given in the files `linalg420src` or `linalg420src.bz`. The `linalg420src` file is an HP48 directory object, and the `linalg420src.bz` is the compressed version of `linalg420src` (compressed by Mika Heiskanen's BZ). Both of these are binary files and must be transferred to the calculator to be viewed. Compilation was done using Jazz v6.7.

1.4 Linalg homepage

The Linalg homepage is at `http://www.math.montana.edu/~hyde/`. The page always contains a link to the latest published version of Linalg.

2 Matrix Factorizations

2.1 CHOL

Cholesky Factorization of a Matrix Command: Returns the Cholesky Factorization of an $n \times n$ positive definite matrix.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_L$

Remarks: Factors a positive definite matrix into

$$\boldsymbol{A} = \boldsymbol{L}\boldsymbol{L}',$$

where $\boldsymbol{L}$ is the unique lower triangular matrix having nonnegative diagonal elements. Uses algorithm found in Werner Huysegom's Cholesky Factorization. found on `http://www.hpcalc.org`.

Related Commands: FR, LQ, LU, QR, SCHUR, SVD, SVL, TRN

2.2 FR

Full Rank Factorization of a Matrix Command: Returns a Full Rank Factorization of an $n \times p$ matrix.

Level 1	$\rightarrow$	Level 2	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_C$	$[[matrix]]_R$

Affected by: The variable TOL in the HOME directory

Remarks: Factors any $n \times p$ matrix $\boldsymbol{A}$ with rank r into two matrices, such that

$$\boldsymbol{A} = \boldsymbol{C}\boldsymbol{R},$$

where $\boldsymbol{C}$ is an $n \times r$ matrix, $\boldsymbol{R}$ is an $r \times p$ matrix, and $\text{rank}(\boldsymbol{C}) = \text{rank}(\boldsymbol{R}) = r$. Both $\boldsymbol{C}$ and $\boldsymbol{R}$ are not unique, because any $r \times r$ full rank matrix and its inverse can be inserted in between $\boldsymbol{C}$ and $\boldsymbol{R}$:

$$\boldsymbol{A} = \underbrace{\boldsymbol{C}\boldsymbol{Q}}_{\boldsymbol{C}^*}\underbrace{\boldsymbol{Q}^{-1}\boldsymbol{R}}_{\boldsymbol{R}^*} = \boldsymbol{C}^*\boldsymbol{R}^*.$$

See RANKQR or RANKSVD on the use of TOL, and on the computation of $\text{rank}(\boldsymbol{A})$.

Related Commands: CHOL, LQ, LU, QR, SCHUR, SVD, SVL, TRN

---

3 Matrix Generation

3.1 COMMUTE

Commutation Matrix Command: Generates the (a, b) Commutation Matrix $\boldsymbol{I}_{(a,b)}$.

Level 1	→	Level 1
{ a b }	→	$[[matrix]]_{I_{(a,b)}}$

Remarks: The Commutation matrix is a permuted identity matrix which has the property

$$\text{vec}\left(\boldsymbol{A}\right) = \boldsymbol{I}_{(a,b)}\,\text{vec}\left(\boldsymbol{A}'\right),$$

where $\boldsymbol{A}$ is an $a \times b$ matrix. In addition, the commutation matrix can interchange the order of a Kronecker product. In short, if $\boldsymbol{A}$ is $a \times b$ and B is $c \times d$, then

$$\boldsymbol{A} \otimes \boldsymbol{B} = \boldsymbol{I}_{(c,a)}\left(\boldsymbol{B} \otimes \boldsymbol{A}\right)\boldsymbol{I}_{(b,d)}.$$

Related Commands: NP, DVECC, KRON, VECC.

3.2 DP

Duplication Matrix Command: Generates the p^{th} Duplication Matrix.

Level 1	→	Level 1
p	→	$[[matrix]]_{D_p}$

Remarks: The Duplication matrix is defined by the equation:

$$\text{vec}\left(\boldsymbol{A}\right) = \boldsymbol{D}_p\,\text{vech}\left(\boldsymbol{A}\right),$$

for a $p \times p$ matrix $\boldsymbol{A}$. The operator vech stands for "vector half", which stacks the distinct elements of $\boldsymbol{A}$ on top of each other (in column order). Since it only includes the lower triangular part of $\boldsymbol{A}$, then $\boldsymbol{D}_p$ is a $p^2 \times \frac{p(p+1)}{2}$ matrix.

Related Commands: NP, VECC, DVECC.

3.3 DVECC

Reverse the Vec Command: Inverts the vec operator.

Level 1	Level 2	→	Level 1
$[[matrix]]_{p^2\times 1}$	{ a b }	→	$[[matrix]]_{a\times b}$

Remarks: Reverses the process of the vec operator by reshaping the matrix according to columns. Further, ab must be the same as p^2.

Related Commands: NP, VECC, DVECC.

3.4 GEN

Generate Random Matrix with Specified Rank Command: Generates a random matrix $\boldsymbol{A}$, such that the rank($\boldsymbol{A}$) = r.

Level 2	Level 1	→	Level 1
$\{n_{\text{rows}} \quad p_{\text{cols}}\}$	r_{rank}	→	$[[matrix]]_A$

Remarks: Uses the RANM command to generate the matrix.

Related Commands: HILBERT, RANM.

3.5 GENGINV

Generate another Generalized Inverse Command: Generates another generalized inverse for a matrix $\boldsymbol{A}$, given the matrix $\boldsymbol{A}$, and a generalized inverse $\boldsymbol{A}^-$.

Level 2	Level 1	→	Level 1
$[[matrix]]_A$	$[[matrix]]_{A^-}$	→	$[[matrix]]_{A^-}$

Remarks: A generalized inverse is defined as only satisfying the first of the four conditions for the Moore Penrose Generalized Inverse (e.g. When $\boldsymbol{A}\boldsymbol{A}^-\boldsymbol{A} = \boldsymbol{A}$, then $\boldsymbol{A}^-$ is a generalized inverse of $\boldsymbol{A}$).

Related Commands: PINVQR, PINVSVD.

3.6 HILBERT

Hilbert Matrix Generation Command: Generates the $n \times n$ Hilbert matrix.

Level 1	→	Level 1
n	→	$[[matrix]]_H$

Remarks: The Hilbert matrices are useful for testing the commands used in Linalg. They are invertible for every value of n, but are ill-conditioned. In fact, $|H_n| \to 0$ as $n \to \infty$. Compute the condition number of the matrices and read the section on Ill-Conditioned matrices in the HP User's Manual (Chapter 14) for more information.

Related Commands: GEN, HILINV, RANM.

3.7 HILINV

Inverse of the Hilbert Matrix Generation Command: Generates the inverse of the $n \times n$ Hilbert matrix.

Level 1	→	Level 1
n	→	$[[matrix]]_{H^{-1}}$

Remarks: This generates the inverse of the Hilbert matrices exactly. Check its answer to the answer produced by the HILBERT command followed by the INV command. These matrices are as ill-conditioned as the Hilbert matrices, yet are always invertible

Related Commands: GEN, HILBERT, RANM.

3.8 NP

NP Matrix Command: Generates the perpendicular projection operator of $\boldsymbol{D}_p$

Level 1	→	Level 1
p	→	$[[matrix]]_{\boldsymbol{N}_p}$

Remarks: The $p^2 \times p^2$ matrix $\boldsymbol{N}_p$ has the unique property of being the projection operator for the vector space of all symmetric matrices. In particular

$$\boldsymbol{N}_p \operatorname{vec} \boldsymbol{A} = \operatorname{vec} \boldsymbol{A},$$

where $\boldsymbol{A}$ is a $p \times p$ symmetric matrix.

Related Commands: DP, DVECC, VECC.

3.9 VECC

Vec Operation Command: Coverts a matrix to a vector by stacking the columns one on top of the other.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{ab\times 1}$

Remarks: Transforms a matrix into a vector by stacking the columns of an $a \times b$ matrix $\boldsymbol{A}$ on top of each other. In addition, the vec operator is related to the Kronecker product through the equation

$$\operatorname{vec}(\boldsymbol{ABC}) = \left(\boldsymbol{C}' \otimes \boldsymbol{A}\right) \operatorname{vec}(\boldsymbol{B}).$$

Related Commands: DP, COMMUTE, DVECC, KRON, NP.

3.10 VECCOL

Random Vector in the Column Space Command: Generates a random vector in the column space of $\boldsymbol{A}$.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{p\times 1}$

Remarks: Uses RANM to find a random vector in the column space of $\boldsymbol{A}$.

Related Commands: GEN, RANM.

---

4 Miscellaneous Linear Algebra Commands

4.1 CEQN

Characteristic Equation of a Matrix Command: Computes the characteristic equation of a square matrix, which is defined as $|\boldsymbol{A} - \lambda \boldsymbol{I}_n|$. It may be more appropriate to call it the characteristic polynomial of a matrix since it is an n^{th} degree polynomial in λ.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[vector]_{ceqn}$

Remarks: This is a direct copy of the CEQN v1.1 command by Sune Bredahl. Incorporated into the library because I used it so much.

Related Commands: EGV, EGVL, EIGV, EIGSPACE

4.2 EIGV

Eigenvalues of a Matrix Command: Finds the eigenvalues of an $n \times n$ matrix $\boldsymbol{A}$.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[vector]_{eigs}$

Affected by: The commands CEQN and PROOT.

Remarks: Finds the eigenvalues of a square matrix $\boldsymbol{A}$ by using the program CEQN and PROOT. Produces results for small matrices quicker than EGVL, however, sometimes produces inaccurate results for larger matrices due to difficulties with solving high order polynomials by PROOT.

Related Commands: COLSPACE, EIGSPACE, EGV, EGVL, NULLQR, NULLSVD

4.3 GRSCMT

Gram-Schmidt Command: Applies the Gram-Schmidt process to convert any basis set of column vectors into an orthogonal basis set of vectors. Because a basis set of column vectors is needed, it only works when $\boldsymbol{A}$ has full column rank. When $\boldsymbol{A}$ has less than full column rank, use COLSPACE instead.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_U$

Affected by: Truncation error when adding vectors.

Remarks: Kept for sentimental purposes. COLSPACE works better in producing an orthogonal basis set of vectors.

Related Commands: COLSPACE, NORMALIZE, NULLQR, NULLSVD.

4.4 HELP420

Help Command: Gives a short description of each command in the Linear Algebra library.

4.5 KRON

Kronecker Product Command: Returns the Kroenecker product of two matrices.

Level 1	Level 2	→	Level 1
$[[matrix]]_A$	$[matrix]]_B$	→	$[[matrix]]_{A\otimes B}$

Remarks: Computes the Kronecker product of two matrices. Suppose that $\boldsymbol{A}$ is an $a \times b$ matrix and $\boldsymbol{B}$ is a $c \times d$ matrix. Then the Kronecker product of $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as the $ac \times bd$ matrix

$$\boldsymbol{A} \otimes \boldsymbol{B} = \{a_{ij}\boldsymbol{B}\}_{ij},$$

e.g. the ij^{th} submatrix is $a_{ij}\boldsymbol{B}$.

Related Commands: COMMUTE, DP, DVECC, NP, VECC

4.6 MATPOW

Integer Power of a Matrix Command: Returns the n^{th} integer power of any square matrix $\boldsymbol{A}$.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{A^n}$

Remarks: I did not write this. Someone posted it to `comp.sys.hp48` a while back. If you know who, let me know.

4.7 MEXP

Matrix Exponential Command: Computes the matrix exponential of an $n \times n$ matrix.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{e^A}$

Affected by: Does not use full 15 digit internal precision. Wish it did, but I don't know how to write 15 digit commands to do matrix multiplication or addition.

Remarks: Computes the matrix exponential of an $n \times n$ matrix defined as

$$e^{\boldsymbol{A}} = \boldsymbol{I} + \boldsymbol{A} + \frac{\boldsymbol{A}^2}{2!} + \frac{\boldsymbol{A}^3}{3!} + \cdots.$$

The preceding equation is good for heuristic purposes, but not for computational purposes. I used an algorithm from MATLAB v5.3.

4.8 NORMALIZE

Normalized Column Vectors Command: Normalizes the column vectors of the matrix $\boldsymbol{A}$.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_U$

Remarks: Normalizes the column vectors of the matrix $\boldsymbol{A}$, such that $\boldsymbol{U}'\boldsymbol{U}$ has ones on the diagonal. If GRSCHMT can be used, then GRSCHMT followed by NORMALIZE produces an orthonormal basis set of vectors.

Related Commands: GRSCMT.

4.9 PINVQR

Moore Penrose Generalized Inverse Command: Computes the Moore Penrose Generalized Inverse of any $n \times p$ matrix.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{A^+}$

Affected by: The variable TOL in the HOME directory

Remarks: Computes the Moore Penrose Generalized Inverse of an $n \times p$ matrix $\boldsymbol{A}$, called $\boldsymbol{A}^+$, using the QR Factorization. Computation is sometimes much faster for PINVQR than with PINVSVD. However, PINVSVD is sometimes more accurate. $\boldsymbol{A}^+$, sometimes called the "pseudo" inverse, satisfies these four properties:

- $\boldsymbol{A}\boldsymbol{A}^+\boldsymbol{A} = \boldsymbol{A}$,
- $\boldsymbol{A}^+\boldsymbol{A}\boldsymbol{A}^+ = \boldsymbol{A}^+$,
- $\boldsymbol{A}\boldsymbol{A}^+$ is symmetric,
- $\boldsymbol{A}^+\boldsymbol{A}$ is symmetric.

See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: GENGINV, PINVSVD.

4.10 PINVSVD

Moore Penrose Generalized Inverse Command: Computes the Moore Penrose Generalized Inverse, $\boldsymbol{A}^+$, of any $n \times p$ matrix.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_{A^+}$

Affected by: The variable TOL in the HOME directory

Remarks: Computes the Moore Penrose Generalized Inverse of a rectangular matrix $\boldsymbol{A}$ using the SVD Factorization. Computation can be much longer than for PINVQR, but sometimes produces

more accurate results. See PINVQR for more information. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: GENGINV, PINVSVD.

4.11 RANKQR

Rank of a Matrix Command: Computes the rank of an $n \times p$ matrix using the QR decomposition.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	r_{rank}

Affected by: The variable TOL in the HOME directory

Remarks: Suppose, without loss of generality, that $n \geq p$. By the QR factorization, $\boldsymbol{A}$ can be decomposed into

$$\boldsymbol{AP} = \boldsymbol{QR},$$

where $\boldsymbol{P}$ is a $p \times p$ permutation matrix ($\boldsymbol{P}^{-1} = \boldsymbol{P}'$), $\boldsymbol{Q}$ is a $n \times n$ orthogonal matrix, and $\boldsymbol{R}$ is an $n \times p$ upper triangular matrix. $\boldsymbol{R}$ can then be partitioned as

$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{R}_1 \\ \boldsymbol{0} \end{pmatrix}, \tag{1}$$

where $\boldsymbol{R}_1$ is a $r \times p$ matrix. Finally,

$$\text{rank}(\boldsymbol{A}) = \text{rank}(\boldsymbol{AP}) = \text{rank}(\boldsymbol{QR}) = \text{rank}(\boldsymbol{R}) = \text{rank}(\boldsymbol{R}_1) = r.$$

Since the HP48GX works mainly with numerical matrices, then if any of the diagonal elements of $\boldsymbol{R}$ are less than TOL $\cdot \|\boldsymbol{A}\|$, they are considered zero. $\|\boldsymbol{A}\|$ is defined as the Frobenius norm:

$$\|\boldsymbol{A}\| = \sqrt{\sum \text{Diag}(\boldsymbol{A}'\boldsymbol{A})}.$$

If no value is specified in TOL, then the value of 10^{-13} is used.

Related Commands: LQ, LSQ, QR, RANKSVD.

4.12 RANKSVD

Rank of a Matrix Command: Computes the rank of an $n \times p$ matrix using the SVD decomposition.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	r_{rank}

Affected by: The variable TOL in the HOME directory.

Remarks: Computes the rank by counting all the singular values of the matrix less than or equal to TOL times the largest computed singular value. The HP48 only uses the value of 1.E-14 for TOL. See RANKQR for more information.

Related Commands: LQ, LSQ, QR, RANKQR.

4.13 TOL

Tolerance variable: Sets a limit on how small "zero" really is. Numbers that are less than or equal to TOL are considered 0. If TOL does not exist in the HOME directory, 1.E-13 is used.

Remarks: Used by: COLSPACE, EIGSPACE, FR, NULLQR, NULLSVD, PINVQR, PINVSVD, POQR, POSVD, PPOQR, PPOSVD, RANKQR, and RANKSVD.

---

5 Projection Operator Commands

5.1 POQR

Projection Operator Command: Generates the $n \times n$ projection matrix which projects onto $\mathcal{R}(\boldsymbol{A})$ along $\mathcal{N}(\boldsymbol{A}'\boldsymbol{B}^{-1})$, where $\boldsymbol{A}$ is an $n \times p$ matrix and $\boldsymbol{B}$ is an $n \times n$ positive definite matrix. No check is made on whether $\boldsymbol{B}$ is positive definite. The QR decomposition is used in the computations.

Level 2	Level 1	→	Level 1
$[[matrix]]_A$	$[[matrix]]_B$	→	$[[matrix]]_P$

Affected by: The variable TOL in the HOME Directory.

Remarks: Specifically,

$$\boldsymbol{P} = \boldsymbol{A}(\boldsymbol{A}'\boldsymbol{B}^{-1}\boldsymbol{A})^{-}\boldsymbol{A}'\boldsymbol{B}^{-1},$$

where $(\boldsymbol{A}'\boldsymbol{B}^{-1}\boldsymbol{A})^{-}$ is any generalized inverse of $\boldsymbol{A}'\boldsymbol{B}^{-1}\boldsymbol{A}$ satisfying only the first property of the Moore Penrose Generalized Inverse (See PINVQR). Even though the generalized inverse indicated above is not unique, $\boldsymbol{P}$ is unique.

$\boldsymbol{P}$ is normally rank deficient. This leads to problems when calculating its rank. However, for projection operators, the rank can be found by simply taking its trace. The HP48's command RANK does not calculate the rank for projection operators very efficiently, so either take the trace of the projection matrix, or use RANKQR or RANKSVD. If necessary, adjust the TOL variable to specify a different tolerance limit. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: POSVD, PPOQR, PPOSVD.

5.2 POSVD

Projection Operator Command: Generates the $n \times n$ projection matrix which projects onto $\mathcal{R}(\boldsymbol{A})$ along $\mathcal{N}(\boldsymbol{A}'\boldsymbol{B}^{-1})$, where $\boldsymbol{A}$ is an $n \times p$ matrix and $\boldsymbol{B}$ is an $n \times n$ positive definite matrix. No check is made on whether $\boldsymbol{B}$ is positive definite. The SVD decomposition is used to compute this.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_P$

Affected by: The variable TOL in the HOME directory.

Remarks: See POQR for more information. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: POQR, PPOQR, PPOSVD.

5.3 PPOQR

Perpendicular Projection Operator Command: Computes the perpendicular projection operator which projects onto $\mathcal{R}(\boldsymbol{A})$ along $\mathcal{N}(\boldsymbol{A}')$, where $\boldsymbol{A}$ is an $n \times p$ matrix. The QR decomposition is used to compute this.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_P$

Affected by: The variable TOL in the HOME directory.

Remarks: Specifically,

$$\boldsymbol{P} = \boldsymbol{A}(\boldsymbol{A}'\boldsymbol{A})^{-}\boldsymbol{A}',$$

where $(\boldsymbol{A}'\boldsymbol{A})^{-}$ is any generalized inverse of $\boldsymbol{A}'\boldsymbol{A}$ satisfying only the first property of the Moore Penrose Generalized Inverse (See PINVQR). Even though the generalized inverse indicated above is not unique, $\boldsymbol{P}$ is unique. $\boldsymbol{P}$ is also symmetric.

$\boldsymbol{P}$ is normally rank deficient. This leads to problems when calculating its rank. However, for projection operators, the rank can be found by simply taking its trace. The HP48's command RANK does not calculate the rank for projection operators very efficiently, so either take the trace of the projection matrix, or use RANKQR or RANKSVD. If necessary, adjust the TOL variable to specify a different tolerance limit. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: POQR, POSVD, PPOSVD.

5.4 PPOSVD

Perpendicular Projection Operator Command: Computes the perpendicular projection operator which projects onto $\mathcal{R}(\boldsymbol{A})$ along $\mathcal{N}(\boldsymbol{A}')$, where $\boldsymbol{A}$ is an $n \times p$ matrix. The SVD decomposition is used to compute this.

Level 1	→	Level 1
$[[matrix]]_A$	→	$[[matrix]]_P$

Affected by: The variable TOL in the HOME directory.

Remarks: See PPOQR for more information. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: POQR, POSVD, PPOQR.

---

6 Symbolic Matrix Tools

6.1 SARRY→

Symbolic Array to Stack Command: Takes a symbolic array and returns its elements as separate objects and a list of the dimensions of the array.

Level 1	→	Level np+1 ... Level 2	Level 1
$\{\{matrix\}\}$	→	$z_1\ z_2\ \cdots\ z_{np}$	$\{n_{\text{rows}}\ \ p_{\text{cols}}\}$

Related Commands: ARRY→, →SARRY.

6.2 →SARRY

Stack to Symbolic Array Command: Returns an $n \times p$ symbolic matrix given the objects on the stack and a list containing

Level $np+1$ ... Level 2	Level 1	→	Level 1
$z_1\ z_2\ \cdots\ z_{np}$	$\{n_{\text{rows}}\ \ p_{\text{cols}}\}$	→	$\{\{matrix\}\}$

Remarks: The elements are entered on the stack in row order.

Related Commands: →ARRY, SARRY→.

6.3 SCOL→

Symbolic Columns to Symbolic Matrix Command: Transforms a series of column vectors and a column count into a symbolic matrix containing those columns.

Level $n+1 \cdots$ Level 2	Level 1	→	Level 1
$\{vector\}_{\text{col1}}\{vector\}_{\text{col2}}\cdots\{vector\}_{\text{coln}}$	n_{colcount}	→	$\{\{matrix\}\}$

Remarks: Works like the COL→ command.

Related Commands: COL→, →SCOL.

6.4 →SCOL

Symbolic Matrix to Columns Command: Transforms a symbolic matrix into a series of symbolic vectors, consisting of the columns of the matrix, and a vector count.

Level 1	→	Level $n+1 \cdots$ Level 2	Level 1
$\{\{matrix\}\}$	→	$\{vector\}_{\text{col1}}\{vector\}_{\text{col2}}\cdots\{vector\}_{\text{coln}}$	n_{colcount}

Related Commands: →COL, SCOL→.

6.5 SROW→

Symbolic Rows to Symbolic Matrix Command: Transforms a series of row vectors and a column count into a symbolic matrix containing those rows.

Level $n+1 \cdots$ Level 2	Level 1	→	Level 1
$\{vector\}_{\text{row1}}\{vector\}_{\text{row2}} \cdots \{vector\}_{\text{rown}}$	n_{rowcount}	→	$\{\{matrix\}\}$

Related Commands: ROW→, →SROW.

6.6 →SROW

Symbolic Matrix to Rows Command: Transforms a symbolic matrix into a series of symbolic vectors, consisting of the columns of the matrix, and a vector count.

Level 1	→	Level $n+1 \cdots$ Level 2	Level 1
$\{\{matrix\}\}$	→	$\{vector\}_{\text{row1}}\{vector\}_{\text{row2}} \cdots \{vector\}_{\text{rown}}$	n_{rowcount}

Related Commands: →ROW, SROW→.

---

7 Vector Space Commands

7.1 COLSPACE

Column Space Command: Finds a basis set of r orthonormal vectors spanning the column space of the $n \times p$ matrix $\boldsymbol{A}$, where $r = \text{rank}(\boldsymbol{A})$.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_R$

Affected by: The variable TOL in the HOME Directory.

Remarks: Finds a basis set of orthonormal vectors spanning the column space of the matrix $\boldsymbol{A}$. The column space of $\boldsymbol{A}$ is defined as

$$\mathcal{R}(\boldsymbol{A}) = \{\boldsymbol{z}; \boldsymbol{z} = \boldsymbol{A}\boldsymbol{b} \text{ for some } \boldsymbol{b} \in \mathbb{R}^p\}.$$

The resulting matrix $\boldsymbol{R}$ contains r linearly independent vectors also spanning the column space of $\boldsymbol{A}$, so that

$$\mathcal{R}(\boldsymbol{A}) = \mathcal{R}(\boldsymbol{R}).$$

Further, $\boldsymbol{R}'\boldsymbol{R} = \boldsymbol{I}_r$. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: EIGSPACE, NULLQR, NULLSVD

7.2 EIGSPACE

Eigenspace of a Matrix Command: Finds a basis set of orthonormal vectors for the eigenspace of an $n \times n$ matrix $\boldsymbol{A}$ corresponding to the eigenvalue λ.

Level 2	Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	λ	$\rightarrow$	$[[matrix]]_{Eigspace}$

Affected by: The variable TOL in the HOME directory.

Remarks: The eigenspace is defined as the nullspace of the matrix $\boldsymbol{A} - \lambda \boldsymbol{I}_n$. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: COLSPACE, EIGV, EGV, EGVL, NULLQR, NULLSVD

7.3 NULLQR

Null space of a Matrix Command: Computes an orthonormal basis for the null space of any $n \times p$ matrix.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_{Null}$

Affected by: The variable TOL in the HOME directory

Remarks: Computes an orthonormal basis for the null space of an $n \times p$ matrix $\boldsymbol{A}$, using the QR decomposition. It is much faster than NULLSVD, but NULLSVD sometimes produces more accurate results. The null space of $\boldsymbol{A}$ is defined as all $p \times 1$ vectors, $\boldsymbol{z}$, such that $\boldsymbol{Az} = \boldsymbol{0}$; In other words,

$$\mathcal{N}(\boldsymbol{A}) = \{\boldsymbol{z}; \boldsymbol{z} \in \mathbb{R}^p,\ \boldsymbol{Az} = \boldsymbol{0}\}.$$

See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: NULLSVD

7.4 NULLSVD

Null space of a Matrix Command: Computes an orthonormal basis for the null space of any $n \times p$ matrix.

Level 1	$\rightarrow$	Level 1
$[[matrix]]_A$	$\rightarrow$	$[[matrix]]_{Null}$

Affected by: The variable TOL in the HOME directory

Remarks: Computes an orthonormal basis for the null space of any matrix using the Singular Value Decomposition, which sometimes produces more accurate results than NULLQR, but can take much longer. See RANKQR or RANKSVD on the use of TOL, and on the computation of rank($\boldsymbol{A}$).

Related Commands: NULLQR.

Index