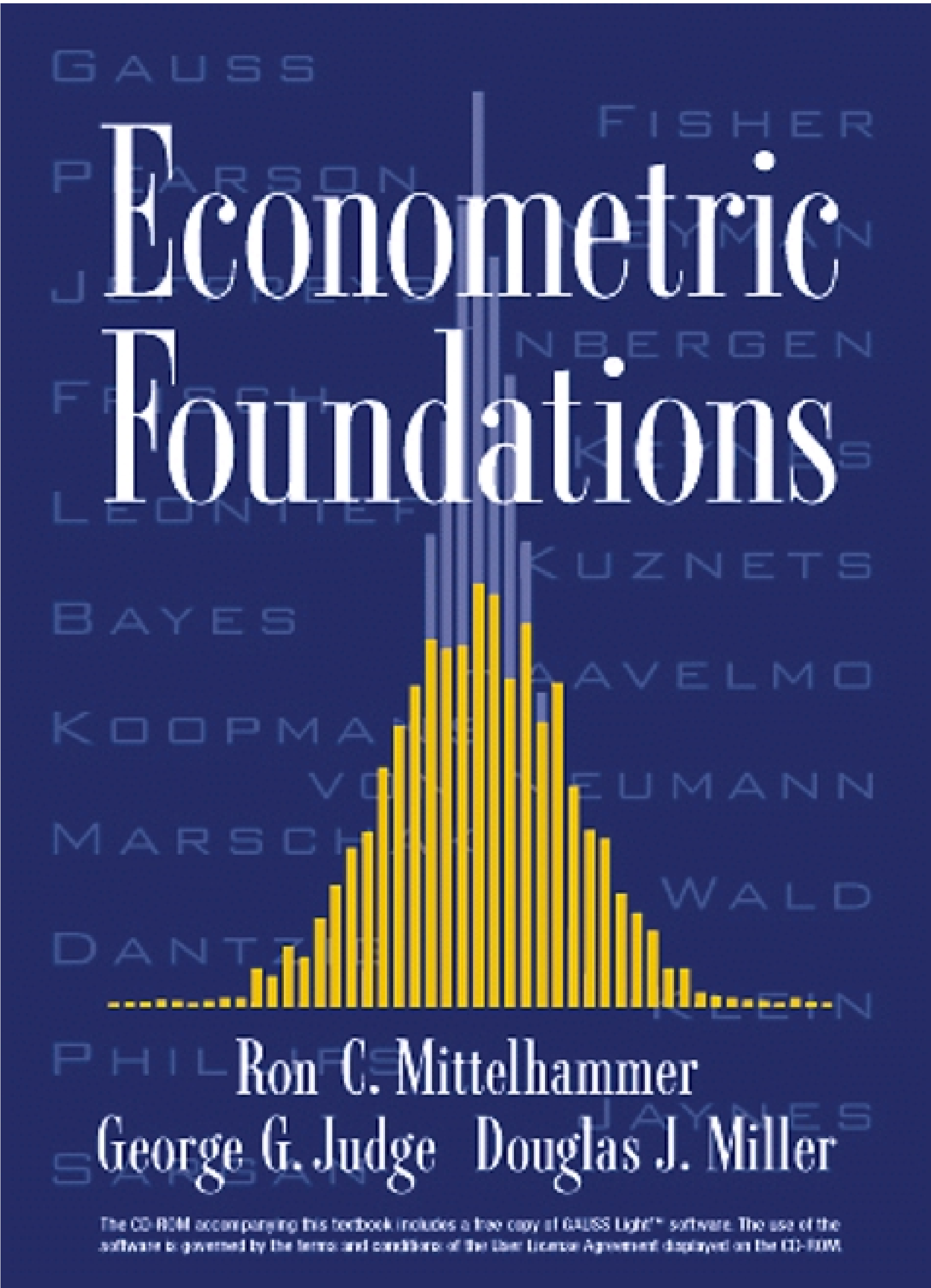
Matrix Review with GAUSS
All Matrix Topics View



Econometric Foundations

Ron C. Mittelhammer

George G. Judge Douglas J. Miller

The CD-ROM accompanying this textbook includes a free copy of GAUSS Light™ software. The use of the software is governed by the terms and conditions of the User License Agreement displayed on the CD-ROM.

Add, Subtract, Hadamard Product, Division

Let **x** and **y** be two matrices having identical dimension n by k.

Addition:

The addition of the two matrices **x** and **y**, denoted by **x** + **y** or equivalently **y** + **x**, results in the n by k matrix, say **z**, whose (i,j)th element is given by z[i,j] = x[i,j] + y[i,j], i.e., the elements of the matrices **x** and **y** are added elementwise.

Subtraction:

The subtraction of the matrix **y** from the matrix **x**, denoted by **x** - **y**, results in the n by k matrix, say **z**, whose (i,j)th element is given by z[i,j] = x[i,j] - y[i,j], i.e., the elements of the matrices are subtracted elementwise.

Multiplication:

The *elementwise* multiplication of the matrix **x** by the matrix **y**, also referred to as the *Hadamard product*, is denoted by $\mathbf{x} \odot \mathbf{y}$, or equivalently by $\mathbf{y} \odot \mathbf{x}$. The resulting matrix, say **z**, has (i,j)th element z[i,j] = x[i,j]y[i,j], i.e., the elements of the matrices **x** and **y** are multiplied together elementwise.

Division:

The division of the matrix **x** by the matrix **y**, denoted by **x./y**, results in the n by k matrix, say **z**, whose (i,j)th element is given by **z**[i,j] = x[i,j]/y[i,j], i.e., the elements of the matrices are divided elementwise.

GAUSS command:

x + y	adds the matrices **x** and **y**
x - y	subtracts the matrix **y** from the matrix **x**
x.*y	Hadamard (elementwise) product of **x** and **y** having identical row and column dimensions**.** Also note that this command can be used to "sweep multiply" a vector **y** elementwise through each of the columns in **x**. If the column vector **y** has the same row dimension as the matrix **x**, then the resultant matrix is a matrix having the same dimension as **x** with each column of **x** elementwise multiplied by the respective entries in **y**. If **y** is a row vector having the same number of columns as **x,** then each column of **x** is scalar multiplied by the respective scalar in **y**.

x./y Divides the matrix **x** by the matrix **y.** Also note that this command can be used to "sweep divide" a vector **y** elementwise through each of the columns in **x**. If the column vector **y** has the same row dimension as the matrix **x**, then the resultant matrix is a matrix having the same dimension as **x** with each column of **x** elementwise divided by the entries in **y**. If **y** is a row vector having the same number of columns as **x,** then each column of **x** is scalar divided by the respective scalar in **y**.

Bilinear Forms

A *bilinear form* in the n by 1 vector **x** and m by 1 vector **y** is defined as the matrix multiplication **y'qx**. The m by n matrix **q** is known as the *matrix of the bilinear form*. In scalar notation, the bilinear form can be represented as:

$$\mathbf{y}'\mathbf{q}\mathbf{x} = \sum_{i=1}^{m}\sum_{j=1}^{n} q_{ij} y_i x_j$$

The form is called a *bilinear* form because it is both linear in **x**, given a value of **y**, and linear in **y**, given a value of **x**.

A bilinear form for which both **x** and **y** have the same dimension is said to be symmetric if **y'qx = x'qy** for all **y** and **x.** A bilinear form is symmetric iff **q** = **q'**. A symmetric bilinear form for which **y** = **x** is called a *quadratic form*.

GAUSS command:

The value of a bilinear form can be calculated in GAUSS by using the syntax **y'q*x**.

Cholesky Decomposition

The Cholesky decomposition of a symmetric positive definite matrix **x** is sometimes thought of as involving a matrix square root of **x**, although the analogy to the square root of a scalar quantity is not as direct as in the case of the symmetric matrix square root. The Cholesky decomposition of the symmetric positive definite matrix **x** is defined in terms of an upper triangular matrix , say **c**, called the Cholesky factor, which is such that **x** = **c'c**. By upper triangular, we mean that only the elements on and above the diagonal of **c** have nonzero values. The Cholesky factor, **c**, and the Cholesky decomposition, **x** = **c'c**, always exist so long as **x** is symmetric and positive definite.

GAUSS command:

chol(x) will calculate the Cholesky factor, say **c**, of the symmetric positive definite matrix **x** = **c'c**.

Column Counts

In order to count up the number of elements in a n by 1 column vector that fall into various range categories of the form $a < x \le b$, one uses the concept of *column counts*. In particular, the column count for the n by 1 vector, **z**, based on the m by 1 vector of ranges, **v**, is the m by 1 vector representing the number of elements that fall within the following ranges:

$$\begin{aligned} z &\le v[1] \\ v[1] < z &\le v[2] \\ v[2] < z &\le v[3] \\ &\vdots \\ v[m-1] < z &\le v[m] \end{aligned}$$

GAUSS command:

counts(x,v) calculates the number of elements in the vector **x** that fall within the interval ranges defined by the break points given in the m by 1 vector **v**, returned as an m by 1 column vector. The value + infinity is an admissible choice for the upper break point of the final interval range (+ infinity can be set by letting v[m] = 1e500, or any number exceeding the maximum value storable on the computer). The vector **v** *must* be sorted in ascending order.

Column and Row Dimension

The row and column *dimensions* of a matrix are the number of rows and the number of columns in the matrix, respectively.

GAUSS command:

rows(x) returns the number of rows contained in the matrix **x.**

cols(x) returns the number of columns contained in the matrix **x**.

Column and Row Product

The value obtained by multiplying together all of the elements in a given column or row of a matrix is called a column or row product, respectively.

GAUSS command:

> **prodc(x)** calculates the column product of each column of the n by k matrix, **x**, and returns the values of these products, in column order, as a k by 1 column vector. The row products of the matrix **x** can be obtained by the command **prodc(x')**.

Column and Row Sum

The value obtained by summing all of the elements in a given column or row of a matrix is called a column or row sum, respectively.

GAUSS command:

> **sumc(x)** calculates the column sum of each column of the n by k matrix, **x**, and returns the values of these sums, in column order, as a k by 1 column vector. The row sums of the matrix **x** can be obtained by the command **sumc(x')**.

Concatenation or Merging

Vectors and matrices can be *concatenated* or, equivalently, *merged* either *horizontally*, or *vertically*. Horizontal concatenation means that two vectors and/or matrices are transformed into one matrix by appending them to each other side-by-side or left-to-right. This requires that the vectors or matrices being concatenated each have the same row dimension. Horizontal concatenation of **x** and **y** is denoted by **x~y**. Vertical concatenation means that two vectors and/or matrices are transformed into one matrix by appending them together from top-to-bottom, or one atop the other. This requires that the vectors or matrices being concatenated each have the same column dimension. Vertical concatenation of **x** and **y** is denoted by **x|y**.

GAUSS command:

x~y	produces a matrix representing the horizontal concatenation of **x** and **y**
x|y	produces a matrix representing the vertical concatenation of **x** and **y**

Condition Number and Linear Dependence Detection

The condition number of a matrix is equal to the ratio of the largest to the smallest *singular values* of a matrix (see the *singular value decomposition* topic in the table of contents). That is, if **x** is a n by k matrix, then the condition number of the matrix is defined to be:

$\kappa(\mathbf{x}) = \frac{\lambda_L}{\lambda_S}$, where λ_L and λ_S are the largest and smallest singular values of **x**.

In the special case where the matrix **x** is symmetric and positive definite, the singular values of **x** are equal to the eigenvalues of **x**, in which case the condition number can be defined equivalently as the ratio of the largest to the smallest eigenvalues of **x**. Note that some authors define the condition number as being equal to the *square root* of the condition number as it is defined here.

The condition number is often used as a measure of the degree to which the columns of data in a data matrix **x** are linearly related or multicollinear. The larger the condition number, the more linearly related or multicollinear are the columns of data in **x**. The polar case is when the smallest singular value of **x** is zero, in which case the condition number is infinity, and the columns of **x** are perfectly linearly related.

Note that the the vectors contained in the left and right singular vectors of the singular value decomposition (SVD) of the **x** matrix will indicate the nature of the linear dependency amongst the columns of **x**. In particular, recall that the SVD of the matrix **x** is given by:

$$\mathbf{x} = \sum_{i=1}^{r} s_i \mathbf{u}[.,i]\mathbf{v}[.,i]'$$

where r is the rank of **x**, s_i is the ith nonzero singular value of **x**, and **u** and **v** are the orthogonal left and right matrices of singular vectors, respectively. The columns of the left and right matrices of singular values are respectively associated with the singular values, from i = 1, ..., r. Now suppose that the jth singular value of **x** is equal to zero. Then note that we can identify both a linear combination of the columns of **x** and of the rows of **x** that equal zero, indicating the nature of the linear dependency among the columns or rows of **x**. In particular, it follows from the orthogonality of the matrices of singular vectors, and from the assumption $s_j = 0$, that

$$\mathbf{x}\mathbf{v}[.,j] = \sum_{i=1}^{r} s_i \mathbf{u}[.,i]\mathbf{v}[.,i]' \mathbf{v}[.,j] = s_j \mathbf{u}[.,j] = \mathbf{0}$$

and

$$\mathbf{u}[.,j]'\,\mathbf{x} = \sum_{i=1}^{r} s_i \mathbf{u}[.,j]'\,\mathbf{u}[.,i]\,\mathbf{v}[.,i]' = s_j \mathbf{v}[.,j]' = \mathbf{0}$$

which indicates linear combination of the columns and the rows of **x**, respectively, that exhibit a linear dependency.

Belsley, Kuh, and Welsh (*Regression Diagnostics: Identifying Influential Data and Sources of Collinearity*, New York:Wiley, 1980) have suggested "rules of thumb" for when the size of the condition number of a matrix suggests a situation where linear dependency/multicollinearity in the **x** matrix will induce problems in a standard linear regression analysis vis-a-vis inflated variances as well as potential invertibility problems with respect to **x'x**. In establishing these rules, they first normalize each of the columns of **x** to unit length by dividing the ith column of **x** by square root of the inner product of the column of data, i.e., they divide the ith column by the scalar $(\mathbf{x}_i'\mathbf{x}_i)^{1/2}$. Belsley, et. al. suggest that a condition number of 20 or greater suggests that a linear dependency problem may adversely effect linear regression results. They also suggest that values in excess of 100 are generally causes for concern. We note that these are only rules of thumb, and in fact results of a linear regression analysis may still be useful, in the sense that estimated parameters are relatively precisely estimated, even when a condition number is "high" by these rules of thumb. Note this can occur when the noise component variance is small, counteracting the effect of inflated diagonal elements of $(\mathbf{x}'\mathbf{x})^{-1}$. See Belsley, et. al., for more information on the relationship between the value of the condition number and the effect on the results of linear regression analysis.

GAUSS command:

The condition number of a matrix, **x**, can be calculated using the GAUSS command **cond(x)**. Alternatively, one could calculate the condition number by obtaining the singular values of the matrix **x** using the singular value calculation procedure **svd(x)**, or by extracting the eigenvalues of the matrix **x** in the case where **x** is symmetric and positive semidefinite using **eigh(x)**, and then forming the appropriate ratio.

To calculate the left and right singular matrices whose column vector entries can be used to identify the nature of linear dependencies in the **x** matrix, the GAUSS command **{u,s,v}=svd1(x)** can be employed. See the topic "singular value decomposition" for further information on this decomposition. RUN the GAUSS application to see an example of how the condition number can be calculated, and how the SVD can be used to detect the form of linear dependencies in **x**.

Correlation Matrix

Letting **x** be a n by k matrix, the sample correlation matix is the k by k matrix whose (i,j)th element is the sample correlation based on the ith and jth columns of data, **x**[.,i] and

x[.,j], in **x**. The (i,j)th element is calculated as

$$\frac{\sum_{k=1}^{n}\left(x[k,i]-\bar{x}[.,i]\right)\left(x[k,j]-\bar{x}[.,j]\right)}{\left[\sum_{k=1}^{n}\left(x[k,i]-\bar{x}[.,i]\right)^2\sum_{k=1}^{n}\left(x[k,j]-\bar{x}[.,j]\right)^2\right]^{1/2}}$$

where the mean value of the elements in the ith column of **x** is denoted by

$$\bar{x}[.,i].$$

GAUSS command:

> **corrx(z)** calculates the k by k sample correlation matrix based on the data in the n by k matrix **z**.
>
> **corrvc(w)** calculates the k by k sample correlation matrix based on the information contained in the k by k covariance matrix represented by **w**.

Covariance Matrix (Sample)

Letting **x** be a n by k matrix, the sample covariance matrix is the k by k matrix whose (i,j)th element is the sample covariance based on the ith and jth columns of data, **x**[.,i] and **x**[.,j], in **x**. The diagonal of this matrix represents the respective sample variances of the columns of **x**. The (i,j)th element is calculated as

$$\frac{1}{n-1}\sum_{k=1}^{n}\left(x[k,i]-\bar{x}[.,i]\right)\left(x[k,j]-\bar{x}[.,j]\right)$$

Note that this form of the sample covariance matrix is an unbiased estimate of the population covariance matrix since it incorporates a degrees of freedom correction, using the divisor n-1. The standard sample covariance matrix estimate based on the empirical distribution function uses a divisor of n rather than n-1. In large samples, the choice of the divisor is immaterial.

GAUSS command:

> **vcx(z)** calculates the k by k (unbiased) sample covariance matrix based on the data in the n by k matrix **z**.

Cross Product Matrix

Letting **x** be a n by k matrix of data, the k by k cross product matrix based on **x** is defined as **x'x**.

GAUSS command:

moment(x,d) will calculate the k by k cross product matrix **x'x**, where d = 0 implies no missing values are checked for, and d = 1 will eliminate any row of **x** from the cross product calculation for which a missing value exists. This command takes into account that **x'x** is symmetric when calculating the matrix, so execution time is approximately halved compared to using the command **x'x**, which is another way to calculate the cross product matrix.

Defining Matrix Elements

An n by k matrix contains nk numbers displayed in the form of n rows and k columns. Each of the nk numbers, or elements, in a matrix are located by a row and column identifier. For example, the 3rd entry in a column vector **x** is identified by x[3], and the notation x[2,3] would indicate the entry in a matrix that resides at the intersection of the 2nd and 3rd column. If **x** is a n by k matrix, then the i^{th} row of the matrix is denoted by **x**[i,.] and the j^{th} column of the matrix is denoted by **x**[.,j], where i and j are some integer values less than or equal to the values of n and k, respectively.

GAUSS Command:

Let x[n,k] = a_1 a_2 a_3...a_{nk}; reads in the nk numbers a_1, a_2,...,a_{nk} *in row order*, i.e., the first k entries in row 1, then the next k entries in row 2, etc.

x[i,j] is the $(i,j)^{th}$ entry in **x**

x[i,.] is the i^{th} row of **x**

x[.,j] is the j^{th} column of **x**

x[j] is the j^{th} entry in the column vector **x**.

Load x[n,k]=a:\mydir\mydata.asc; will load an n by k matrix of numbers from a file called mydata.asc (use your own file name, but use the .asc extension) on drive A (use your own drive letter) in directory mydir (enter the appropriate directory). The data is read in row by row, beginning with the first row, and the file mydata.asc contains at least nk numbers to fill in the elements of the **x** matrix.

Loadm x=c:\mydir\mydata; will load the GAUSS matrix file mydata.fmt (fmt is the standard extension for GAUSS matrix files and will be implicitly understood if not supplied) that resides in the directory mydir on the C drive into the matrix **x**.

Save/mydir/mydata=x; will save the matrix **x** in the directory mydir as the GAUSS matrix file mydata.fmt.

Also see the command **readr(f1,r)** in the GAUSS online Help file.

Definite and Semidefinite Matrices

Let **x** be a n by n symmetric matrix, and let **c** be a n by 1 column vector. Then the following definitions characterize whether **x** is a definite or semidefinite matrix:

Positive Definite (PD):

$$\mathbf{c}'\mathbf{x}\mathbf{c} > 0, \quad \forall\ \mathbf{c} \neq \mathbf{0}$$

Positive Semidefinite (PSD):

$$\mathbf{c}'\mathbf{x}\mathbf{c} \geq 0, \quad \forall\ \mathbf{c}$$

Negative Definite (ND):

$$\mathbf{c}'\mathbf{x}\mathbf{c} < 0, \quad \forall\ \mathbf{c} \neq \mathbf{0}$$

Negative Semidefinite (NSD):

$$\mathbf{c}'\mathbf{x}\mathbf{c} \leq 0, \quad \forall\ \mathbf{c}$$

Thus, a symmetric matrix **x** will be positive definite (semidefinite) iff the sign of the quadratic form **c'xc** is always positive (nonnegative) regardless of the choice of the nonzero vector **c**. The symmetric matrix **x** is negative definite (semidefinite) iff the quadratic form **c'xc** is always negative (nonpositive) regardless of the choice of the nonzero vector **c**.

Properties:

1. For any matrix **z**, **z'z** is PSD, and is also PD if **z** has full column rank

2. If **x** is PD (PSD), then **x** is nonsingular (singular).

3. If **x**, **z**, and **x**-**z** are all PD then $\mathbf{z}^{-1} - \mathbf{x}^{-1}$ is PD and det (**x**) > det(**z**).

4. The diagonal elements of a PD (PSD) matrix **x** are all positive (nonnegative).

5. The trace of a PD (PSD) matrix **x** is positive (nonnegative).

6. If **x** is PD (PSD), then -**x** is ND (NSD).

GAUSS command:

deftest(x) will test to see whether the symmetric matrix **x** is PD, PSD, ND, NSD, or not definite, and will print the outcome of the test, if the following GAUSS procedure is contained in the GAUSS program:

```
proc(0)=deftest(x);
Local va;
va = eigh(x);
if va > 0;
        print "Matrix is PD";
elseif va > = 0;
        print "Matrix is PSD";
elseif va < 0;
        print "Matrix is ND";
elseif va <= 0;
        print "Matrix is NSD";
else;
        print "Matrix is Not Definite";
endif;
retp;
endp;
```

Note: Numerical accuracy can prevent the proper detection of definiteness when nonsingular matrices are very close to being singular.The fuzzy comparison operators **fgt**, **fge**, **flt**, and **fle** can replace the standard comparison operators in the above proc to produce fuzzy comparisons to within 1e-15. See the on-line GAUSS help discussion of fuzzy comparison operators.

Determinant

The determinant of a square matrix **x**, denoted by det(**x**), is a scalar that indicates the degree to which the columns or rows of **x** are linearly related. The closer det(**x**) is to zero, the more nearly linearly related are two or more rows or columns of **x**, and det(**x**)=0 indicates a linear dependence among the columns or rows in **x**, in which case the matrix is said to be *singular*. Geometrically, the value of det(**x**) equals the volume contained within the "box" defined by rays from the origin represented by the column (or row) vectors of **x**, a zero volume coinciding with a linear dependence among these rays. The value of det(**x**) can be computed in a number of different ways. One method is given in property 1 below.

Properties:

1. $\det(\mathbf{x}) = \prod_{i=1}^{n} \lambda_i$, where λ_i is the ith eigenvalue of **x**.

2. $\det(\mathbf{x}^{-1}) = [\det(\mathbf{x})]^{-1}$

3. $\mathbf{x}^{-1}$ exists iff $\det(\mathbf{x}) \neq 0$

4. $\det(\mathbf{xy}) = \det(\mathbf{x})\det(\mathbf{y})$

5. $\det(\mathbf{x}) = \det(\mathbf{x}')$

GAUSS command:

det(x)

Diagonal Matrix

A diagonal matrix is a matrix for which the only nonzero elements occur along the diagonal of the matrix. The identity matrix is a special type of diagonal matrix having all ones along its diagonal and zeros elsewhere.

GAUSS command:

diag(x) extracts the diagonal of a matrix **x** and forms a column vector containing the diagonal elements.

diagrv(x,v) inserts the column vector **v** into the diagonal of the matrix **x**. If **x** is an identity matrix, then this command forms a diagonal matrix with the elements of **v** as the elements of the diagonal.

Diagonalization of Matrices

If **x** is any symmetric positive definite matrix, it can be diagonalized (transformed into a diagonal matrix) by pre and post multiplication using **x**'s eigenvector matrix, **p**, as **p'xp** = Λ. Here Λ is the diagonal matrix having **x**'s eigenvalues (respectively associated with the columns of eigenvectors in **p**) along its diagonal.

Since **p'p** = **pp'** = **I**, the diagonalization can be reversed to represent **x** in terms of eigenvalues and eigenvectors as **x** = **p**Λ**p'** .

Diagonalization of matrices can be accomplished in GAUSS by extracting the eigenvalues and eigenvectors of **x** and then performing the appropriate matrix multiplications as described above. RUN the GAUSS code to see how it is done.

Direct or Kronecker Product

Let **x** be n by k and **y** be m by p. The direct or Kronecker product of **x** and **y**, denoted by

$$\mathbf{x} \otimes \mathbf{y}$$

is the nm by kp matrix defined as follows:

$$\mathbf{z} = \begin{bmatrix} x_{11}\mathbf{y} & x_{12}\mathbf{y} & \cdots & x_{1k}\mathbf{y} \\ x_{21}\mathbf{y} & x_{22}\mathbf{y} & \cdots & x_{2k}\mathbf{y} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1}\mathbf{y} & x_{n2}\mathbf{y} & \cdots & x_{nk}\mathbf{y} \end{bmatrix}$$

Properties: (Assuming matrix conformability)

1. $(\mathbf{w} \otimes \mathbf{x})(\mathbf{y} \otimes \mathbf{z}) = \mathbf{wy} \otimes \mathbf{xz}$
2. $(\mathbf{x} \otimes \mathbf{z})' = (\mathbf{x}' \otimes \mathbf{z}')$
3. $\mathbf{x} \otimes (\mathbf{y} \otimes \mathbf{z}) = (\mathbf{x} \otimes \mathbf{y}) + (\mathbf{x} \otimes \mathbf{z})$
4. $(\mathbf{x} + \mathbf{y}) \otimes \mathbf{z} = (\mathbf{x} \otimes \mathbf{z}) + (\mathbf{y} \otimes \mathbf{z})$
5. $\mathbf{x} \otimes (\mathbf{y} \otimes \mathbf{z}) = (\mathbf{x} \otimes \mathbf{y}) \otimes \mathbf{z}$
6. If **x** is m by m and **y** is n by n, then:

   a) $\operatorname{tr}(\mathbf{x} \otimes \mathbf{y}) = \operatorname{tr}(\mathbf{x})\operatorname{tr}(\mathbf{y})$

   b) $\det(\mathbf{x} \otimes \mathbf{y}) = [\det(\mathbf{x})]^{n}[\det(\mathbf{y})]^{m}$

GAUSS command:

x.*.y generates the Kronecker product of **x** and **y**.

Eigenvalues & Eigenvectors

A scalar eigenvalue (or characteristic root), λ_i , and eigenvector (or characteristic vector), $\mathbf{p}_i$, of a square matrix **x** occur in pairs, and are characterized by the fact that they satisfy the equation $\mathbf{x}\,\mathbf{p}_i = \lambda_i\,\mathbf{p}_i$. There are as many eigenvalues and corresponding eigenvectors as there are rows or columns in the square matrix **x**. Thus, if **x** is an m by m matrix, then there will be m pairs of eigenvalues and eigenvectors. Letting **p** be the m by m matrix whose columns are the m eigenvectors of **x**, and letting Λ represent a diagonal matrix having the respective eigenvalues of x along the diagonal, it follows that $\mathbf{x}\,\mathbf{p} = \mathbf{p}\,\Lambda$.

<u>General (All) Matrices:</u>

1. The trace of matrix **x** is equal to the sum of its eigenvalues, i.e., $\sum_{i=1}^{m}\lambda_i = \mathrm{tr}(\mathbf{x})$.
.
2. The determinant of matrix **x** is equal to the product of its eigenvalues, $\prod_{i=1}^{m}\lambda_i = \det(\mathbf{x})$.
3. $\det(\mathbf{x} - \lambda\mathbf{I}) = 0$
4. The eigenvalues of the inverse of **x** are equal to the recipricols of the eigenvalues of **x**. The eigenvectors of $\mathbf{x}^{-1}$ and **x** are identical.

<u>Symmetric Matrices:</u>

5. The eigenvalues and eigenvectors of a symmetric matrix are real-valued. This need not be the case for nonsymmetric matrices.
6. The eigenvectors of a symmetric matrix can always be defined to have unit length and to be orthogonal to one another; the latter property does not necessarily apply to nonsymmetric matrices.
7. The rank of a symmetric matrix is equal to the number of nonzero eigenvalues of the matrix.
8. A symmetric matrix **x** is positive definite (positive semidefinite, negative definite, or negative semidefinite) iff all of its eigenvalues are positive (nonnegative, negative, or nonnegative).

GAUSS command:

eig(x) for eigenvalues of general matrix x, returned as a column vector.

{va,ve}= eigv(x) for both eigenvalues,va, and eigenvectors,ve, of a general matrix x, returned as a column vector and a matrix whose columns are the corresponding eigenvectors, respectively.

eigh(x) for eigenvalues of a symmetric matrix x, returned as a column vector (faster, and more accurate when it applies).

{va,ve}=eighv(x) for both eigenvalues,va, and eigenvectors,ve, of a symmetric matrix x, returned as a column vector and a matrix whose columns are the corresponding eigenvectors, respectively (faster and more accurate when it applies).

Elementwise Square Root

There are a number of concepts that all relate to the idea of a square root operation applied to a matrix. The simplest of these is the *elementwise square root* operation, whereby the square root operation is applied to each and every element in a matrix **x**. This results in a matrix, say **z**, such that $z[i,j] = (x[i,j])^{1/2}$ for every i,j. The elementwise square root matrix can be denoted by the notation

$$\sqrt{\mathbf{x}} \text{ or } (\mathbf{x})^{1/2}$$

GAUSS command:

x^(1/2) or **sqrt(x)** will calculate the elementwise square root of the matrix **x**.

Generalized Inverse

For any m by n matrix **x**, there exists a unique generalized inverse matrix, denoted by $\mathbf{x}^{-}$, that has the following four properties:

$$\mathbf{x}\mathbf{x}^{-}\mathbf{x} = \mathbf{x}\,,\;\; \mathbf{x}^{-}\mathbf{x}\mathbf{x}^{-} = \mathbf{x}^{-},\;\; (\mathbf{x}\mathbf{x}^{-})^{-} = \mathbf{x}\mathbf{x}^{-},\;\; (\mathbf{x}^{-}\mathbf{x})^{-} = \mathbf{x}^{-}\mathbf{x}\,.$$

<u>Properties:</u>

1. If **x** is m by m and also has full rank , then $\mathbf{x}^{-} = \mathbf{x}^{-1}$.

2. If **x** is idempotent , then **x** is its own generalized inverse, $\mathbf{x} = \mathbf{x}^{-}$.

3. If **x** has full column rank, then $\mathbf{x}^{-} = (\mathbf{x}'\mathbf{x})^{-1}\,\mathbf{x}'$

4. $(\mathbf{x}^{-})' = (\mathbf{x}')^{-}$

5. $(\mathbf{x}^{-})^{-} = \mathbf{x}$

6. The rank of $\mathbf{x}$ is equivalent to the rank of $\mathbf{x}^{-}$.

7. $(\mathbf{x}\mathbf{x}^{-})^{-} = \mathbf{x}\mathbf{x}^{-}$ and $(\mathbf{x}^{-}\mathbf{x})^{-} = \mathbf{x}^{-}\mathbf{x}$

GAUSS command:

pinv(x)

Gradients, Hessians, and Jacobians

Gradient Vector

Let f(**z**) be a scalar-valued function of the k by 1 column vector **z**. The *gradient vector* of f(**z**) with respect to **z** is the k by 1 column vector of first derivatives of f(**z**), i.e.,

$$\cdot\frac{\partial f(\mathbf{z})}{\partial \mathbf{z}} = \left[\frac{\partial f(\mathbf{z})}{\partial z_1} \quad \frac{\partial f(\mathbf{z})}{\partial z_2} \quad \cdots \quad \frac{\partial f(\mathbf{z})}{\partial z_k}\right]'$$

Hessian Matrix

The *Hessian matrix* of the scalar-valued function f(**z**) with respect to the k by 1 column vector **z** is the k by k matrix of second order derivatives of f(**z**), i.e.,

$$\frac{\partial^2 \mathbf{f}(\mathbf{z})}{\partial \mathbf{z}\partial \mathbf{z}'} = \begin{bmatrix} \frac{\partial^2 f(\mathbf{z})}{\partial z_1^2} & \frac{\partial f(\mathbf{z})}{\partial z_1 \partial z_2} & \cdots & \frac{\partial f(\mathbf{z})}{\partial z_1 \partial z_k} \\ \frac{\partial f(\mathbf{z})}{\partial z_2 \partial z_1} & \frac{\partial^2 f(\mathbf{z})}{\partial z_2^2} & \cdots & \frac{\partial f(\mathbf{z})}{\partial z_2 \partial z_k} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f(\mathbf{z})}{\partial z_k \partial z_1} & \frac{\partial f(\mathbf{z})}{\partial z_k \partial z_2} & \cdots & \frac{\partial^2 f(\mathbf{z})}{\partial z_k^2} \end{bmatrix}$$

Jacobian Matrix

If **f(z)** = [f_1(**z**), f_2(**z**),...,f_m(**z**)]' is a m by 1 vector valued function of the k by 1 vector **z**, then the *Jacobian* matrix of **f**(**z**) with respect to **z** is given by the k by m matrix

$$\frac{\partial \mathbf{f}(\mathbf{z})}{\partial \mathbf{z}} = \begin{bmatrix} \frac{\partial f_1(\mathbf{z})}{\partial z_1} & \cdots & \frac{\partial f_m(\mathbf{z})}{\partial z_1} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_1(\mathbf{z})}{\partial z_k} & \cdots & \frac{\partial f_m(\mathbf{z})}{\partial z_k} \end{bmatrix}$$

GAUSS command:

gradp(&f, z0) will return the *transpose* of the k by 1 gradient vector of the function f(**z**) evaluated at the k by 1 vector value of **z0** if f is a scalar-valued function. If **f**(**z**) is m by 1, then **gradp(&f, z0)** will return the *transpose* of the Jacobian matrix evaluated at **z0.** Note the symbol & is required, but the symbol f can be changed to any other set of characters that are used to denote a procedure (see below) that defines the function.

```
proc f(z);
local f1;
f1 = (z[1]^ 2) + (z[2] * z[3]);
retp(f1);
endp;
```

Hessp(&f,z0) will return the Hessian matrix of a scalar function f(**z**) evaluated at the k by 1 vector value of **z0** (approximately 3-4 digits of accuracy can be expected). Note the symbol & is required, but the symbol f can be changed to any other set of characters that are used to denote a procedure (see below) that defines the function.

Both preceding commands require that the function **f**(**z**) be defined via a GAUSS procedure. To illustrate such a procedure, consider the function

$$\mathbf{f}(\mathbf{z}) = \begin{bmatrix} f_1(\mathbf{z}) \\ f_2(\mathbf{z}) \end{bmatrix} = \begin{bmatrix} z_1^2 + z_2 z_3 \\ z_1 z_2 \end{bmatrix}$$

This vector function could be represented in a procedure thus:

```
proc f(z);
local f1, f2;
f1 = (z[1]^ 2) + (z[2] * z[3]);
f2 = z[1] * z[2];
retp(f1|f2);
endp;
```

Hadamard Product

Let **x** and **y** be two matrices having identical dimension n by k. The *elementwise* multiplication of the matrix **x** by the matrix **y**, also referred to as the *Hadamard product*, is denoted by

$$\mathbf{x} \odot \mathbf{y}, \text{ or equivalently by } \mathbf{y} \odot \mathbf{x}.$$

The resulting matrix, say **z**, has (i,j)th element z[i,j] = x[i,j]y[i,j], i.e., the elements of the matrices **x** and **y** are multiplied together elementwise.

GAUSS command:

x.*y Hadamard (elementwise) product of **x** and **y** having identical row and column dimensions**.** Also note that this command can be used to "sweep multiply" a vector **y** elementwise through each of the columns in **x**. If the column vector **y** has the same row dimension as the matrix **x**, then the resultant matrix is a matrix having the same dimension as **x** with each column of **x** elementwise multiplied by the respective entries in **y**. If **y** is a row vector having the same number of columns as **x,** then each column of **x** is scalar multiplied by the respective scalar in **y**.

Idempotent Matrices

A square n by n matrix is idempotent if $\mathbf{x} = \mathbf{x}^2 = \mathbf{xx}$, and is symmetric idempotent if **x** is idempotent and **x** = **x'**.

<u>Properties</u>:

1. If **x** is an n by n idempotent matrix with rank r less than or equal to n, then **x** has r eigenvalues equal to 1 and n-r eigenvalues equal to zero.

2. If **x** is symmetric idempotent, then **x** is postive semidefinite .

3. If **x** is symmetric, then it is idempotent iff all of the nonzero eigenvalues of **x** equal 1.

4. If **x** is idempotent, then **I**-**x** and **x'** are also idempotent.

GAUSS command:

A check for idempotency can be undertaken by comparing the matrix **x** with the matrix **x*x** to see if these matrices are the same. We caution however that matrix multiplication roundoff error can cause problems in the comparison, and one should only expect that **x** and **x*x** will agree to 14-15 or so decimals of accuracy. In addition, if **x** is symmetric, then one could check to see whether all of the eigenvalues take only the values 1 and 0 (again, at least to within 14-15 decimals of accuracy).

Identity Matrix

An identity matrix of dimension n by n is a diagonal matrix, denoted by **I**, whose diagonal elements all equal the same value, 1. The identify matrix is the *identity element* for matrix multiplication, i.e., **x** = **Ix** or **x** = **xI**, analogous to the number 1 being the *identity element* for scalar multiplication, i.e., c = 1c = c1.

GAUSS command:

eye(n) creates an n-dimensional identity matrix.

Matrix Differentiation

Just as there are rules and formulae that can be applied for differentiating expressions involving scalar variables, there are also rules and formulae that can be applied for differentiating matrix expressions with respect to vectors, and matrices, of variables. These rules and formulae are referred to as "matrix differentiation", and are contained under the broader category of "matrix calculus". There are a myriad of rules and formulae for differentiating matrix expressions, and we do not attempt to review them all here. Instead we review some of the more often used rules and formulae. All of the rules and formulae can be verified by applying standard rules for differentiating functions of multiple scalar variables, and then reconstituting the results in terms of matrix expressions.

Excellent sources of additional information relating to matrix calculus are the books by Harville (1997, chapter 15) and Graybill (1983, chapter 10). See the "INFO" button on the opening page of this manual for the full references.

1. Differentiation of Linear Forms

a. $\frac{\partial \mathbf{c}'\mathbf{x}}{\partial \mathbf{x}} = \mathbf{c}$, where **c** is a conformable vector

b. $\frac{\partial \mathbf{qx}}{\partial \mathbf{x}} = \mathbf{q}'$, where **q** is a conformable matrix

c. $\frac{\partial \mathbf{x}'\mathbf{q}}{\partial \mathbf{x}} = \mathbf{q}$, where **q** is a conformable matrix or vector

d. $\frac{\partial \mathbf{x}}{\partial \mathbf{x}'} = \frac{\partial \mathbf{x}'}{\partial \mathbf{x}} = \mathbf{I}$

2. Differentiation of Quadratic Forms

a. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}} = 2\mathbf{q}\mathbf{x}$, if **q** is a symmetric matrix

b. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}} = (\mathbf{q}+\mathbf{q}')\mathbf{x}$

c. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}\partial \mathbf{x}'} = 2\mathbf{q}$, if **q** is a symmetric matrix

d. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}\partial \mathbf{x}'} = \mathbf{q}+\mathbf{q}'$

e. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{x}\mathbf{x}'$

f. $\frac{\partial \mathbf{x}'\mathbf{q}\mathbf{x}}{\partial \mathbf{q}} = 2\mathbf{x}\mathbf{x}' - \mathbf{D}_{\mathbf{x}\mathbf{x}'}$, when symmetry of **q** is maintained, $\mathbf{D}_{\mathbf{x}\mathbf{x}'}$ being a diagonal matrix containing the diagonal of $\mathbf{x}\mathbf{x}'$

We emphasize that when "symmetry is maintained", so that q[i,j] = q[j,i], it follows that q[j,i] cannot be held constant when q[i,j] is perturbed (or else symmetry is not maintained), and it is for this reason that e) and f) differ above, as well as ahead. In particular, in e) above, it is assumed that all of the entries in **q** are independent of one another, so that they can be independently perturbed in establishing derivatives.

3. Differentiation of Bilinear Forms

a. $\frac{\partial \mathbf{y}'\mathbf{q}\mathbf{x}}{\partial \mathbf{y}} = \mathbf{q}\mathbf{x}$

b. $\frac{\partial \mathbf{y}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{q}'\mathbf{x}$

c. $\frac{\partial \mathbf{y}'\mathbf{q}\mathbf{x}}{\partial \mathbf{x}} = \mathbf{y}\mathbf{x}'$

d. $\frac{\partial \mathbf{y}'\mathbf{q}\mathbf{x}}{\partial \mathbf{q}} = 2\mathbf{x}\mathbf{y}' - \mathbf{D}_{\mathbf{x}\mathbf{y}'}$, when symmetry of **q** is maintained, $\mathbf{D}_{\mathbf{x}\mathbf{y}'}$ being a diagonal matrix containing the diagonal of $\mathbf{x}\mathbf{y}'$

4. Differentiation of Determinants

a. $\frac{\partial \det(\mathbf{q})}{\partial \mathbf{q}} = (\mathbf{q}')^{-1} \det(\mathbf{q})$

b. $\frac{\partial \ln[\det(\mathbf{q})]}{\partial \mathbf{q}} = (\mathbf{q}')^{-1}$, when $\det(\mathbf{q}) > 0$

c. $\frac{\partial \det(\mathbf{q})}{\partial \mathbf{q}} = 2\mathbf{q}^{-1} \det(\mathbf{q}) - \mathbf{D}_{\mathbf{q}^{-1}\det(\mathbf{q})}$ when symmetry of **q** is maintained, where $\mathbf{D}_{\mathbf{q}^{-1}\det(\mathbf{q})}$ is a diagonal matrix containing the same diagonal elements as that of $\mathbf{q}^{-1}\det(\mathbf{q})$

d. $\frac{\partial \ln[\det(\mathbf{q})]}{\partial \mathbf{q}} = 2\mathbf{q}^{-1} - \mathbf{D}_{\mathbf{q}^{-1}}$ when symmetry of **q** is maintained, where $\mathbf{D}_{\mathbf{q}^{-1}}$ is a diagonal matrix containing the same diagonal as that of $\mathbf{q}^{-1}$, and $\det(\mathbf{q})$ is positive.

5. Differentiation of Trace

a. $\frac{\partial \mathrm{tr}(\mathbf{q})}{\partial \mathbf{q}} = \mathbf{I}$

b. $\frac{\partial \mathrm{tr}(\mathbf{qx})}{\partial \mathbf{x}} = \frac{\partial \mathrm{tr}(\mathbf{xq})}{\partial \mathbf{x}} = \mathbf{q}'$

c. $\frac{\partial \mathrm{tr}(\mathbf{qx})}{\partial \mathbf{x}} = \frac{\partial \mathrm{tr}(\mathbf{xq})}{\partial \mathbf{x}} = \mathbf{q} + \mathbf{q}' - \mathbf{D}_{\mathbf{q}}$ when symmetry of **x** is maintained, where $\mathbf{D}_{\mathbf{q}}$ is a diagonal matrix with the same diagonal as **q**

6. Differentiation of Inverse

a. $\frac{\partial \mathbf{q}^{-1}}{\partial q_{ij}} = -\mathbf{q}^{-1}\Delta_{ij}\mathbf{q}^{-1}$, where Δ_{ij} is a matrix of zeros, except for a 1 in the (i, j) position

b. $\frac{\partial \mathbf{q}^{-1}}{\partial q_{ij}} = -\mathbf{q}^{-1}\left(\Delta_{ij} + \Delta_{ji}\right)\mathbf{q}^{-1}$ when symmetry of **q** is maintained, where Δ_{ij} is defined as in 6a.

7. Differentiation of Matrix Sum and Product

a. $\frac{\partial \mathbf{cq}}{\partial x} = \mathbf{c}\frac{\partial \mathbf{q}}{\partial x} + \frac{\partial \mathbf{c}}{\partial x}\mathbf{q}$, where **c** and **q** are conformable matrices, and x is a scalar on which the elements of **c** and/or **q** depend.

b. $\frac{\partial(\mathbf{c}+\mathbf{q})}{\partial x} = \frac{\partial \mathbf{c}}{\partial x} + \frac{\partial \mathbf{q}}{\partial x}$, where **c** and **q** are conformable matrices, and x is a scalar on which the elements of **c** and/or **q** depend.

GAUSS command:

GAUSS can be used to calculate numerically the derivatives of all of the matrix expressions examine above, as well as any other differentiable matrix function. The rules and formulae displayed above represent exact analytical representations of the matrix derivatives. The numerical differentiation procedures in GAUSS can be used to illustrate or verify the validity of rules and formulae, and we do this in the GAUSS application associated with this matrix topic. Note that numerical first derivatives are generally accurate to 7-8 decimal places, while second order derivatives are generally accurate to 3-4 decimal places, depending on the complexity of the functions being differentiated. The primary differentiation procedures in GAUSS are the **gradp(&f, z0)** and **hessp(&f,z0)** procedures for first and second order differentiation, respectively. See the "Differentiation: Functions of Vectors" topic for more information.

Matrix Inverse

The (ordinary) inverse of a square matrix **x**, denoted by $\mathbf{x}^{-1}$, is a matrix that when multiplied into **x** results in the identity matrix . That is, $\mathbf{x}\,\mathbf{x}^{-1} = \mathbf{x}^{-1}\mathbf{x} = \mathbf{I}$. The inverse matrix always exists so long as the matrix **x** has full rank , which means that the columns and the rows of the matrix **x** are linearly independent vectors.

Properties:

1. The inverse of **x** will exist iff the determinant of **x** is nonzero.

2. The inverse of **x** will exist iff all of the eigenvalues of the matrix **x** are unequal to zero.

3. The inverse of an inverse matrix is equal to the original (noninverted) matrix, i.e., $(\mathbf{x}^{-1})^{-1} = \mathbf{x}$.

4. The inverse of the transpose of a matrix is equal to the transpose of the inverse of the matrix, i.e., $(\mathbf{x}')^{-1} = (\mathbf{x}^{-1})'$.

5. The inverse of a diagonal matrix , **x**, is equal to a diagonal matrix whose diagonal entries are the recipricols of the diagonal elements of **x**, i.e., the (i,i)th entry in the diagonal matrix **x** is equal to $(x[i,i])^{-1}$.

GAUSS command:

inv(x) for any square nonsingular matrix

invpd(x) if **x** is symmetric positive definite (faster)

Matrix Multiplication, Inner and Outer Products

Let **x** be an n by m matrix and **y** be a j by k matrix. The matrices **x** and **y** are said to be conformable for matrix multiplication **xy** if m=j, otherwise the matrices are said to be not conformable and this matrix multiplication cannot be applied. The matrix multiplication **xy** of conformable matrices **x** and **y** results in the matrix, say **z**, whose (i,j)th element is defined by

$$z[i,j] = \sum_{v=1}^{m} x[i,v]y[v,j]$$

<u>Inner Product of Vectors</u>:

The inner product of two n by 1 column vectors **x** and **y** is defined by the matrix multiplication **x'y** or equivalently **y'x**, producing a scalar outcome.

<u>Outer Product of Vectors</u>:

The outer product of an n by 1 column vector **x** by the k by 1 column vector **y** is defined by the matrix multiplication **x**(**y'**) or **y**(**x'**), resulting in a n by k or k by n matrix, respectively.

***GAUSS command*:**

x*y	matrix multiplication of **x** by **y**
x'y	inner product of vectors **x** and **y**
x*y'	outer product of vector **x** by vector **y**

Matrix Powers

Let r be a positive integer. Then the matrix **x** raised to the power r is defined as

$$\mathbf{x}^r = \prod_{j=1}^{r} \mathbf{x} = \mathbf{xx}\ldots\mathbf{x},$$

i.e., it is the matrix **x** multiplied by itself r-1 times. For example, $\mathbf{x}^2$ = **xx** and $\mathbf{x}^3$ = **xxx**.

GAUSS command:

matpower(x,r) will produce the value of $\mathbf{x}^r$ when the following procedure is included in a GAUSS program.

```
proc matpow(x,r);
local holdx;
holdx=x;
for(1, r-1, 1);
x = x*holdx;
endfor;
retp(x);
endp;
```

Mean Values (Sample)

The sample mean of a n by 1 vector of data, **x**, is the simple average of the elements in **x**, calculated as

$$n^{-1}\sum_{i=1}^{n} x[i]$$

GAUSS command:

meanc(z) will calculate the sample means of each of the k columns of data in the n by k matrix **z** and display them, in column order, as a k by 1 column vector.

Median Values (Sample)

The sample median of a n by 1 vector of data, **x**, is a value representing a central point of the sorted (from lowest to highest) values in **x**. In particular, it is a value such that greater than or equal to 50% of the values in **x** are both greater than or equal to and less than or equal to this value.

GAUSS command:

median(z) will find the median value in each of the k columns of data in the n by k matrix **z** and display them, in column order, as a k by 1 column vector.

Mode (Sample)

The sample mode of a set of sample outcomes contained in the n by 1 vector **x** is the value having the highest frequency of occurrence. Multiple modes occur when there is more than one value that occurs most frequently.

GAUSS command:

The mode of the values in the n by 1 vector x can be found by utilizing a procedure for finding the most frequently occurring values in the vector, such as the procedure given below.

```
proc  mode(x);
local z, freq, maxfreq, modes;
        z = unique(x,1);
        freq = counts(x,z);
        maxfreq=maxc(freq)
        modes=selif(z~freq,freq.=maxfreq);
        modes=modes[.,1];
retp(modes);
endp;
```

Orthogonal Matrices

A square matrix **x** is orthogonal iff any of the following equivalent conditions hold: $\mathbf{x}' = \mathbf{x}^{-1}$, $\mathbf{x}'\mathbf{x} = \mathbf{I}$, $\mathbf{x}\mathbf{x}' = \mathbf{I}$.

<u>Properties</u>:

1. An orthogonal matrix has full rank .

2. If **x** is orthogonal, then det(**x**) = +1 or -1.

3. If **x** and **y** are orthogonal, **xy** is also orthogonal (assuming multiplication conformability).

4. If **x** is orthogonal, $\mathbf{x}^{-1}$ and **x**' are orthogonal.

5. A real-valued eigenvalue of an orthogonal matrix is equal to 1 or -1.

GAUSS command:

Ortho(x) will test to see whether **x** is orthogonal and will print the results of the test if the following procedure is included in a GAUSS program:

```
proc ortho(x);
if feq(x'x,eye(rows(x)));
        print "orthogonal";
else;
        print "not orthogonal";
endif;
endp;
```

Note: The fuzzy equality check **feq(a,b)** is used to mitigate the roundoff errors that can occur in the calculation of **x'x**, and in the subsequent comparison to **I**.

Partitioned Inverse and Partitioned Determinant

There are special formulae that can be applied to represent, in block-wise form, the inverse of a partitioned matrix. There are also special formulae that can be used to represent the determinant of a partitioned matrix in terms of the blocks of the matrix.

Let **a** be a matrix partitioned into the following four blocks:

$$\mathbf{a} = \left[\begin{array}{c|c} \mathbf{a}_{11} & \mathbf{a}_{12} \\ \hline \mathbf{a}_{21} & \mathbf{a}_{22} \end{array}\right]$$

Then assuming that the appropriate inverse matrices exist for the submatrices, the following formulae apply in representing the inverse of the matrix **a** block-wise:

$$\mathbf{a}^{-1} = \left[\begin{array}{c|c} \mathbf{a}_{11} & \mathbf{a}_{12} \\ \hline \mathbf{a}_{21} & \mathbf{a}_{22} \end{array}\right]^{-1} = \left[\begin{array}{c|c} \mathbf{a}_{11}^{-1} + \mathbf{a}_{11}^{-1}\mathbf{a}_{12}\mathbf{q}^{-1}\mathbf{a}_{21}\mathbf{a}_{11}^{-1} & -\mathbf{a}_{11}^{-1}\mathbf{a}_{12}\mathbf{q}^{-1} \\ \hline -\mathbf{q}^{-1}\mathbf{a}_{21}\mathbf{a}_{11}^{-1} & \mathbf{q}^{-1} \end{array}\right]$$

or

$$\mathbf{a}^{-1} = \left[\begin{array}{c|c} \mathbf{a}_{11} & \mathbf{a}_{12} \\ \hline \mathbf{a}_{21} & \mathbf{a}_{22} \end{array}\right]^{-1} = \left[\begin{array}{c|c} \mathbf{v}^{-1} & -\mathbf{v}^{-1}\mathbf{a}_{12}\mathbf{a}_{22}^{-1} \\ \hline -\mathbf{a}_{22}^{-1}\mathbf{a}_{21}\mathbf{v}^{-1} & \mathbf{a}_{22}^{-1} + \mathbf{a}_{22}^{-1}\mathbf{a}_{21}\mathbf{v}^{-1}\mathbf{a}_{12}\mathbf{a}_{22}^{-1} \end{array}\right]$$

where the matrices **q** and **v** are defined as

$$\mathbf{q} = \mathbf{a}_{22} - \mathbf{a}_{21}\mathbf{a}_{11}^{-1}\mathbf{a}_{12} \quad \text{and} \quad \mathbf{v} = \mathbf{a}_{11} - \mathbf{a}_{12}\mathbf{a}_{22}^{-1}\mathbf{a}_{21} \ .$$

The determinant of a matrix partitioned as the matrix **a** above can be represented in terms of the blocks of the matrix in one or both of the following ways. Applying the representations depend on the existence of the inverse matrices in the formulae.

$$\det(\mathbf{a}) = \det(\mathbf{a}_{11})\det\left(\mathbf{a}_{22} - \mathbf{a}_{21}\mathbf{a}_{11}^{-1}\mathbf{a}_{12}\right)$$

or

$$\det(\mathbf{a}) = \det(\mathbf{a}_{22})\det\left(\mathbf{a}_{11} - \mathbf{a}_{12}\mathbf{a}_{22}^{-1}\mathbf{a}_{21}\right) .$$

The inverse and the determinant of a *block diagonal* matrix have particularly simple forms. The inverse matrix is a block diagonal matrix having a diagonal consisting of the respective inverses of the diagonal blocks of the original matrix. For example, a 3 by 3 block diagonal matrix

$$\mathbf{a} = \left[\begin{array}{c|c|c} \mathbf{a}_{11} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{a}_{22} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{a}_{33} \end{array}\right]$$

would have an inverse given by

$$\mathbf{a}^{-1} = \left[\begin{array}{c|c|c} \mathbf{a}_{11}^{-1} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{a}_{22}^{-1} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{a}_{33}^{-1} \end{array}\right].$$

The determinant of a block diagonal matrix is simply equal to the products of the determinants of the blocks. For example, in the preceding case of the the 3 by 3 block diagonal matrix, the determinant would be given by

$$\det(\mathbf{a}) = \det(\mathbf{a}_{11})\det(\mathbf{a}_{22})\det(\mathbf{a}_{33}) .$$

GAUSS command:

There are no special commands in GAUSS for operating on partitioned matrices since fundamentally, all of the standard matrix operations can be applied directly to a matrix, whether or not it has been defined in a paritioned form. However, GAUSS can be used to verify some of the properties of partitioned matrices. Please use the Code and RUN buttons to view this application.

Partitioned Matrices

A partitioned matrix, **x**, is a matrix that is defined in terms of a collection of submatrices. The submatrices can be thought of as being defined by drawing vertical and horizontal lines between various rows and columns of the matrix. The submatrices defined in this way are often also referred to as *blocks* of the matrix. For example, a 3 by 3 matrix consisting of nine submatrices, or blocks, can be represented as follows:

$$\mathbf{a} = \left[\begin{array}{c|c|c} \mathbf{a}_{11} & \mathbf{a}_{12} & \mathbf{a}_{13} \\ \hline \mathbf{a}_{21} & \mathbf{a}_{22} & \mathbf{a}_{23} \\ \hline \mathbf{a}_{31} & \mathbf{a}_{32} & \mathbf{a}_{33} \end{array}\right]$$

In this case, each of the submatrices or blocks, $\mathbf{a}_{ij}$, can be a matrix of values.

A *block diagonal* matrix is one in which only blocks along the diagonal of the paritioned matrix have nonzero elements. For example, a 3 by 3 block diagonal matrix would be represented as follows:

$$\mathbf{a} = \left[\begin{array}{c|c|c} \mathbf{a}_{11} & \mathbf{0} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{a}_{22} & \mathbf{0} \\ \hline \mathbf{0} & \mathbf{0} & \mathbf{a}_{33} \end{array}\right]$$

Standard matrix operations on partitioned matrices, such as addition, subtraction, and multiplication, are performed analogously to how they are performed for ordinary matrices (see the matrix manual entries relating to Addition, Subtraction, and Multiplication of matrices).. Moreover, the operations can be applied "block-wise", effectively acting notationally as if the blocks were actually scalar matrix elements. For example, in a 2 by 2 partitioned matrix case, addition, subtraction, and multiplication of two matrices would proceed as follows. Let the matrix **a** be defined as in the first matrix above except for it now being a 2 by 2 matrix, and define the matrix **b** analogously, simply replacing **a** with **b** in the matrix definition. Also assume that the blocks *are defined conformably*, so that all of the matrix operations ahead are defined. Then:

$$\mathbf{a} + \mathbf{b} = \left[\begin{array}{c|c} \mathbf{a}_{11} + \mathbf{b}_{11} & \mathbf{a}_{12} + \mathbf{b}_{12} \\ \hline \mathbf{a}_{21} + \mathbf{b}_{21} & \mathbf{a}_{22} + \mathbf{b}_{22} \end{array}\right]$$

$$\mathbf{a} - \mathbf{b} = \left[\begin{array}{c|c} \mathbf{a}_{11} - \mathbf{b}_{11} & \mathbf{a}_{12} - \mathbf{b}_{12} \\ \hline \mathbf{a}_{21} - \mathbf{b}_{21} & \mathbf{a}_{22} - \mathbf{b}_{22} \end{array}\right]$$

$$\mathbf{ab} = \left[\begin{array}{c|c} \mathbf{a}_{11}\mathbf{b}_{11} + \mathbf{a}_{12}\mathbf{b}_{21} & \mathbf{a}_{11}\mathbf{b}_{12} + \mathbf{a}_{12}\mathbf{b}_{22} \\ \hline \mathbf{a}_{21}\mathbf{b}_{11} + \mathbf{a}_{22}\mathbf{b}_{21} & \mathbf{a}_{21}\mathbf{b}_{12} + \mathbf{a}_{22}\mathbf{b}_{22} \end{array}\right]$$

The transpose of a partitioned matrix can be defined in a special block-wise fashion whereby block subscripts are reversed and transposes are then applied to the blocks. For example, in the 2 by 2 case, the transpose of the partioned matrix can be represented as follows:

$$\mathbf{a}' = \left[\begin{array}{c|c} \mathbf{a}'_{11} & \mathbf{a}'_{21} \\ \hline \mathbf{a}'_{12} & \mathbf{a}'_{22} \end{array}\right]$$

Note carefully the subscripts on the matrix blocks.

The inverse of a partitioned matrix, as well as the determinant of a partitioned matrix, can be obtained in exactly the same way as they are for any matrix (see the matrix manual entries dealing with matrix inverses and determinants). There are also special block-wise ways of representing inverses and determinants of partitioned matrices. See the Partitioned Inverse and Partitioned Determinant entries in the matrix review manual for a discussion of these representations.

GAUSS command:

There are no special commands in GAUSS for operating on partitioned matrices since fundamentally, all of the standard matrix operations can be applied directly to a matrix, whether or not it has been defined in a paritioned form. However, GAUSS can be used to verify some of the properties of partitioned matrices. Please use the Code and RUN buttons to view this application.

Projections and Projection Matrix

The *projection* of a n by 1 vector **y** onto the linear subspace spanned by the k columns of the n by k matrix **x** is defined by

$$\hat{\mathbf{y}} = \mathbf{x}(\mathbf{x}'\mathbf{x})^{-1}\mathbf{x}'\mathbf{y} = \mathbf{p}_{\mathbf{x}}\mathbf{y}, \text{ where } \mathbf{p}_{\mathbf{x}} \equiv \mathbf{x}(\mathbf{x}'\mathbf{x})^{-1}\mathbf{x}'.$$

The matrix $\mathbf{p}_{\mathbf{x}}$ is called a *projection matrix*.

In effect, the projection of **y** onto **x** is the best approximation to **y** possible via linear combinations of the columns of **x**, which is to say, the best approximation within the linear subspace spanned by the columns of **x**. Best in this sense means closest to **y** in terms of Euclidean distance between the approximation vector and the value of **y**. Note in particular that

$$\hat{\mathbf{y}} = \mathbf{p}_{\mathbf{x}}\mathbf{y} = \arg\min_{\mathbf{z}}\left[\left((\mathbf{z}-\mathbf{y})'(\mathbf{z}-\mathbf{y})\right)^{1/2} \text{ s.t. } \mathbf{z} = \mathbf{x}\mathbf{b}\right]$$

where **b** is a vector representing coefficients defining a linear combination of the columns of **x**. Note the very close analogy between projections and least squares fits of **y** on **x**.

Note that the n by n projection matrix $\mathbf{p}_{\mathbf{x}}$ is symmetric, idempotent, and has rank k. Also, $\mathbf{p}_{\mathbf{x}}\,\mathbf{x} = \mathbf{x}$.

GAUSS command:

The projection of **y** onto the linear subspace spanned by the columns of **x** can be calculated using the GAUSS syntax **x*invpd(x'x)*x'y** . Another way of defining the projection is through use of the syntax **x*(y/x)** where if **y** is a vector and **x** is a conformable matrix, **(y/x)** is an alternative method of calculating **invpd(x'x)*x'y**. The projection matrix can be calculated as **x*invpd(x'x)*x'**.

Quadratic Forms

A *quadratic form* in the n by 1 vector **x** is defined as the matrix multiplication **x'qx**. The n by n matrix **q** is known as the *matrix of the quadratic form*. In scalar notation, the quadratic form can be represented as:

$$\begin{aligned}\mathbf{x'qx} &= \sum_{i=1}^{n}\sum_{j=1}^{n} q_{ij}x_i x_j \\ &= \sum_{i=1}^{n} q_{ii}x_i^2 + \sum_{i=1}^{n}\sum_{j\neq i} q_{ij}x_i x_j \\ &= \sum_{i=1}^{n} q_{ii}x_i^2 + 2\sum_{i=1}^{n}\sum_{j<i} q_{ij}x_i x_j \quad \text{(if } \mathbf{q} \text{ is symmetric)}\end{aligned}$$

A quadratic form can be classified as being positive definite, positive semidefinite, negative definite, negative semidefinite, or indefinite depending on whether the matrix **q** of the quadratic form is positive definite, positive semidefinite, negative definite, negative semidefinite, or indefinite, respectively. In these cases, the value of the quadratic form is positive, nonegative, negative, nonpositive, or sign indeterminate, respectively, for arbitrary nonzero choices of the vector **x**. See the topic "positive definite" for more information on the definiteness characterisitic of matrices.

GAUSS command:

The value of a quadratic form can be calculated in GAUSS by using the syntax **x'q*x**. The definiteness of the matrix can be determined by an examination of the eigenvalues of the matrix of the quadratic form, **q**. See the "positive definite" topic for more information.

Random Matrices

A matrix of random numbers, or random matrix, is a matrix whose elements are the outcomes of random variables. The elements of the n by k matrix **x** can represent the outcomes of nk iid scalar random variables. Alternatively, the rows of the n by k matrix **x** can represent n iid outcomes of some k-variate random variable. More generally, the random variable outcomes displayed in **x** need not be iid. There exists a wide variety of procedures for generating random matrix outcomes on the computer, and GAUSS contains a number of commands that will generate these outcomes for the distribution listed in the subtopic list above. We list below a number of the possibilities for generating random variable outcomes that are built in to GAUSS. See the GAUSS online help for other possibilities.

***GAUSS command*:**

Beta: **rndbeta(r,c,a,b)** generates an r by c matrix of iid Beta(a,b) random variables outcomes.

Gamma: **rndgam(r,c,alpha)** generates an r by c matrix of iid Gamma(alpha,1) random variable outcomes. To generate an r by c matrix of iid Gamma(alpha, beta) random variable outcomes, use **beta*rndgam(r,c,alpha)**.

Normal: **rndn(r,c)** generates a r by c matrix of iid N(0,1) random variable outcomes. To generate an r by c matrix of iid $N(\mu,\sigma^2)$ random variable outcomes when a = μ and $b^2 = \sigma^2$, use **b*rndn(r,c) + a**.

Multivariate Normal: **rndmultn(r,c,u,sigma)** will generate an r by c matrix of r iid outcomes from the c-dimensional multivariate normal distribution N(u,sigma) having c by 1 mean vector u and c by c covariance matrix sigma if the following procedure is included in a GAUSS program:

```
proc rndmultn(r,c,u,sigma);
retp((chol(sigma)'rndn(c,r)+u)');
endp;
```

Poisson: **rndp(r,c,lambda)** generates an r by c matrix of iid Poisson(lambda) random variable outcomes.

Uniform: **rndu(r,c)** generates an r by c matrix of iid Uniform(0,1) random variable outcomes. To generate iid outcomes from the Uniform(a,b) distribution, use **a+(b-a)*rndu(r,c)**.

Note: See the command **rndseed** in the online GAUSS help file for information about changing the seed value of the random number generators in **rndu** and **rndn**.

Rank of a Matrix

The rank of a n by k matrix **x**, denoted by rank(**x**), is equal to the maximum number of linearly independent columns or (equivalently) rows of the matrix. If rank(**x**) = k, then **x** is said to have *full column rank*, and if rank(**x**) = n, then **x** is said to have *full row rank*. If **x** is square and nonsingular, so that det(**x**) is unequal to zero, then **x** is simply said to have *full rank*.

Properties:

1. $\operatorname{rank}(\mathbf{xy}) \leq \min(\operatorname{rank}(\mathbf{x}), \operatorname{rank}(\mathbf{y}))$.

2. If **x** and **y** are nonsingular square matrices, then for any matrix **z** for which the matrix multiplications are defined,

   rank(**z**) = rank(**xz**) = rank(**zy**) = rank(**xzy**).

3. If **x** is a square matrix, then **x** has full rank iff det(**x**) is unequal to zero.

4. For any matrix **x**, rank(**x**) = rank(**x'**) = rank(**x'x**) = rank(**xx'**).

GAUSS command:

rank(x)

Note: When **x** is square and symmetric, rank(**x**) is determined by the number of eigenvalues of **x** that are $\geq 10^{-13}$ in absolute value. For nonsymmetric matrices, rank(**x**) equals the number of singular values in the singular value decomposition of **x** that are $\geq 10^{-13}$ in absolute value. The test value of 10^{-13} can be changed by assigning the global variable _svdtol to the desired test value. In theory, the rank of **x** equals the number of nonzero singular values of **x**.

Reshaping a Matrix

Let **x** be any n by k matrix. The reshape of matrix **x**, denoted by reshape(**x**,r,c) for integer-valued r and c, is the r by c matrix that displays rc elements of **x** in row major order. That is, referring to the elements of **x** from left-to-right, in progression from the first to the nth row, reshape(**x**,r,c) has the first c elements of **x** in its first row, the next c elements of **x** in its 2nd row, and so on. If rc < nk, the remaining nk - rc elements of **x** are deleted. If rc > nk, then the first rc - nk elements of **x** are repeated in the last elements of the reshape matrix.

GAUSS command:

reshape (x,r,c) displays rc of the elements of **x** in row major order.

Singular Value Decomposition and Spectral Decomposition

For any n by k matrix **x**, the matrix can be represented in terms of its so-called *singular value decomposition* (SVD) as **x** = **usv'**. In theory, the singular value decomposition representation of the matrix **x** *always* exists. In this representation, the matrix **u** is n by n and contains (column-wise) the *left singular vectors* of **x** , and the matrix **v** is k by k and contains (column-wise) the *right singular vectors* of **x**. The n by k matrix **s** is a diagonal matrix whose diagonal elements are called *singular values*. The singular values are displayed in descending order along the diagonal of **s**.

In the special case where **x** is symmetric, **u** = **v**, and the decomposition is then also referred to as the *spectral decomposition* of **x**.

Properties of the SVD:

1. The matrices **u** and **v** are orthogonal, so that **u'u** = **I and v'v** = **I.**

2. **u'xv** = **s**

3. The number of nonzero singular values is equal to the rank of **x**.

4. The n columns of **u** are the eigenvectors of **xx**'.

5. The k columns of **v** are the eigenvectors of **x'x**.

6. The nonzero singular values of **x** are the positive square roots of the eigenvalues of **x'x**.

7. If **x** is symmetric and positive semidefinite, then **u** = **v** and the diagonal elements of **s** equal the eigenvalues of **x**..

8. $\mathbf{x} = \sum_{i=1}^{r} s_i \mathbf{u}[.,i]\,\mathbf{v}[.,i]'$, where $r = \text{rank}(\mathbf{x})$.

GAUSS command:

{u,s,v}=svd1(x) calculates the singular value decomposition of the n by k matrix, **x**, and returns the n by n matrix left singular vectors in **u**, the right k by k matrix of singular vectors in **v**, and the diagonal n by k matrix of singular values in **s** (arrange in descending order along the diagonal). Then **x** can be represented, in terms of GAUSS syntax, as **x** = **u*****s*****v**', and **x** can be diagonalized by **u'x*****v** = **s**.

Solving Systems of Linear Equations

The method of solving a system of linear equations **cx = r** for the value of **x** depends on whether or not **c** is invertible.

Case Where c is Invertible

If **c** is invertible, then the solution for **x** is found by premultiplying both sides of the system of linear equations by the inverse of the matrix **c**. In this case, the solution for **x** can be represented as $\mathbf{x} = \mathbf{c}^{-1}\mathbf{r}$.

Case Where c is Not Invertible

If **c** is not invertible, then a solution to the system of linear equations **cx = r** may or may not exist. There are a number of existence conditions that can be used to determine whether or not a solution does exist. Three such conditions are listed below.

A solution to the system of linear equations **cx = r** exists:

Existence 1: iff rank(**c**) = rank(**c**~**r**)

Existence 2: if **c** is m by n and rank(**c**) = m.

Existence 3: iff $\mathbf{c}\mathbf{c}^{-}\mathbf{r} = \mathbf{r}$

In words, a solution to the system of linear equations exists iff the rank of the matrix **c** equals the rank of the augmented (horizontally concatenated) matrix **c**~**r**, or if the matrix **c** has full row rank, or iff premultiplication of the vector **r** by the product of **c** times its generalized inverse leaves the vector **r** unchanged.

Assuming that at least one solution exists (i.e., an existence condition holds), then all of the solutions of the system of linear equations can be represented by the following:

$$\mathbf{x} = \mathbf{c}^{-}\mathbf{r} + \left(\mathbf{I} - \mathbf{c}^{-}\mathbf{c}\right)\mathbf{h}$$, where the value of **h** is arbitrary.

That is, every **x** that can be defined by choosing **h** to be any (conformable) vector of values is a solution to the system of linear equations **cx = r**.

GAUSS command:

The matrix inverse procedure **inv( )**, the generalized inverse procedure **pinv( )**, and the rank procedure **rank( )** can be used to implement all of the formulae and conditions discussed above. See the GAUSS Code, and also RUN the code to see how the GAUSS application for this topic is written and operates.

Sorting Matrices

The concept of *sorting a matrix* refers to the operation of reordering the positions of the rows (columns) of a matrix according to the magnitude of the elements in one of the columns (rows) of a matrix.

GAUSS command:

sortc(x,v) will produce a matrix in which the rows of **x** have been sorted (reordered) according to the magnitude of the elements in the column number indicated by the integer value of v, the sort being in *ascending* order. If a sort in descending order is desired, the rows can be reversed after the sort using the GAUSS command **rev( )** as **rev(sortc(x,v))**, which reverses the order of the rows in a matrix.

Submatrix Extraction

Beginning with a n by k matrix, **x**, a submatrix of **x** is any matrix formed by taking all of the elements in the intersection of designated rows and columns of **x** and displaying them as a new matrix.

GAUSS command:

submat(x,r,c) defines the submatrix of **x** formed by intersecting the row indices defined in the column vector **r** with the column indices defined in the column vector **c**. If r = 0, all rows are used, and if c = 0, all columns are used.

x[r,c] defines the same submatrix as above when **r** and **c** are column vectors of indices.

Standard Deviation (Sample)

The (unbiased) sample variance estimate corresponding to a n by 1 vector of data, **x**, is calculated as

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x[i]-\bar{x}\right)^2.$$

The sample standard deviation derived from the (unbiased) sample variance is the square root of the (unbiased) sample variance. Note that the sample variance based on the empirical distribution function of the sample data uses a divisor of n rather n-1 in the preceding formula. A sample standard deviation can then also be calculated by taking the square root of the latter sample variance estimate. As the sample size increases, the difference in divisors becomes immaterial.

GAUSS command:

stdc(z) will calculate the sample standard deviation (based on the unbiased sample variance estimate) of each of the k columns of data in the n by k matrix **z** and display them, in column order, as a k by 1 column vector. In order to calculate the sample variances of each of the k columns of data in **z** and display them as a k by 1 column vector, one can use **(stdc(z)^2)**. Alternatively, one can use the diagonal of the covariance matrix **vcx(z)** as **diag(vcx(z))**.

Symmetric Matrix

The matrix, **x**, is a symmetric matrix iff **x** = **x**'. In other words, a matrix **x** is symmetric iff x[i,j] = x[j,i] for every i and j.

GAUSS command:

To check whether a matrix is symmetric, the following GAUSS code could be used in a program:

```
if x == x';
        print "symmetric";
else;
        print "nonsymmetric";
endif;
```

Symmetric Matrix Square Root

The symmetric matrix square root (SMSR) of a positive definite symmetric (PDS) matrix **x**, denoted by $\mathbf{x}^{1/2}$, is a symmetric matrix for which, $\mathbf{x} = \mathbf{x}^{1/2}\,\mathbf{x}^{1/2}$. In other words, the matrix square root $\mathbf{x}^{1/2}$ operates analogously in terms of *matrix* multiplication to how the scalar square root, $z^{1/2}$ operates in terms of *scalar* multiplication, i.e., $z = z^{1/2}z^{1/2}$. The matrix $\mathbf{x}^{1/2}$ will be a PDS matrix.

The SMSR of a PDS matrix **x** can be defined in terms of the eigenvalues and eigenvectors of **x**. In particular, let **p** be the matrix whose column vectors represent the eigenvectors of **x**, and let Λ be the diagonal matrix whose diagonal elements are the eigenvalues of **x** corresponding to the eigenvectors in **p**, respectively. Then the SMSR is defined by $\mathbf{x}^{1/2}= \mathbf{p}\Lambda^{1/2}\mathbf{p}'$ where $\Lambda^{1/2}$ is a diagonal matrix having the square roots of the diagonal elements of Λ on its diagonal. Since $\mathbf{p'p} = \mathbf{pp'} = \mathbf{I}$, and $\Lambda^{1/2}\Lambda^{1/2} = \Lambda$ then $\mathbf{x} = \mathbf{p}\Lambda\mathbf{p}' = \mathbf{x}^{1/2}\,\mathbf{x}^{1/2}$, which follows directly from the diagonalization of symmetric matrices, whereby $\mathbf{p'xp} = \Lambda$.

GAUSS command:

SMSR(x) will calculate the symmetric matrix square root of the PDS matrix **x** if the following GAUSS procedure is contained in a GAUSS program:

```
proc SMSR(x);
Local va, ve;
{va,ve} = eighv(x);
retp(ve.*SQRT(va')*(ve'));
endp;
```

Toeplitz Matrix

A Toeplitz matrix is a square matrix whose elements assume values in a special band-type pattern. A *band* of a matrix is either the diagonal or a strip of numbers that is parallel to the diagonal. More specifically, a band of the square matrix **x** is defined by all of the elements x[i,j] for which i-j is a fixed integer value. Thus the diagonal band is given by i-j = 0. The first upper band is given by i-j = -1, the first lower band is given by i-j = 1, and so on. A Toeplitz matrix is such that all of the elements along any band of the matrix have identical values. If the Toeplitz matrix is symmetric, then pairs of progressive bands above and below the diagonal have identical element values. More formally, **x** is a Toeplitz matrix if x[i,j] = a[i-j] for every i, j. The Toeplitz matrix is symmetric if a[i-j]=a[j-i].

GAUSS command:

Toeplitz(a) creates a symmetric Toeplitz matrix from the elements in the column vector **a**. The first entry in **a** equals the value of the diagonal band, the second entry in **a** equals the value of the first bands above and below the diagonal, and so on.

Trace of a Matrix

The trace of a n by n matrix, tr(**x**), is the sum of the diagonal elements of **x**,

i.e., $\mathrm{tr}(\mathbf{x}) = \sum_{i=1}^{n} x[i,i]$.

Properties: (Assuming matrix conformability.)

1. tr(**x** + **y**) = tr(**x**) + tr(**y**)

2. tr(**xy**) = tr(**yx**)

3. tr(**xyz**) = tr(**zxy**) = tr(**yzx**)

4. tr(c**x**) = c tr(**x**), where c is a scalar.

5. If **x** is idempotent, then tr(**x**) = rank(**x**).

GAUSS command:

sumc(diag(x)) will calculate the sum of the diagonal elements of **x**, i.e., the trace of **x**.

Transpose of a Matrix

The transpose of a matrix **x**, denoted by **x'**, is the matrix that results when the rows and columns of **x** are interchanged. That is, the elements in column j of **x** become the elements in row j of **x'** so that x[i,j] = x[j,i] for every i, j.

GAUSS command:

x' , where **x** can be replaced by any other variable name representing a matrix.

Triangular Form of a Symmetric Matrix

If **x** is a symmetric matrix, then the unique elements of **x** are represented by the elements on the diagonal and above (upper triangle) or equivalently on the diagonal and below (lower triangle). The upper triangular matrix operator, upmat(**x**), stores the n(n+1)/2 elements of the upper triangle in their appropriate place, and fills in the rest of the elements of the matrix with zeros. The lower triangular matrix operator, lowmat(**x**), stores the n(n+1)/2 elements of the lower triangle in their appropriate place, and fills in the rest of the elements of the matrix with zeros. Either triangular matrix thus contains all of the unique elements in **x** in either an upper or lower triangle of the matrix.

In order to economize on the storage of a symmetric matrix, it is possible to store only the n(n+1)/2 unique elements of the matrix as a column vector. The matrix operator to do this is the triangular storage operator vech(**x**), which stores the lower triangle of the symmetric matrix in row order form, from the first to the last row. The original n by n symmetric matrix **x** can be recovered from the vech(**x**) storage of the matrix by using the triangle expansion operator xpnd(**x**).

GAUSS command:

lowmat(x), upmat(x), vech(x), xpnd(x) as explained above.

Triangular Matrix

A triangular matrix is a square matrix that has all of its nonzero values either on and above (an upper triangular matrix) or on and below (a lower triangular matrix) the diagonal of the matrix, thereby forming a "triangle" of nonzero values within the matrix. More formally,

upper triangular: x[i,j] = 0 for every i-j > 0
lower triangular: x[i,j] = 0 for every i-j < 0

Properties:

1. The product of two upper (lower) triangular matrices is also an upper (lower) triangular matrix.

2. The determinant of a triangular matrix is equal to the product of its diagonal elements.

3. The eigenvalues of a triangular matrix are equal to the diagonal elements of the matrix.

4. The inverse of an upper (lower) triangular matrix is also an upper (lower) triangular matrix.

Unit Matrix

An n by k unit matrix is a matrix that has each of its nk elements equal to the same value, 1.

GAUSS command:

ones(n,k) creates an n by k unit matrix.

Variance (Sample)

The (unbiased) sample variance estimate corresponding to a n by 1 vector of data, **x**, is calculated as

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}\left(x[i]-\bar{x}\right)^2.$$

The sample standard deviation derived from the (unbiased) sample variance is the square root of the (unbiased) sample variance. Note that the sample variance based on the empirical distribution function of the sample data uses a divisor of n rather n-1 in the preceding formula. A sample standard deviation can then also be calculated by taking the square root of the latter sample variance estimate. As the sample size increases, the difference in divisors becomes immaterial.

GAUSS command:

stdc(z) will calculate the sample standard deviation (based on the unbiased sample variance estimate) of each of the k columns of data in the n by k matrix **z** and display them, in column order, as a k by 1 column vector. In order to calculate the sample variances of each of the k columns of data in **z** and display them as a k by 1 column vector, one can use **(stdc(z)^2)**. Alternatively, one can use the diagonal of the covariance matrix **vcx(z)** as **diag(vcx(z))**.

Vectorization of a Matrix

Any n by k *matrix* **x** can be represented in the form of a nk by 1 column *vector* via the process of vectorization. Vectorization can occur either column-wise or row-wise. Vec(**x**) denotes column-wise vectorization whereby the successive columns of **x** are stacked one below the other, beginning with the first column and ending with the kth·. Vecr(**x**) denotes row-wise vectorization in which the successive rows of **x** are appended one to the right of the other, beginning with the first row and ending with the nth, and then the extended row vector is transposed, and thus represented, in column vector form.

GAUSS command:

vec(x) is a column vector representing the column-wise vectorization of **x**.

vecr(x) is a column vector representing the row-wise vectorization of **x**.

Zero Matrix

A n by k zero matrix is a matrix that has each of its nk elements equal to the same value, 0.

GAUSS command:

zeros(n,k) creates an n-dimensional zero matrix.