

Linear Algebra

Linear algebra is a branch of mathematics that is concerned with the theory and application of systems of linear equations. The mathematics of matrices and vectors are fundamental to the theory of linear algebra. Those who are unfamiliar with matrix notation may at first find it a little confusing. However, matrices provide a convenient method of dealing with systems of many variables. Often in working with linear equations, only the coefficients of the variables are important, not the variables themselves. Matrices are just a mathematical “shorthand” that maintains the respective positions of each of the coefficients. Computer libraries of matrix functions allow the analyst to quickly and efficiently solve large sets of linear equations that would be much too tedious by hand.

6.1 MATRICES

A matrix is a rectangular array of numbers. Each element of a matrix is a scalar quantity (a real, or a complex number). For example, the matrix M has two rows and three columns, and a total of $2 \times 3 = 6$ elements:

$$M = \begin{bmatrix} 1 & 2 & 4 \\ 3 & 7 & 9 \end{bmatrix}$$

6.1.1 Matrix Addition and Subtraction

Matrices can be added and subtracted only if they have the same number of rows and columns (i. e., the same dimensions). To add or subtract two matrices, simply add or subtract the corresponding elements and enter the answers into the same places of the resulting matrix. The following is an example of the addition of 2×2 matrices:

$$\begin{bmatrix} 1 & 2 \\ 3 & 7 \end{bmatrix} + \begin{bmatrix} 7 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 1+7 & 2+2 \\ 3+3 & 7+4 \end{bmatrix} = \begin{bmatrix} 8 & 4 \\ 6 & 11 \end{bmatrix}$$

Subtracting the above matrices yields:

$$\begin{bmatrix} 1 & 2 \\ 3 & 7 \end{bmatrix} - \begin{bmatrix} 7 & 2 \\ 3 & 4 \end{bmatrix} = \begin{bmatrix} 1-7 & 2-2 \\ 3-3 & 7-4 \end{bmatrix} = \begin{bmatrix} -6 & 0 \\ 0 & 3 \end{bmatrix}$$

See Functions: `mxadd()`, `Cmxadd()`, `mxsub()`, `Cmxsub()`, `vadd()`, `Cvadd()`, `vsub()`, `Cvsub()`.

6.1.2 Scaling a Matrix

Sometimes it is necessary to multiply a matrix by a scalar quantity (i. e., a real or a complex number). The following is an example of scalar multiplication:

$$3 \times \begin{bmatrix} 1 & 2 \\ 3 & 7 \end{bmatrix} = \begin{bmatrix} 3 \times 1 & 3 \times 2 \\ 3 \times 3 & 3 \times 7 \end{bmatrix} = \begin{bmatrix} 3 & 6 \\ 9 & 21 \end{bmatrix}$$

See Functions: `mxscale()`, `CmxscalR()`, `CmxscalC()`, `vscale()`, `CvscalR()`, `CvscalC()`.

6.1.3 Matrix Multiplication

Two matrices can be multiplied only if the number of columns in the first matrix is equal to the number of rows in the second matrix. The element in the i th row and j th column of the product matrix is formed by taking the dot product (see dot product in the Vectors section 6.3.1) between the i th row of the first matrix and the j th column of the second matrix. The following is an example of multiplication of a 2×3 matrix by a 3×2 matrix:

$$\begin{bmatrix} 1 & 2 & 9 \\ 3 & 7 & 5 \end{bmatrix} \begin{bmatrix} 1 & 7 \\ 2 & 3 \\ 4 & 1 \end{bmatrix} = \begin{bmatrix} 1 \times 1 + 2 \times 2 + 9 \times 4 & 1 \times 7 + 2 \times 3 + 9 \times 1 \\ 3 \times 1 + 7 \times 2 + 5 \times 4 & 3 \times 7 + 7 \times 3 + 5 \times 1 \end{bmatrix} \\ = \begin{bmatrix} 41 & 22 \\ 37 & 47 \end{bmatrix}$$

Unlike scalar arithmetic, the order of matrix multiplication is important since, in general, $A * B$ is not equal to $B * A$. Thus, matrix multiplication is not commutative.

6.1.4 Matrix Transpose

The **transpose** of a matrix is obtained by interchanging the row and column indices of the elements of the matrix. For example, if

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ a_{31} & a_{32} & a_{33} & \dots & a_{3n} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & a_{m3} & \dots & a_{mn} \end{bmatrix}$$

then

$$A^T = \begin{bmatrix} a_{11} & a_{21} & a_{31} & \dots & a_{m1} \\ a_{12} & a_{22} & a_{32} & \dots & a_{m2} \\ a_{13} & a_{23} & a_{33} & \dots & a_{m3} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{1n} & a_{2n} & a_{3n} & \dots & a_{mn} \end{bmatrix}$$

Note that since A is an $m \times n$ matrix, A^T must be an $n \times m$ matrix. Some useful properties of matrix transposition are

1. $(ABC \dots)^T = \dots C^T B^T A^T$
2. $(A + B)^T = A^T + B^T$
3. $(cA)^T = cA^T$, where c is a scalar
4. $(A^T)^T = A$

See Functions: `mxtersp()`, `Cmxtersp()`.

6.1.5 Some Special Square Matrices

Square matrices are the most important class of matrices, and their algebra has some special properties. The redundancy in special square matrices can be exploited to reduce the number of computations required to solve sets of simultaneous linear equations.

For square matrices it is possible to define an operation that is somewhat analogous to scalar division called matrix inversion (see matrix inversion in section 6.1.9).

For scalar arithmetic, multiplication by 1 (unity) results in the original scalar value. A similar operation also exists for matrices. The **identity matrix** I is a square matrix with all of the diagonal elements equal to one and all other elements equal to zero, for example

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Multiplication of a matrix with the identity matrix produces the original matrix.

A matrix U is said to be **upper diagonal** if all of its elements below the diagonal are zero. Upper diagonal matrices have the following form:

$$U = \begin{bmatrix} u_{11} & u_{12} & u_{13} & u_{1n} \\ 0 & u_{22} & u_{23} & u_{2n} \\ 0 & 0 & u_{33} & u_{3n} \\ 0 & 0 & 0 & u_{nn} \end{bmatrix}$$

There are, in general, only $n(n+1)/2$ nonzero elements in an upper diagonal matrix.

A matrix L is **lower diagonal** if all of its elements above the diagonal are zero. Lower diagonal matrices have the following form:

$$L = \begin{bmatrix} l_{11} & 0 & 0 & 0 \\ l_{21} & l_{22} & 0 & 0 \\ l_{31} & l_{32} & l_{33} & 0 \\ l_{n1} & l_{n2} & l_{n3} & l_{nn} \end{bmatrix}$$

Like the upper diagonal matrix, there are at most only $n(n+1)/2$ nonzero elements in a lower diagonal matrix.

A matrix T is **tridiagonal** if it is of the form

$$T = \begin{bmatrix} t_{11} & t_{12} & 0 & 0 \\ t_{21} & t_{22} & t_{23} & 0 \\ 0 & t_{32} & t_{33} & t_{mn} \\ 0 & 0 & t_{nm} & t_{nn} \end{bmatrix} \quad \text{where } m = n - 1$$

There are, in general, only $3n - 2$ nonzero elements in a tridiagonal matrix.

The elements of a **symmetric** matrix have the following symmetry around the main diagonal:

$$a_{ij} = a_{ji} \quad \text{where } 0 \leq i, j \leq n - 1$$

The transpose of a symmetric matrix is the original matrix. Householder developed an efficient method solving linear systems of real-symmetric matrices.

A **Hermitian matrix** has complex elements that exhibit complex conjugate symmetry around the main diagonal. The following is an example of a 3×3 Hermitian matrix:

$$M = \begin{bmatrix} a + i0 & b - ic & c - id \\ b + ic & e + i0 & f - ig \\ c + id & f + ig & h + i0 \end{bmatrix} \quad \text{where } i^2 = -1$$

Note that the elements along the main diagonal of the Hermitian matrix must be real. Thus, a real symmetric matrix is also Hermitian, since the imaginary parts of its elements are all zero.

A **Toeplitz matrix** M is a real symmetric matrix with the additional row and column symmetry shown below:

$$M = \begin{bmatrix} m_1 & m_2 & m_3 & \dots & m_n \\ m_2 & m_1 & m_2 & \dots & m_{n-1} \\ m_3 & m_2 & m_1 & \dots & m_{n-2} \\ \vdots & \vdots & \vdots & & \vdots \\ m_n & m_{n-1} & m_{n-2} & \dots & m_1 \end{bmatrix}$$

Note that the entire Toeplitz matrix is defined by any one of its row or column vectors. Levinson developed an efficient method for solving linear systems of Toeplitz matrices.

Sets of equations that can be represented by upper diagonal, lower diagonal, tridiagonal, or Toeplitz matrices are much easier to solve than those represented by more general matrices.

See Functions: `mxident()`, `levinson()`.

6.1.6 Matrix Trace

The **trace** is defined for a square matrix as the sum of all its diagonal elements.

Two useful properties of the trace of a matrix are

1. $\text{Tr}(A + B) = \text{Tr}(A) + \text{Tr}(B)$
2. $\text{Tr}(ABC) = \text{Tr}(BCA) = \text{Tr}(CAB)$

For example, the trace of the 3×3 identity matrix is **3**. Note that, in general, the trace of the product of two matrices is not equal to the product of the traces of each of the matrices.

See Functions: `mxtrace()`, `Cmxtrace()`.

6.1.7 Matrix Determinant

The **determinant** of a matrix, $|A|$, is a scalar value that indicates whether or not a system of linear equations has a unique solution. If the determinant of the system matrix is nonzero, then the set of equations has a unique solution. The determinant is of fundamental importance to the concepts of **matrix rank**, **linear independence**, and **matrix singularity**.

There is a common formula that can be followed to find the determinant of any square matrix. For two-dimensional matrices, the determinant is given by

$$|A| = a_{11}a_{22} - a_{12}a_{21}$$

For three-dimensional matrices, the determinant is given by

$$|A| = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{13}a_{22}a_{31} - a_{11}a_{21}a_{33}$$

where a_{ij} is the element of the i th row and j th column of matrix A .

Determinants of real matrices have an interesting geometrical interpretation. If A is a two-dimensional square matrix, the absolute value of its determinant is equal to the area of the parallelogram formed by its row (or column) vectors. Similarly, if A is a three-dimensional matrix, the absolute value of its determinant is equal to the volume of the parallelepiped formed by its row (or column) vectors.

If two rows (or columns) of a matrix are interchanged, then the determinant changes its sign.

6.1.8 Matrix Singularity and Rank

A square matrix is **singular** if its determinant is zero. Conversely, if the determinant of a matrix is nonzero, then it is nonsingular. This is important because algebraic equations that have a nonsingular system matrix have a unique solution.

The **rank** of a square matrix is defined as the size of the largest nonzero determinant that can be formed from the matrix. If A is an $n \times n$ matrix, then the maximum rank of A is n . Furthermore, if A is of maximal rank, then A is said to be nonsingular. Equivalently, if A is maximal rank, then it has a nonzero determinant and the rows (and columns) of A are a **linear independent basis set**.

6.1.9 Matrix Inverse

A square matrix A has an **inverse**, A^{-1} , if and only if

$$A^{-1}A = AA^{-1} = I$$

where I is the identity matrix.

There is just one condition required for the existence of an inverse matrix: the original matrix must be nonsingular. That is, the matrix must be of maximal rank, or equivalently, its determinant must be nonzero. Matrix inversion is similar to scalar division and trying to invert a singular matrix is analogous to a scalar divide by zero.

See Functions: `mxinv()`, `mxinv22()`, `mxinv33()`, `Cmxinv()`, `Cmxinv22()`, `Cmxinv33()`.

6.2 SIMULTANEOUS LINEAR EQUATIONS

Any system of simultaneous equations can be expressed in matrix form. For example,

$$\begin{aligned} 3x + 2y &= 2 \\ 5x + 5y &= 1 \end{aligned}$$

in matrix form is

$$\begin{bmatrix} 3 & 2 \\ 5 & 5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Note that the matrix representation eliminates the need for repeating the “*plus*” signs, the “*equals*” signs, and the x and y characters.

There are a variety of applied techniques available for solving this set of linear equations. It is beyond the scope of this section to describe them all. It is more important to know when to use each technique. The reader may verify that the correct answer is $x = 1.6$, $y = -1.2$.

Cramer’s rule is useful for hand calculations with low order systems such as this. This formula is easily generalized for linear systems of two equations. Consider the following two equations:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

The solution to this system of equations can be calculated as the ratio of two determinants. Solving for x yields

$$x = \frac{\begin{vmatrix} b_1 & a_{12} \\ b_2 & a_{22} \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}} = \frac{b_1 a_{22} - a_{12} b_2}{a_{11} a_{22} - a_{21} a_{12}}$$

and similarly, the solution for y is

$$y = \frac{\begin{vmatrix} a_{11} & b_1 \\ a_{21} & b_2 \end{vmatrix}}{\begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix}} = \frac{a_{11} b_2 - b_1 a_{21}}{a_{11} a_{22} - a_{21} a_{12}}$$

Note that the determinant of the system matrix is in the denominator in both of the above expressions, and must be nonzero for there to be a solution (i. e., the system matrix must be nonsingular).

When the number of equations is large, Cramer's rule becomes cumbersome. For these cases, a variety of computer solutions are more appropriate. The "brute-force" approach is to employ matrix inversion. Although matrix inversion is a convenient method for solving systems of linear equations, it is not the most efficient, nor the most accurate. Direct matrix reduction techniques such as LU decomposition and Gaussian elimination often provide superior speed and accuracy. This is especially true for large matrices that are nearly singular. However, for small matrices, matrix inversion is usually adequate.

See Functions: `lineqn()`, `Clineqn()`.

6.2.1 Pseudo-Inverse

Pseudo-inverse approaches are very useful in matching simple geometric shapes to real-world objects. These techniques have also proved to be very useful in calibration and coordinate alignment for robotic vision. These approaches are very powerful and are optimal in the sense that they minimize mean-square error (see Chapter 5, Statistics).

Suppose a data set is known to belong to a linear process. We may wish to estimate the parameters that characterize this process but we find that the data is corrupted with random noise (e. g., measurement errors). For example, the points on a wall all lie in a plane. With no noise, only three points are needed to characterize the plane's equation. However, in measuring the distance to the wall, random measurement noise is added. This noise degrades the estimate of the plane's true equation. One way to reduce these random effects is to take several additional data points (more than three) and overconstrain the system. Pseudo-inverse methods can be used to solve this augmented set of equations.

Consider the following set of overconstrained linear equations:

$$D_{MN} v_N = z_M$$

where M is the number of equations and N is the number of unknowns ($M \geq N$).

Solving for v_N would be straightforward if D_{MN} were square (i. e., $M = N$) and invertible. Nonetheless, the above equation can be easily transformed into an equivalent square system by premultiplying both sides of the equation by the transpose matrix, D_{NM}^T :

$$[D_{NM}^T D_{MN}] v_N = D_{NM}^T z_M$$

Observe that this transforms the overconstrained system of equations into a set that can be easily solved.

Let

$$A_{NN} = [D_{NM}^T D_{MN}] \quad \text{and} \quad w_N = D_{NM}^T z_M$$

then v_N can be determined from the equivalent system of equations:

$$\begin{aligned} A_{NN} v_N &= w_N \\ v_N &= A_{NN}^{-1} w_N \end{aligned}$$

If A_{NN} is nonsingular, then the **pseudo-inverse matrix** A_{NN}^{-1} exists and the solution v_N is minimum mean-square. The pseudo-inverse matrix is a real-symmetric matrix.

See Function: pseudinv().

6.3 VECTORS

A vector is simply a matrix with one of its dimensions equal to one. A row vector is a $1 \times n$ matrix, and a column vector is an $n \times 1$ matrix. All of the algebra of matrices also applies to vectors. For vectors with real elements, the **dot product**, **cross product**, and **magnitude** have useful geometrical applications.

6.3.1 Dot Product

The **dot product** (or inner product) has an interesting geometrical interpretation. If θ is the angle between the two vectors v_1 and v_2 , then:

$$v_1 * v_2 = |v_1| |v_2| \cos(\theta)$$

When the dot product is normalized by the magnitude of the larger vector, it represents the projection of the smaller vector onto the larger. If the dot product between two vectors is zero, then the vectors are perpendicular (i. e., the angle) between them is 90 degrees).

See Function: vdot().

6.3.2 Cross Product

The **cross product** (sometimes called the vector product) between two three-dimensional vectors is given by the following determinant:

$$v_1 \times v_2 = \begin{vmatrix} i & j & k \\ x_1 & y_1 & z_1 \\ x_2 & y_2 & z_2 \end{vmatrix}$$

where i , j and k are the unit vectors in the x , y , and z directions. The variables x_1 , y_1 , and z_1 are the components of v_1 . The variables x_2 , y_2 , and z_2 are the components of v_2 .

The cross product has an interesting geometrical interpretation. If θ is the angle between the two vectors, then:

$$v_1 \times v_2 = u|v_1||v_2|\sin(\theta)$$

where u is the unit vector perpendicular to the plane of v_1 and v_2 , and so directed that a right-handed screw driven in the direction of u would carry v_1 into v_2 .

The cross product is not commutative, since

$$v_1 \times v_2 = -v_2 \times v_1$$

If the cross product between two vectors is zero, then the vectors are colinear (i. e., the angle between them is 0 degrees).

See Function: `vcross()`.

6.3.3 Vector Magnitude

The **magnitude** of a real vector, v , is the square root of the sum of the squares of its elements:

$$|v| = [v_1^2 + v_2^2 + v_3^2 + \dots + v_n^2]^{0.5}$$

The magnitude of a vector is the same as its length.

See Functions: `vmag()`, `Cvmag()`.

6.4 EIGENANALYSIS

Eigensystems play a crucial part in the theory of electrical and mechanical resonance and in the theory of statics as well. Eigenvalues and eigenvectors can be used to determine the stability of feedback systems and to control the convergence of associated tracking algorithms. The concepts of similarity transforms and matrix diagonalization are of fundamental importance to eigenanalysis because the eigenvalues are invariant to such transforms. The general eigenvalue problem is one of determining the similarity of a matrix to a diagonal form. This section deals with eigenvalues of **real symmetric** and **complex Hermitian** matrices.

6.4.1 Real symmetric matrices

A nonsingular $n \times n$ real symmetric matrix has n real eigenvalues and n distinct eigenvectors. The most common eigenanalysis technique for real symmetric matrices is Jacobi's method. The **cyclic Jacobi** method consists of a sequence of similarity transforms designed to diagonalize a real symmetric matrix **A**. This technique derives an orthogonal transformation matrix **P** whose columns are the desired n eigenvectors. The n eigenvalues are the diagonal elements of the diagonalized matrix

$$C = P^{-1}AP$$

This diagonalization is sometimes referred to as a **similarity transformation**.

The Jacobi method obtains the orthogonal matrix P as an infinite product of rotation matrices P_l where the operative elements are of the form:

$$P_l = \begin{bmatrix} \cos \omega & -\sin \omega \\ \sin \omega & \cos \omega \end{bmatrix}$$

and all of the other elements are identical to the identity matrix. The cosine terms are constrained to intersect the main diagonal; the sine terms can be anywhere off the main diagonal. Let the above four entries represent the elements in positions (i, i) , (i, k) , (k, i) , and (k, k) . Then the corresponding elements of $P_l^{-1}AP_l$ are computed as:

$$\begin{aligned} b_{ij} &= a_{ii} \cos^2 \omega + 2a_{ik} \sin \omega \cos \omega + a_{kk} \sin^2 \omega \\ b_{ki} &= b_{ik} = (a_{kk} - a_{ii}) \sin \omega \cos \omega + a_{ik} (\cos^2 \omega - \sin^2 \omega) \\ b_{kk} &= a_{ii} \sin^2 \omega - 2a_{ik} \sin \omega \cos \omega + a_{kk} \cos^2 \omega \end{aligned}$$

The angle ω is chosen such that

$$\tan(2\omega) = 2a_{ik}/(a_{ii} - a_{kk})$$

which then makes $b_{ki} = b_{ik} = 0$. At each step, the Jacobi algorithm annihilates a pair of off-diagonal elements. Unfortunately, successive steps introduce nonzero contributions to positions that were formerly zero. Nevertheless, the iterative product $P_0P_1P_2 \dots$ produces the desired orthogonal matrix P .

Similarly, the desired diagonal matrix C is obtained from

$$C = \dots P_2^{-1}P_1^{-1}P_0^{-1}AP_0P_1P_2 \dots$$

The cyclic Jacobi method is practically foolproof for all real symmetric matrices.

See Function: `mxeigen()`.

6.4.2 Complex Hermitian matrices

A nonsingular $n \times n$ complex Hermitian matrix has n real eigenvalues and n distinct eigenvectors. A Hermitian matrix exhibits complex conjugate symmetry around its main diagonal, which also means that the elements along the main diagonal must be real. Thus, a real symmetric matrix is also Hermitian, since the imaginary parts of its elements are all zero.

The eigenanalysis of any $n \times n$ Hermitian matrix can be transformed into an equivalent problem that involves a $2n \times 2n$ real symmetric matrix. The real symmetric eigensystem can then be solved with Jacobi transformations (see W. H. Press, et al., pp. 381–382).

The method is as follows:

1. The complex matrix m is separated into its real and imaginary parts:

$$m[n][n] = A[n][n] + iB[n][n]$$

2. Likewise, the complex eigenvector, v , is separated into its real and imaginary components:

$$v[n] = a[n] + ib[n]$$

3. The following complex eigenvalue equation transforms into one real equation of twice the size according to:

$$(A + iB)\lambda(x + iy) = \lambda(a + ib)$$

$$\begin{bmatrix} A & -B \\ B & A \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \lambda \begin{bmatrix} a \\ b \end{bmatrix}$$

4. The real symmetric eigensystem is solved with Jacobi transformations. The eigenvalues of the real system are ordered as redundant pairs:

$$\begin{aligned} \text{eigenvalues} &== \lambda_1, \lambda_1, \lambda_2, \lambda_2, \dots, \lambda_n, \lambda_n \\ \text{eigenvectors} &== \begin{bmatrix} a1 \\ b1 \end{bmatrix}, \begin{bmatrix} -b1 \\ a1 \end{bmatrix}, \begin{bmatrix} a2 \\ b2 \end{bmatrix}, \begin{bmatrix} -b2 \\ a2 \end{bmatrix}, \dots, \begin{bmatrix} an \\ bn \end{bmatrix}, \begin{bmatrix} -bn \\ an \end{bmatrix} \end{aligned}$$

5. The complex eigensystem solution is then found by taking every other eigenvalue and eigenvector from the real system:

$$\begin{aligned} \text{eigenvalues} &== \lambda_1, \lambda_2, \dots, \lambda_n \\ \text{eigenvectors} &== \begin{bmatrix} a1 \\ b1 \end{bmatrix}, \begin{bmatrix} a2 \\ b2 \end{bmatrix}, \dots, \begin{bmatrix} an \\ bn \end{bmatrix} \end{aligned}$$

See Function: Cmxeigen().