2.2 Creating a Two-Dimensional Array (Matrix)

A two-dimensional array, also called a matrix, has numbers in rows and columns. Matrices can be used to store information like the arrangement in a table. Matrices play an important role in linear algebra and are used in science and engineering to describe many physical quantities.

In a square matrix the number of rows and the number of columns is equal. For example, the matrix

$$\begin{array}{ccc} 7 & 4 & 9 \\ 3 & 8 & 1 \\ 6 & 5 & 3 \end{array} \qquad 3\times 3 \text{ matrix}$$

is square, with three rows and three columns. In general, the number of rows and columns can be different. For example, the matrix:

$$\begin{array}{rrrrrr} 31 & 26 & 14 & 18 & 5 & 30 \\ 3 & 51 & 20 & 11 & 43 & 65 \\ 28 & 6 & 15 & 61 & 34 & 22 \\ 14 & 58 & 6 & 36 & 93 & 7 \end{array} \qquad 4\times 6 \text{ matrix}$$

has four rows and six columns. A $m\times n$ matrix has m rows and n columns, and m by n is called the size of the matrix.

A matrix is created by assigning the elements of the matrix to a variable. This is done by typing the elements, row by row, inside square brackets [ ]. First type the left bracket [then type the first row, separating the elements with spaces or commas. To type the next row type a semicolon or press **Enter**. Type the right bracket] at the end of the last row.

```
variable_name=[1st row elements; 2nd row elements; 3rd
               row elements; ... ; last row elements]
```

The elements that are entered can be numbers or mathematical expressions that may include numbers, predefined variables, and functions. *All the rows must have the same number of elements.* If an element is zero, it has to be entered as such. MATLAB displays an error message if an attempt is made to define an incomplete matrix. Examples of matrices defined in different ways are shown in Tutorial 2-2.

Tutorial 2-2: Creating matrices.

```
>> a=[5  35  43;  4  76  81;  21  32  40]
a =
     5    35    43
     4    76    81
    21    32    40
>> b = [7  2  76  33  8
1  98  6  25  6
5  54  68  9  0]
```

A semicolon is typed before a new line is entered.

The **Enter** key is pressed before a new line is entered.

Tutorial 2-2: Creating matrices. (Continued)

```
b =
     7     2    76    33     8
     1    98     6    25     6
     5    54    68     9     0
>> cd=6; e=3; h=4;
>> Mat=[e, cd*h, cos(pi/3); h^2, sqrt(h*h/cd), 14]
Mat =
    3.0000   24.0000    0.5000
   16.0000    1.6330   14.0000
>>
```

Three variables are defined.

Elements are defined by mathematical expressions.

Rows of a matrix can also be entered as vectors using the notation for creating vectors with constant spacing, or the `linspace` command. For example:

```
>> A=[1:2:11; 0:5:25; linspace(10,60,6); 67 2 43 68 4 13]
A =
     1     3     5     7     9    11
     0     5    10    15    20    25
    10    20    30    40    50    60
    67     2    43    68     4    13
>>
```

In this example the first two rows were entered as vectors using the notation of constant spacing, the third row was entered using the `linspace` command, and in the last row the elements were entered individually.

2.2.1 *The* `zeros`, `ones` *and,* `eye` *Commands*

The `zeros(m,n)`, `ones(m,n)`, and `eye(n)` commands can be used to create matrices that have elements with special values. The `zeros(m,n)` and the `ones(m,n)` commands create a matrix with *m* rows and *n* columns in which all elements are the numbers 0 and 1, respectively. The `eye(n)` command creates a square matrix with *n* rows and *n* columns in which the diagonal elements are equal to 1 and the rest of the elements are 0. This matrix is called the identity matrix. Examples are:

```
>> zr=zeros(3,4)
zr =
     0     0     0     0
     0     0     0     0
     0     0     0     0
>> ne=ones(4,3)
```