**Chapter 3 Arrays**

An array is a list of numbers arranged in rows and/or columns. The simplest array (one dimensional) is a row, or a column of numbers. A more complex array (two‐dimensional) has a set of numbers arranged in rows and columns.

**3.1 Creating a One‐Dimensional Array (Vector)**

**3.1.1 Creating a Vector from a Known List of Numbers**

The vector is created by typing the elements (numbers) inside square brackets **[ ].**

variable\_name= [ type vector elements ]

**Row vector:** To create a row vector type the elements with a space or a comma between the elements inside the square brackets.

 **Column vector:** To create a column vector type the left square bracket **[** and the enter the elements with a semicolon between them, or press the Enter key after each element. Type the right square bracket **]**.

لتكوين متجه معروف العناصر تكتب عناصره بين قوسين مربعين مفصولة بفراغات أو فوارز لمتجه الصف أومفصولة بفارز منقطة أو بالضغط على**Enter** بين عنصر وآخر بالنسبة لمتجه العمود.



**3.1.2 Creating a Vector with Constant Spacing by Specifying the First Term, the spacing, and the Last Term**

In a vector with constant spacing the difference between the elements is the same. A vector in which the first term is m, the spacing is q, and the last term is n is created by typing:

variable\_name= m: q: n

variable\_name= [m: q: n]

 or

المتجه الذي بدايته m و نهايته n و الفرق بين كل عنصريين متتاليين هو q يكتب باستخدام احد العبارتين اعلاه. 



**لان عدد اعمدة المتجه اكبر من سعة الشاشة فيتم تقسيم المتجه على الشاشة و ليس في الذاكرة**

**Remark:**

• If the spacing q is not mentioned, it will be considered 1 (example 3.7).

• If the numbers m, q, and n, are such that the value of n cannot be obtained by adding q’s to m, then (for positive n) the last element in the vector will be the last number that does not exceed n (example 3.9).

**3.1.3 Creating a Vector with Constant Linear Spacing by Specifying the First and Last Terms and the Number of Terms.**

A vector in which the first element is a, the last element is b, and the number ofelements is n linearly spaced is created by typing the **linspace** command:

variable\_name= linspace(a, b, n)

لتكوين متجه عنصره الاول a و الاخير b و عدد عناصره n موزعة خطيا نستخدم الايعاز linspace .



**Remark:**

• If the number of elements n is not mentioned, it will be considered 100 elements

(example 3.12).

**3.1.4 Creating a Vector with Constant Logarithmic Spacing by Specifying the First and Last Terms and the Number of Terms.**

A vector in which the first element is a, the last element is b, and the number of elements is n logarithmically spaced is created by typing the **logspace** command:

variable\_name= logspace(a, b, n)

This command generates n points between decades 10^a and 10^b.It is especially useful for creating frequency vectors.

y = logspace(a,pi)

Generates the points between 10^a and pi, which is useful for digital signal processing where frequencies over this interval go around the unit circle.

لتكوين متجه عنصره الاول a و الاخير b وعدد عناصره n موزعة لوغارتميا نستخدم الايعازlogspace.





**Remark:**

* If the number of elements n is not mentioned, it will be considered 50 elements (example 3.15).
* All the arguments to **logspace** must be scalars.

**3.2 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 in a table. Matrices play an important role in linear algebra.

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 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.

تكتب المصفوفة بذآر عناصرها صفاً صفاً وتفصل بين الصفوف فارزة منقوطة أو الضغط على**Enter** بين صف و اخر.

 

**3.3 The zeros , ones and eye Commands**

The **zeros(m,n)**, the **ones(m,n)**, and **eye(n)** commands can be used to create matrices that have elements with special values. The **zeros(m,n)** command creates a matrix with m rows and n columns in which all elements are the numbers 0. The **ones(m,n)** command creates a matrix with m rows and n columns in which all elements are the numbers 1. The **eye(n)** command creates an identity matrix with n rows and n columns.



**3.4 The Transpose Operator**

The transpose operator, when applied to a vector, switches a row (column) vector to a column (row) vector. When applied to a matrix, it switches the rows (columns) to columns (rows). The transpose operator is applied by typing a single quote ( ‘ )following the variable to be transposed. When applied to an array containing complex numbers, it switches each complex number to its conjugate.

يمكن إيجاد منقول مصفوفة باستخدام علامة الاقتباس (’ )مع ملاحظة إن منقول مصفوفة حاوية على أعداد عقدية يغير الأعداد إلى مرافقها العقدي.



**3.5 Array Addressing**

Elements in an array (either vector or matrix) can be addressed individually or in subgroups. This is useful when there is need to redefine only some of the elements, to use specific elements in calculations, or when a subgroup of the elements is used to define a new variable.

يمكن أن تعنون العناصر في مصفوف فرادى أو مجموعات لإعادة تعريف بعض العناصر أو لاستخدام عناصر محددة في حسابات أو لاستخدام عناصر محددة في تعريف متغيرات جديدة.

**3.5.1 Vector**

The address of an element in a vector is its position in the row (or column). We can use colons ( : ) to address a range of elements in a vector. A single vector element, v(k), can be used just as a variable. For example, it is possible to change the value of only one element of a vector by reassigning a new value to a specific address. This is done by typing:

v(k)=value. If v is a vector, addressing elements in v follows the rules:

* v(k) Refers to the element in position k.
* v( : ) Refers to all the elements of the vector v (either a row or a column vector).
* v(m:n) Refers to elements m through n of the vector v.

**Example 3.26:**

Consider the vector v=(13,15,12,11,16,22,17,3344).Using MATLAB, create the vector then:

1) Change the 7th element of v to 35.

 2) Address the 3rd element of v.

3) Execute y=v(6:end) (or equivalently: address the last four columns of v).

4) Add the 1st to the 7th element.

5) Execute v(5)^v(8)+sqrt(v(7)).

6) Execute the command: d=[v([1, 3 4]);v([2,3,5]);v([1,5,7])].

7) Execute the command: x=v([1 3 5;2:4]).

8) Execute the command: C=v([1,3:5,9]).



**3.5.2 Matrix**

The address of an element in a matrix is its position, defined by the row number and the column number where it is located. As with vectors, it is possible to change the value of just one element of a matrix by assigning a new value to that element. Also, single elements can be used like variables in mathematical expressions and functions. We can use colons ( : ) to address a range of elements in a matrix. If A is a matrix, addressing elements in A follows the rules:

* **A(k,p)** Refers to the element in row k and column p of the matrix A.
* **A( : )** Refers to the elements of the matrix A (write them vertically column by column).
* **A(:,n)** Refers to the elements in all the rows of column n of the matrix A.
* **A(m,:)** Refers to the elements in all the columns of row m of the matrix A.
* **A( : , m : n)** Refers to the elements in all the rows between columns m and n of the matrix A.
* **A(m : n , :)** Refers to the elements in all the columns between rows m and n of the matrix A.
* **A(m : n, p : q)** Refers to the elements in rows m through n and columns p through q of the matrix A.

**Example 3.27:**

Consider the matrix A= $\left(\begin{matrix}11&22&\begin{matrix}33&44&55\end{matrix}\\1&1&\begin{matrix}1&1&1\end{matrix}\\0&17&\begin{matrix}13&77&98\end{matrix}\end{matrix}\right). $Using MATLAB creates the matrix A, then:

1) Address the element in the 3rd row, 4th column of A.

2) Create a column vector that contains the elements of the 4th column of A.

3) Create a row vector that contains the elements of the 2nd row of A.

4) Create a matrix that contains 2nd column through 4th column of A.

5) Create a matrix that contains 1st row through 2nd row of A.

6) Create a matrix that contains 2nd column through 4th column and 1st row through 2nd row of A.

7) Create a row vector that contains the elements of the 1st and 5th column of A.

8) Create a row vector that contains the elements of the 3rd row and 4th column of A.

9) Change the element A(2,3) to 5.

10) Find A(1,2)^A(2,3)+A(3,5).

11) E=A([1,2],[1,2,4:5]).



**Remark:**

New vectors and matrices can be created from existing ones by using a range of

elements, or a range of rows and columns (using : )(see examples 3.26(3), 3.27(2, 3, 4, 5, and 6)). It is possible, however, to select only specific elements or specific rows and columns of existing variables to create new variables. This is done by typing the selected elements or rows or columns inside brackets (see examples 3.26(6, 7, and 8), 3.27(7, 8 and 11)).

**3.6 Adding Elements to Existing Variables**

A variable that exists as a vector, or a matrix, can be changed by adding elements to it (remember that a scalar is a vector with one element).

**3.6.1 Adding Elements to a Vector**

A vector can be changed to have more elements, or it can be changed to be a twodimensional matrix. Elements can be added to an existing vector by assigning values to the new elements. If a vector has n elements and a new value is assigned to an element with address of n+2 or larger, MATLAB assigns zeros to the elements that are between the last original element and the new element.



Elements can also be added to a vector by appending existing vectors.





**3.6.2 Adding Elements to a Matrix**

Rows and/or columns can be added to an existing matrix to obtain a matrix of different size. This can be done by assigning new values, or by appending existing variables. This must be done carefully since the size of the added rows or columns must fit the existing matrix.



**3.7 Deleting Elements**

An element or a range of elements, of an existing variable can be deleted by reassigning nothing to these elements. This is done by using square brackets with nothing typed between them. By deleting elements a vector can be made shorter and a matrix can be made to have a smaller size.



**3.8 Built‐In Functions for Handling Arrays**

MATLAB has many built‐in functions for managing and handling arrays. Some of these are listed below (see the laboratory guide):





**Example 3.41:**

Create a 6 x 6 matrix AB in which the middle two rows, and the middle two columns are 1’s, and the rest are 0’s.

**Example 3.42**

Given are a 5 x 6 matrix A, a 3 x 6 matrix B, and a 9 element long vector v.

 A=$ \left(\begin{matrix}\begin{matrix}2&5&8\\3&6&9\end{matrix}&\begin{matrix}11&14&17\\12&15&18\end{matrix}\\\begin{matrix}\begin{matrix}4&7&10\end{matrix}\\\begin{matrix}5&8&11\end{matrix}\\\begin{matrix}6&9&12\end{matrix}\end{matrix}&\begin{matrix}\begin{matrix}13&16&19\end{matrix}\\\begin{matrix}14&17&20\end{matrix}\\\begin{matrix}15&18&21\end{matrix}\end{matrix}\end{matrix}\right)$, B = $\left(\begin{matrix}\begin{matrix}5&10\\30&35\end{matrix}&\begin{matrix}\begin{matrix}15&20&\begin{matrix}25&30\end{matrix}\end{matrix}\\\begin{matrix}40&45&\begin{matrix}50&55\end{matrix}\end{matrix}\end{matrix}\\\begin{matrix}55&60\end{matrix}&\begin{matrix}65&70&\begin{matrix}75&80\end{matrix}\end{matrix}\end{matrix}\right)$,

V= **(**$\begin{matrix}99&98&\begin{matrix}97&96&\begin{matrix}95&94&\begin{matrix}93&92&91\end{matrix}\end{matrix}\end{matrix}\end{matrix}$**)**

Create the three arrays in the Command Window, and then, by one command, replace the last four columns of the 1st and 3rd rows of A with the first four columns of the first two rows of B, the last four columns of the 4th row of A with the elements 5 through 8 of v, and the last four columns of the 5th row of A with column 3 through 5 of the third row of B.

**Solution:**

>> A=[2:3:17 ; 3:3:18 ; 4:3:19 ; 5:3:20 ; 6:3:21]

A =

 2 5 8 11 14 17

 3 6 9 12 15 18

 4 7 10 13 16 19

 5 8 11 14 17 20

 6 9 12 15 18 21

>> B=[5:5:30 ; 30:5:55 ; 55:5:80]

B =

 5 10 15 20 25 30

 30 35 40 45 50 55

 55 60 65 70 75 80

>> v=[99:‐1:91]

v =

 99 98 97 96 95 94 93 92 91

>> A([1 3 4 5] , 3:6)=[B([1 2] , 1:4) ; v(5:8) ; B(3,2:5)]

A =

 2 5 5 10 15 20

 3 6 9 12 15 18

 4 7 30 35 40 45

 5 8 95 94 93 92

 6 9 60 65 70 75

**3.9 Strings and Strings as Variables**

A string is an array of characters. It is created by typing the characters within single quotes. Strings can include letters, digits, other symbols, and spaces. Strings have several different uses in MATLAB. They are used in output commands to display text messages, in formatting commands of plots, and as input arguments of some functions. Strings can also be assigned to variables by simply typing the string on the right side of the assignment operator.

Example 3.43:

>> B='My name is ABC'

B =

My name is ABC

When a variable is defined as a string, the characters of the string are stored in an array just as numbers are. Each character, including a space, is an element in the array. This means that a one‐line string is a row vector in which the number of elements is equal to the number of characters. The elements of the vectors are addressed by their position.



As with a vector that contains numbers, it is also possible to change specific elements by addressing them directly.

**Example 3.45:**

>> B='My name is ABC';

>> B(12:14)='cdf'

B =

Strings can also be placed in a matrix. As with numbers, this is done by typing semicolon ( ; ) ( or pressing the **Enter** key) at the end of each row. As with a numerical matrix, the number of elements in all rows must be the same. This requirement can cause problems when the intention is to create rows with specific wording. Rows can be made to have the same number of elements by adding spaces.

MATLAB has a built‐in function named **char** that creates an array with rows that have the same number of characters from an input of rows that are not of the same length. MATLAB makes the length of all the rows equal to the longest row by adding spaces at the end of the short lines. In the char function, the rows are entered as strings separated by a comma according to the following format:



**Example 3.46:**



A variable can be defined as a number or a string that is made up of the same digits.

**Example 3.47:**

****

**Problems (2)**

Q 3.1: Create a row vector that has the elements:32, 4, 81, e2.5, 63, cos π/4.

Q 3.2: Create a column vector in which the first element is 15, the elements decrease with increment of ‐5, and the last element is ‐25. (A column vector can be created by the transpose of a row vector).

Q 3.3: Create a row vector with 15 equally linearly spaced elements in which the first element is 7 and the last element is 40.

Q 3.4: Execute the following using MATLAB: 1) a=0:2:6 2) b=[a a] 3) c=[a , a] 4) d=[a ; a] 5) f=[a’ a’] 6) e=[a’ , a’] 7) g=[a’ ; a’].