



# MATLAB

*for Engineering Applications*  
**Fifth Edition**

William J. Palm III

© McGraw Hill LLC. All rights reserved. No reproduction or distribution without the prior written consent of McGraw Hill LLC.

## Chapter 02

### Numeric, Cell, and Structure Arrays

© McGraw Hill LLC

2

## Vectors <sub>1</sub>

To create a *row* vector, separate the elements by commas. Use square brackets. For example,

```
>>p = [3, 7, 9]
p =
     3     7     9
```

You can create a *column* vector by using the *transpose* notation (').

```
>>p = [3, 7, 9] '
p =
     3
     7
     9
```

## Vectors <sub>2</sub>

You can also create a column vector by separating the elements by semicolons. For example,

```
>>g = [3; 7; 9]
g =
     3
     7
     9
```

## Vectors <sub>3</sub>

You can create vectors by "appending" one vector to another.

For example, to create the row vector  $u$  whose first three columns contain the values of  $r = [2, 4, 20]$  and whose fourth, fifth, and sixth columns contain the values of  $w = [9, -6, 3]$ , you type  $u = [r, w]$ . The result is the vector  $u = [2, 4, 20, 9, -6, 3]$ .

## Vectors <sub>4</sub>

The colon operator ( $:$ ) easily generates a large vector of regularly spaced elements. Parentheses are not needed but can be used for clarity. Do not use square brackets.

Typing

```
>>x = m:q:n
```

creates a vector  $x$  of values with a spacing  $q$ . The first value is  $m$ . The last value is  $n$  if  $m - n$  is an integer multiple of  $q$ . If not, the last value is less than  $n$ .

## Vectors <sub>5</sub>

For example, typing `x = 0:2:8` creates the vector `x = [0, 2, 4, 6, 8]`, whereas typing `x = 0:2:7` creates the vector `x = [0, 2, 4, 6]`.

To create a row vector `z` consisting of the values from 5 to 8 in steps of 0.1, type `z = 5:0.1:8`.

If the increment `q` is omitted, it is presumed to be 1. Thus typing `y = -3:2` produces the vector `y = [-3, -2, -1, 0, 1, 2]`.

## Vectors <sub>6</sub>

The `linspace` command also creates a linearly spaced row vector, but instead you specify the number of values rather than the increment.

The syntax is `linspace(x1, x2, n)`, where `x1` and `x2` are the lower and upper limits and `n` is the number of points.

For example, `linspace(5, 8, 31)` is equivalent to `5:0.1:8`.

If `n` is omitted, the spacing is 1.

## Vectors 7

The `logspace` command creates an array of *logarithmically* spaced elements.

Its syntax is `logspace(a,b,n)`, where  $n$  is the number of points between  $10^a$  and  $10^b$ .

For example, `x = logspace(-1,1,4)` produces the vector `x = [0.1000, 0.4642, 2.1544, 10.000]`.

If  $n$  is omitted, the number of points defaults to 50.

## Magnitude, Length, and Absolute Value of a Vector 1

Keep in mind the precise meaning of these terms when using MATLAB.

The `length` command gives the *number of elements* in the vector.

The *magnitude* of a vector  $\mathbf{x}$  having elements  $x_1, x_2, \dots, x_n$  is a scalar, given by  $\sqrt{(x_1^2 + x_2^2 + \dots + x_n^2)}$ , and is the same as the vector's geometric length.

The *absolute value* of a vector  $\mathbf{x}$  is a vector whose elements are the absolute values of the elements of  $\mathbf{x}$ .

## Magnitude, Length, and Absolute Value of a Vector <sub>2</sub>

For example, if  $\mathbf{x} = [2, -4, 5]$ ,

- its length is 3; (computed from `length(x)`).
- its magnitude is  $\sqrt{2^2 + (-4)^2 + 5^2} = 6.7082$ ; (computed from `sqrt(x*x')`).
- Magnitude can also be calculated using the norm function. `norm(x)`
- its absolute value is  $[2, 4, 5]$  (computed from `abs(x)`).

## Matrices

A matrix has multiple rows and columns. For example, the matrix

$$M = \begin{bmatrix} 2 & 4 & 10 \\ 16 & 3 & 7 \\ 8 & 4 & 9 \\ 3 & 12 & 15 \end{bmatrix}$$

has four rows and three columns.

Vectors are special cases of matrices having one row or one column.

## Creating Matrices

If the matrix is small you can type it row by row, separating the *elements* in a given row with *spaces* or *commas* and separating the *rows* with semicolons. For example, typing

```
>>A = [2, 4, 10; 16, 3, 7];
```

creates the following matrix:

$$A = \begin{bmatrix} 2 & 4 & 10 \\ 16 & 3 & 7 \end{bmatrix}$$

Remember, spaces or commas separate elements in different *columns*, whereas semicolons separate elements in different *rows*.

## Creating Matrices from Vectors

Suppose  $a = [1, 3, 5]$  and  $b = [7, 9, 11]$  (row vectors). Note the difference between the results given by  $[a \ b]$  and  $[a; b]$  in the following session:

```
>>c = [a b];
```

```
c =
```

```
1 3 5 7 9 11
```

```
>>D = [a;b]
```

```
D =
```

```
1 3 5
```

```
7 9 11
```

## Array Addressing <sub>1</sub>

You need not use symbols to create a new array. For example, you can type

```
>> D = [[1, 3, 5]; [7, 9, 11]];
```

The colon operator selects individual elements, rows, columns, or “subarrays” of arrays. Here are some examples:

- $v(:)$  represents all the row or column elements of the vector  $v$ .
- $v(2:5)$  represents the second through fifth elements; that is  $v(2)$ ,  $v(3)$ ,  $v(4)$ ,  $v(5)$ .

## Array Addressing <sub>2</sub>

$A(:,3)$  denotes all the elements in the third column of the matrix  $A$ .

$A(:,2:5)$  denotes all the elements in the second through fifth columns of  $A$ .

$A(2:3,1:3)$  denotes all the elements in the second and third rows that are also in the first through third columns.

$v = A(:)$  creates a vector  $v$  consisting of all the columns of  $A$  stacked from first to last.

$A(\text{end},:)$  denotes the last row in  $A$ , and  $A(:,\text{end})$  denotes the last column.



## Array Addressing <sub>3</sub>

You can use array indices to extract a smaller array from another array. For example, if you first create the array **B**

$$\mathbf{B} = \begin{bmatrix} 2 & 4 & 10 & 13 \\ 16 & 3 & 7 & 18 \\ 8 & 4 & 9 & 25 \\ 3 & 12 & 15 & 17 \end{bmatrix}$$

then type  $\mathbf{C} = \mathbf{B}(2:3, 1:3)$ , you can produce the following array:

$$\mathbf{C} = \begin{bmatrix} 16 & 3 & 7 \\ 8 & 4 & 9 \end{bmatrix}$$

## Additional Array Functions <sub>1</sub>

$[u, v, w] = \text{find}(A)$  Computes the arrays  $u$  and  $v$ , containing the row and column indices of the nonzero elements of the matrix  $A$ , and the array  $w$ , containing the values of the nonzero elements. The array  $w$  may be omitted.

$\text{length}(A)$  Computes either the number of elements of  $A$  if  $A$  is a vector or the largest value of  $m$  or  $n$  if  $A$  is an  $m \times n$  matrix.

## Additional Array Functions <sub>2</sub>

`max (A)`      Returns the algebraically largest element in  $A$  if  $A$  is a *vector*.

                 Returns a row vector containing the largest elements in each column if  $A$  is a *matrix*.

                 If any of the elements are *complex*, `max (A)` returns the elements that have the largest magnitudes.

## Additional Array Functions <sub>3</sub>

`[x, k] = max (A)`      Similar to `max (A)` but stores the maximum values in the row vector  $x$  and their indices in the row vector  $k$ .

`min (A)`              Like `max` but returns minimum values.

and  
`[x, k] = min (A)`

Q1-For the matrix A given below, the values returned by max(A) functions is

- a) Max=4+3i 4 (correct)
- b) Max= 1 4
- c) Max=4i+3 4
- d) Max= 1 0

```
>> A=[2,4;1,3;3*i+4,0]
A =
2.0000 + 0.0000i 4.0000 + 0.0000i
1.0000 + 0.0000i 3.0000 + 0.0000i
4.0000 + 3.0000i 0.0000 + 0.0000i

>> max(A)
ans =
4.0000 + 3.0000i 4.0000 + 0.0000i
```

© McGraw Hill LLC

Q2- For the matrix B, use MATLAB to find B'.

$$B = \begin{bmatrix} 2 & 4 & 6 \\ 16 & 3 & 7 \\ 32 & 1 & 9 \end{bmatrix}$$

$$a) \quad B' = \begin{bmatrix} 7 & 6 & 1 \\ 4 & 3 & 32 \\ 2 & 16 & 9 \end{bmatrix}$$

$$b) \quad B' = \begin{bmatrix} 2 & 16 & 32 \\ 4 & 3 & 1 \\ 6 & 7 & 9 \end{bmatrix} \text{ (correct)}$$

$$c) \quad B' = \begin{bmatrix} 16 & 2 & 32 \\ 3 & 4 & 1 \\ 6 & 7 & 9 \end{bmatrix}$$

$$d) \quad B' = \begin{bmatrix} 6 & 7 & 32 \\ 4 & 3 & 1 \\ 2 & 16 & 9 \end{bmatrix}$$

```
>> B=[2,4,6;16,3,7;32,1,9];
>> B'
ans =
2 16 32
4 3 1
6 7 9
```

© McGraw Hill LLC

Q3-For the matrix B, find the array that results from the operation  $A = [B; B']$ . Use MATLAB to determine what number is in row 6, column 3 of the result.

$$B = \begin{bmatrix} 2 & 4 & 6 \\ 16 & 3 & 7 \\ 32 & 1 & 9 \end{bmatrix}$$

- a) 4
- b) 16
- c) 2
- d) 9 (correct)

```
>> B=[2,4,6;16,3,7;32,1,9];
>> A=[B;B']
```

```
A =
     2     4     6
    16     3     7
    32     1     9
     2    16    32
     4     3     1
     6     7     9
```

```
>> X=A(:,3);
>> Y=X(6,:);
>> Y
```

```
Y =
     9
```

```
>> Y=A(6,3)
```

```
Y =
     9
```

© McGraw Hill LLC

## Additional Array Functions <sub>4</sub>

<code>size(A)</code>	Returns a row vector $[m \ n]$ containing the sizes of the $m \times n$ array A.
<code>sort(A)</code>	Sorts each column of the array A in ascending order and returns an array the same size as A.
<code>sum(A)</code>	Sums the elements in each column of the array A and returns a row vector containing the sums.

© McGraw Hill LLC

24

## Additional Array Functions <sup>s</sup>

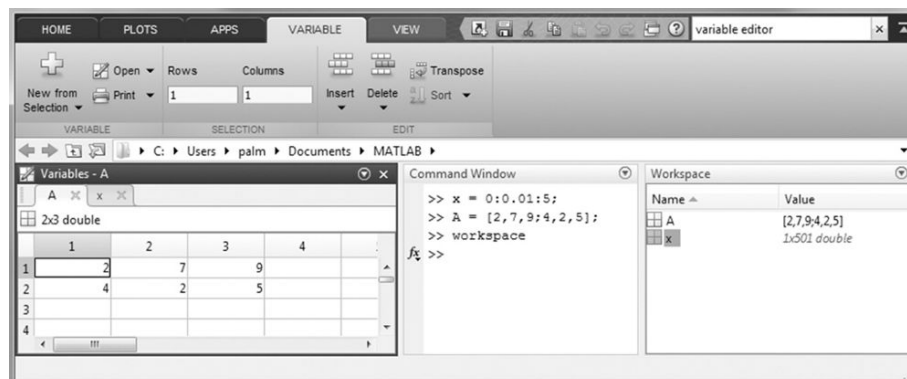
The function `size(A)` returns a row vector  $[m \ n]$  containing the sizes of the  $m \times n$  array **A**. The `length(A)` function computes either the number of elements of **A** if **A** is a vector or the largest value of  $m$  or  $n$  if **A** is an  $m \times n$  matrix.

For example, if

$$\mathbf{A} = \begin{bmatrix} 6 & 2 \\ -10 & -5 \\ 3 & 0 \end{bmatrix}$$

then `max(A)` returns the vector  $[6, 2]$ ; `min(A)` returns the vector  $[-10, -5]$ ; `size(A)` returns  $[3, 2]$ ; and `length(A)` returns 3.

## The Workspace Browser and Variable Editor



## Multidimensional Arrays

They consist of two-dimensional matrices “layered” to produce a third dimension. Each “layer” is called a *page*.

`cat(n,A,B,C, ...)` Creates a new array by concatenating the arrays *A*, *B*, *C*, and so on along the dimension *n*.

## Array Addition and Subtraction

For example:

$$\begin{bmatrix} 6 & -2 \\ 10 & 3 \end{bmatrix} = \begin{bmatrix} 9 & 8 \\ -12 & 14 \end{bmatrix} = \begin{bmatrix} 15 & 6 \\ -2 & 17 \end{bmatrix} \quad (1)$$

Array subtraction is performed in a similar way.

The addition shown in Equation 1 is performed in MATLAB as follows:

```
>>A = [6,-2;10,3];
```

```
>>B = [9,8;-12,14]
```

```
>>A+B
```

```
ans =
```

```
15      6
-2     17
```

## Multiplication 1

Multiplying a matrix **A** by a scalar  $w$  produces a matrix whose elements are the elements of **A** multiplied by  $w$ . For example:

$$3 \begin{bmatrix} 2 & 9 \\ 5 & -7 \end{bmatrix} = \begin{bmatrix} 6 & 27 \\ 15 & -21 \end{bmatrix}$$

This multiplication is performed in MATLAB as follows:

```
>>A = [2, 9; 5,-7];
>>3*A
ans =
     6    27
    15   -21
```

## Multiplication 2

Multiplication of an array by a scalar is easily defined and easily carried out.

However, multiplication of two *arrays* is not so straightforward.

MATLAB uses two definitions of multiplication:

- (1) *array* multiplication (also called *element-by-element* multiplication), and
- (2) *matrix* multiplication.

## Division and exponentiation

Division and exponentiation must also be carefully defined when you are dealing with operations between two arrays.

MATLAB has two forms of arithmetic operations on arrays. Next, we introduce one form, called *array* operations, which are also called *element-by-element* operations. Then we will introduce *matrix* operations. Each form has its own applications.

## Element-by-element operations

Symbol	Operation	Form	Examples
+	Scalar-array addition	$A + B$	$[6, 3] + 2 = [8, 5]$
-	Scalar-array subtraction	$A - B$	$[8, 3] - 5 = [3, -2]$
+	Array addition	$A + B$	$[6, 5] + [4, 8] = [10, 13]$
-	Array subtraction	$A - B$	$[6, 5] - [4, 8] = [2, -3]$
.*	Array multiplication	$A .* B$	$[3, 5] .* [4, 8] = [12, 40]$
./	Array right division	$A ./ B$	$[2, 5] ./ [4, 8] = [2/4, 5/8]$
.\	Array left division	$A .\ B$	$[2, 5] .\ [4, 8] = [2 \setminus 4, 5 \setminus 8]$
.^	Array exponentiation	$A .^ B$	$[3, 5] .^ 2 = [3^2, 5^2]$ $2 .^ [3, 5] = [2^3, 2^5]$ $[3, 5] .^ [2, 4] = [3^2, 5^4]$



## Element-by-element operations <sub>2</sub>

*Array* or *Element-by-element* multiplication is defined only for arrays having the same size. The definition of the product  $\mathbf{x} \cdot \mathbf{y}$ , where  $\mathbf{x}$  and  $\mathbf{y}$  each have  $n$  elements, is

$$\mathbf{x} \cdot \mathbf{y} = [\mathbf{x}(1)\mathbf{y}(1), \mathbf{x}(2)\mathbf{y}(2), \dots, \mathbf{x}(n)\mathbf{y}(n)]$$

if  $\mathbf{x}$  and  $\mathbf{y}$  are row vectors. For example, if

$$\mathbf{x} = [2, 4, -5], \quad \mathbf{y} = [-7, 3, -8]$$

then  $\mathbf{z} = \mathbf{x} \cdot \mathbf{y}$  gives

$$\mathbf{z} = [2(-7), 4(3), -5(-8)] = [-14, 12, 40]$$

## Element-by-element operations <sub>3</sub>

If  $\mathbf{x}$  and  $\mathbf{y}$  are column vectors, the result of  $\mathbf{x} \cdot \mathbf{y}$  is a column vector. For example  $\mathbf{z} = (\mathbf{x}') \cdot (\mathbf{y}')$  gives

$$\mathbf{z} = \begin{bmatrix} 2(-7) \\ 4(3) \\ -5(-8) \end{bmatrix} = \begin{bmatrix} -14 \\ 12 \\ 40 \end{bmatrix}$$

Note that  $\mathbf{x}'$  is a column vector with size  $3 \times 1$  and thus does not have the same size as  $\mathbf{y}$ , whose size is  $1 \times 3$ .

The operations  $\mathbf{x}' \cdot \mathbf{y}$  and  $\mathbf{y} \cdot \mathbf{x}'$  this will generate a  $3 \times 3$  matrix

## Element-by-element operations <sub>4</sub>

The array operations are performed between the elements in corresponding locations in the arrays. For example, the array multiplication operation  $A .* B$  results in a matrix  $C$  that has the same size as  $A$  and  $B$  and has the elements  $c_{ij} = a_{ij} b_{ij}$ . For example, if

$$A = \begin{bmatrix} 11 & 5 \\ -9 & 4 \end{bmatrix} \quad B = \begin{bmatrix} -7 & 8 \\ 6 & 2 \end{bmatrix}$$

then  $C = A .* B$  gives this result:

$$C = \begin{bmatrix} 11(-7) & 5(8) \\ -9(6) & 4(2) \end{bmatrix} = \begin{bmatrix} -77 & 40 \\ -54 & 8 \end{bmatrix}$$

## Element-by-element operations <sub>5</sub>

The built-in MATLAB functions such as `sqrt(x)` and `exp(x)` automatically operate on array arguments to produce an array result the same size as the array argument  $x$ .

Thus, these functions are said to be *vectorized* functions.

For example, in the following session the result  $y$  has the same size as the argument  $x$ .

```
>>x = [4, 16, 25];
```

```
>>y = sqrt(x)
```

```
y =
```

```
2    4    5
```

## Element-by-element operations 6

However, when multiplying or dividing these functions, or when raising them to a power, you must use element-by-element operations if the arguments are arrays.

For example, to compute  $z = (e^y \sin x) \cos^2 x$ , you must type

```
z = exp(y) .* sin(x) .* (cos(x) ) .^2.
```

You will get an error message if the size of  $x$  is not the same as the size of  $y$ . The result  $z$  will have the same size as  $x$  and  $y$ .

## Array Division 1

The definition of array division is similar to the definition of array multiplication except that the elements of one array are divided by the elements of the other array. Both arrays must have the same size. The symbol for array right division is `./`. For example, if

```
x = [8, 12, 15]    y = [-2, 6, 5]
```

then  $z = x ./ y$  gives

```
z = [8/(-2), 12/6, 15/5] = [-4, 2, 3]
```

## Array Division <sub>2</sub>

Also, if

$$\mathbf{A} = \begin{bmatrix} 24 & 20 \\ -9 & 4 \end{bmatrix} \quad \mathbf{B} = \begin{bmatrix} -4 & 5 \\ 3 & 2 \end{bmatrix}$$

then  $\mathbf{C} = \mathbf{A} ./ \mathbf{B}$  gives

$$\mathbf{C} = \begin{bmatrix} 24/(-4) & 20/5 \\ -9/3 & 4/2 \end{bmatrix} = \begin{bmatrix} -6 & 4 \\ -3 & 2 \end{bmatrix}$$

Q4- Given the vectors  $\mathbf{x} = [6 \ 5 \ 10 \ 12]$  and  $\mathbf{y} = [3 \ 9 \ 8 \ 1]$

The product  $\mathbf{w} = \mathbf{x}.*\mathbf{y}$  is:

- a) 25 18 80 12
- b) 44 45 17 10
- c) 18 45 80 12 (correct)
- d) 62 15 10 8

```
>> x=[6,5,10,12];
>> y=[3,9,8,1];
>> w=x.*y;
>> w

w =

    18    45    80    12

>> z=x*y;
Error using .*
Inner matrix dimensions must agree.
```

Q5- Given the vectors  $x = [6 \ 5 \ 10 \ 12]$  and  $y = [3 \ 9 \ 8 \ 1]$

$z = \text{sqrt}(y) + \exp(x)$  :

```
>> x=[6,5,10,12];
>> y=[3,9,8,1];
>> z=sqrt(y)+exp(x);
>> z

z =

1.0e+05 *

0.0041    0.0015    0.2203    1.6276
```

a)  $10^5[0.165 \ 0.0005 \ 0.62 \ 4.44]$

b)  $10^5[0.114 \ 0.1010 \ 0.72 \ 3.61]$

c)  $10^5[0.204 \ 0.5110 \ 0.22 \ 2.03]$

d)  $10^5[0.0041 \ 0.0015 \ 0.2203 \ 1.6276]$  (correct)

© McGraw Hill LLC

Q6- Given the matrices  $A = \begin{bmatrix} 21 & 7 \\ -28 & 8 \end{bmatrix}$   $B = \begin{bmatrix} -7 & -3 \\ 9 & 4 \end{bmatrix}$

The array product  $w = A.*B$  and the array product  $z = B.*A$  are

a)  $\begin{bmatrix} -147 & -21 \\ -252 & 32 \end{bmatrix}$  (correct)

b)  $\begin{bmatrix} -147 & 21 \\ -252 & 32 \end{bmatrix}$

c)  $\begin{bmatrix} 147 & -21 \\ -252 & 32 \end{bmatrix}$

d)  $\begin{bmatrix} -147 & -21 \\ -252 & -32 \end{bmatrix}$

```
>> A=[21,7;-28,8];
>> B=[-7,-3;9,4];
>> w=A.*B;
>> w

w =

-147    -21
-252     32
```

```
>> z=B.*A;
>> z

z =

-147    -21
-252     32
```

© McGraw Hill LLC

Q7- Given the vectors  $x = [6 \ 15 \ 10]$  and  $y = [3 \ 19 \ 8]$   
 The array quotient  $w = x./y$  and  $z = y./x$  are

- a)  $W = 1 \ 0.9 \ 1.2, Z = 3 \ 1.4 \ 0.4$
- b)  $W = 8 \ 2.1 \ 3.5, Z = 1.5 \ 2.2 \ 1.8$
- c)  $W = 2 \ 0.78 \ 1.25, Z = 0.5 \ 1.26 \ 0.8$  (correct)
- d)  $W = 6 \ 5.1 \ 4.2, Z = 2 \ 2.6 \ 2.3$

```
>> x=[6,15,10];
>> y=[3,19,8];
>> w=x./y

w =

    2.0000    0.7895    1.2500

>> z=y./x

z =

    0.5000    1.2667    0.8000
```

© McGraw Hill LLC

Q8- Given the matrices  $A = \begin{bmatrix} 6 & 5 & 10 \\ 3 & 9 & 8 \end{bmatrix}$  and  $B = \begin{bmatrix} 3 & 9 & 8 \\ 4 & 5 & 6 \end{bmatrix}$

Use MATLAB to find  $C = A./B$  and  $E = A.\backslash B$ .

- a)  $C = [1 \ 2.1 \ 1.3; 1.8 \ 0.7 \ 0.3]$   
 $E = [1.5 \ 1.2 \ 2.8; 1.3 \ 0.6 \ 0.8]$
- b)  $C = [1.3 \ 2 \ 1.1; 0.4 \ 1.1 \ 1.1]$   
 $E = [1.3 \ 0.6 \ 0.8; 0.5 \ 1.8 \ 0.8]$
- c)  $C = [-0.2 \ 1.6 \ 0.3; 1.3 \ 1 \ 1.4]$   
 $E = [0.3 \ 1.1 \ 0.3; 1.1 \ 1.6 \ 1.8]$
- d)  $C = [2 \ 0.6 \ 1.3; 0.8 \ 1.8 \ 1.3]$   
 $E = [0.5 \ 1.8 \ 0.8; 1.3 \ 0.6 \ 0.8]$   
 (correct)

```
>> A=[6,5,10;3,9,8];
>> B=[3,9,8;4,5,6];
>> C=A./B

C =

    2.0000    0.5556    1.2500
    0.7500    1.8000    1.3333

>> E=A.\B

E =

    0.5000    1.8000    0.8000
    1.3333    0.5556    0.7500
```

```
>> C.*E

ans =

    1    1    1
    1    1    1
```

© McGraw Hill LLC

Q9- Given the matrices  $A = \begin{bmatrix} 21 & 7 \\ -28 & 8 \end{bmatrix}$  and  $B = \begin{bmatrix} 2 & 7 \\ 12 & 5 \end{bmatrix}$   
 Their array product is

a)  $C = \begin{bmatrix} 42 & 49 \\ -336 & 40 \end{bmatrix}$  (correct)

b)  $C = \begin{bmatrix} -142 & 29 \\ 14 & 63 \end{bmatrix}$

c)  $C = \begin{bmatrix} 234 & 249 \\ 162 & -40 \end{bmatrix}$

d)  $C = \begin{bmatrix} 21 & 43 \\ -136 & 48 \end{bmatrix}$

```
>> A=[21,7;-28,8];
>> B=[2,7;12,5];
>> C=A.*B

C =

    42    49
   -336    40

>> D=B.*A

D =

    42    49
   -336    40
```

© McGraw Hill LLC

Q10- Given the matrices  $B = \begin{bmatrix} 2 & 7 \\ 12 & 5 \end{bmatrix}$

Use MATLAB to calculate B raised to the third power element by element. The answer is

a)  $B = \begin{bmatrix} 1232 & 217 \\ 104 & 115 \end{bmatrix}$

b)  $B = \begin{bmatrix} 8 & 343 \\ 1728 & 125 \end{bmatrix}$  (correct)

c)  $B = \begin{bmatrix} 801 & 313 \\ 1725 & 25 \end{bmatrix}$

d)  $B = \begin{bmatrix} 128 & 423 \\ 1725 & 125 \end{bmatrix}$

```
>> B=[2,7;12,5];
>> C=B.^3;
>> C

C =

     8    343
   1728    125
```

© McGraw Hill LLC

## Array Exponentiation <sub>1</sub>

MATLAB enables us not only to raise arrays to powers but also to raise scalars and arrays to *array* powers.

To perform exponentiation on an element-by-element basis, we must use the `.^` symbol.

For example, if `x = [3, 5, 8]`, then typing `x.^3` produces the array `[33, 53, 83] = [27, 125, 512]`.

## Array Exponentiation <sub>2</sub>

We can raise a scalar to an array power. For example, if `p = [2, 4, 5]`, then typing `3.^p` produces the array `[32, 34, 35] = [9, 81, 243]`.

Remember that `.^` is a *single* symbol. The dot in `3.^p` is *not* a *decimal point* associated with the number 3. The following operations, with the value of `p` given here, are equivalent and give the correct answer:

`3.^p`

`3.0.^p`

`3. .^p`

`(3).^p`

`3.^ [2, 4, 5]`



## Matrix-Matrix Multiplication <sub>1</sub>

In the product of two matrices  $\mathbf{AB}$ , the number of *columns* in  $\mathbf{A}$  must equal the number of *rows* in  $\mathbf{B}$ . The row-column multiplications form column vectors, and these column vectors form the matrix result. The product  $\mathbf{AB}$  has the same number of *rows* as  $\mathbf{A}$  and the same number of *columns* as  $\mathbf{B}$ . For example,

$$\begin{bmatrix} 6 & -2 \\ 10 & 3 \\ 4 & 7 \end{bmatrix} \begin{bmatrix} 9 & 8 \\ -5 & 12 \end{bmatrix} = \begin{bmatrix} (6)(9) + (-2)(-5) & (6)(8) + (-2)(12) \\ (10)(9) + (3)(-5) & (10)(8) + (3)(12) \\ (4)(9) + (7)(-5) & (4)(8) + (7)(12) \end{bmatrix}$$

$$= \begin{bmatrix} 64 & 24 \\ 75 & 116 \\ 1 & 116 \end{bmatrix}$$

© McGraw Hill LLC

49

## Matrix-Matrix Multiplication <sub>2</sub>

Use the operator `*` to perform matrix multiplication in MATLAB. The following MATLAB session shows how to perform matrix multiplication.

```
>>A = [6,-2;10,3;4,7];
```

```
>>B = [9,8;-5,12];
```

```
>>A*B
```

```
ans =
```

```
64    24
```

```
75    116
```

```
1     116
```

© McGraw Hill LLC

50

## Matrix-Matrix Multiplication <sub>3</sub>

Matrix multiplication does not have the commutative property; that is, in general,  $\mathbf{AB} \neq \mathbf{BA}$ . A simple example will demonstrate this fact:

$$\mathbf{AB} = \begin{bmatrix} 6 & -2 \\ 10 & 3 \end{bmatrix} \begin{bmatrix} 9 & 8 \\ -12 & 14 \end{bmatrix} = \begin{bmatrix} 78 & 20 \\ 54 & 122 \end{bmatrix}$$

whereas

$$\mathbf{BA} = \begin{bmatrix} 9 & 8 \\ -12 & 14 \end{bmatrix} \begin{bmatrix} 6 & -2 \\ 10 & 3 \end{bmatrix} = \begin{bmatrix} 134 & 6 \\ 68 & 66 \end{bmatrix}$$

Reversing the order of matrix multiplication is a common and easily made mistake.

© McGraw Hill LLC

51

Q11- Given the vectors  $\mathbf{x} = \begin{bmatrix} 6 \\ 7 \\ 3 \end{bmatrix}$   $\mathbf{y} = [2 \ 4 \ 7]$

Use MATLAB to find the matrix product  $\mathbf{w} = \mathbf{x} * \mathbf{y}$  and  $\mathbf{z} = \mathbf{y} * \mathbf{x}$ .

a)  $\mathbf{w} = \begin{bmatrix} 6 & 2 & 21 \\ 14 & 28 & 49 \\ 12 & 24 & 42 \end{bmatrix}$ ,  $\mathbf{z} = 78$

b)  $\mathbf{w} = 61$ ,  $\mathbf{z} = \begin{bmatrix} 12 & 24 & 42 \\ 14 & 28 & 49 \\ 6 & 2 & 21 \end{bmatrix}$

c)  $\mathbf{w} = \begin{bmatrix} 12 & 24 & 42 \\ 14 & 28 & 49 \\ 6 & 12 & 21 \end{bmatrix}$ ,  $\mathbf{z} = 61$  (correct)

d)  $\mathbf{W} = 78$ ,  $\mathbf{z} = \begin{bmatrix} 6 & 2 & 21 \\ 14 & 28 & 49 \\ 12 & 24 & 42 \end{bmatrix}$

```
>> x=[6;7;3];
>> y=[2,4,7];
>> w=x*y;
>> z=y*x;
>> w

w =

    12    24    42
    14    28    49
     6    12    21

>> z

z =

    61
```

© McGraw Hill LLC

Q12- Use MATLAB to compute the dot product of the following vectors:

$$u = 6i - 12j + 4k \quad w = 7i + 3j - 4k$$

- a) -6
- b) 7
- c) -10 (correct)
- d) 12

```
>> u=[6,-12,4];
>> w=[7;3;-4];
>> z=u*w

z =

    -10
```

© McGraw Hill LLC

## Special Matrices <sup>1</sup>

Two exceptions to the noncommutative property are the *null* or *zero* matrix, denoted by **0** and the *identity*, or *unity*, matrix, denoted by **I**.

The null matrix contains all zeros and is not the same as the *empty* matrix [ ], which has no elements.

These matrices have the following properties:

$$\mathbf{0A} = \mathbf{A0} = \mathbf{0}$$

$$\mathbf{IA} = \mathbf{AI} = \mathbf{A}$$

© McGraw Hill LLC

54

## Special Matrices <sub>2</sub>

The identity matrix is a square matrix whose diagonal elements are all equal to one, with the remaining elements equal to zero.

For example, the  $2 \times 2$  identity matrix is

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

The functions `eye(n)` and `eye(size(A))` create an  $n \times n$  identity matrix and an identity matrix the same size as the matrix **A**.

## Special Matrices <sub>3</sub>

Sometimes we want to initialize a matrix to have all zero elements. The `zeros` command creates a matrix of all zeros.

Typing `zeros(n)` creates an  $n \times n$  matrix of zeros, whereas typing `zeros(m,n)` creates an  $m \times n$  matrix of zeros.

Typing `zeros(size(A))` creates a matrix of all zeros having the same dimension as the matrix **A**. This type of matrix can be useful for applications in which we do not know the required dimension ahead of time.

The syntax of the `ones` command is the same, except that it creates arrays filled with ones.

## Matrix Left Division and Linear Equations

$$6x + 12y + 4z = 70$$

$$7x - 2y + 3z = 5$$

$$2x + 8y - 9z = 64$$

```
>>A = [6,12,4;7,-2,3;2,8,-9];
```

```
>>B = [70;5;64];
```

```
>>Solution = A\B
```

```
Solution =
```

```
3
```

```
5
```

```
-2
```

The solution is  $x = 3$ ,  $y = 5$ , and  $z = -2$ .

© McGraw Hill LLC

57

Q13- Use MATLAB to solve the following set of equations.

$$14x - 2y = 6$$

$$3x + 15y = -1$$

a) 0.41, -0.15 (correct)

b) -0.67, 0.71

c) 0.93, 0.17

d) 1.41, -2.43

```
>> A=[14,-2;3,15];
```

```
>> B=[6;-1];
```

```
>> Solution=A\B
```

```
Solution =
```

```
0.4074
```

```
-0.1481
```

© McGraw Hill LLC

Q14- Use MATLAB to solve the following set of equations.

$$\begin{aligned} 6x + 4y - 8z &= 12 \\ 15x - 3y + 7z &= 17 \\ 4x + 9y - 5z &= -6 \end{aligned}$$

- a) 2.1, 3.1, -2.5
- b) 1.4, -2.1, -1.5 (correct)
- c) 1.2, -0.1, 0.5
- d) 4.4, -8.1, 3.5

```
>> A=[6,4,-8;15,-3,7;4,9,-5];
>> B=[12;17;-6];
>> Solution=A\B

Solution =

    1.4106
   -2.1312
   -1.5076
```

© McGraw Hill LLC

## Polynomial Coefficients

The function `poly(r)` computes the coefficients of the polynomial whose roots are specified by the vector `r`. The result is a *row* vector that contains the polynomial's coefficients arranged in descending order of power.

For example,

```
>>c = poly([-2, -5])
```

```
c =
```

```
    1    7   10
```

© McGraw Hill LLC

60

## Polynomial Roots

The function `roots(a)` computes the roots of a polynomial specified by the coefficient array `a`. The result is a *column* vector that contains the polynomial's roots.

For example,

```
>>r = roots([2, 14, 20])

r =

    -2
   -5
```

© McGraw Hill LLC

61

Q15- Use MATLAB to obtain the roots of  

$$x^3 + 23x^2 + 15x + 6 = 0$$
  
 Use the poly function to confirm your answer.

- a) -22.3 , -0.3 + 0.4i, -0.3 - 0.4i (correct)
- b) -22.3 , 0.3 - 0.4i, 0.3 - 0.4i
- c) -22.3 , -0.3 - 0.4i, 0.3 - 0.4i
- d) -22.3 , 0.3 - 0.4i, -0.3 - 0.4i

```
>> r=roots([1,23,15,6])

r =

-22.3406 + 0.0000i
 -0.3297 + 0.3998i
 -0.3297 - 0.3998i

>> s=poly(r)

s =

    1.0000    23.0000    15.0000     6.0000
```

© McGraw Hill LLC

## Polynomial Multiplication and Division

The function `conv(a,b)` computes the product of the two polynomials described by the coefficient arrays `a` and `b`. The two polynomials need not be the same degree. The result is the coefficient array of the product polynomial.

The function `[q,r] = deconv(num,den)` computes the result of dividing a numerator polynomial, whose coefficient array is `num`, by a denominator polynomial represented by the coefficient array `den`. The quotient polynomial is given by the coefficient array `q`, and the remainder polynomial is given by the coefficient array `r`.

© McGraw Hill LLC

63

## Polynomial Multiplication and Division: Examples

```
>>a = [9,-5,3,7];
>>b = [6,-1,2];
>>product = conv(a,b)

product =
    54   -39    41    29    -1    14

>>[quotient, remainder] = deconv(a,b)

quotient =
    1.5   -0.5833

remainder =
    0    0   -0.5833    8.1667
```

© McGraw Hill LLC

64



Q16- Use MATLAB to calculate the quotient and the remainder of

$$\frac{18x^3 - 5x^2 - 2x + 3}{3x^2 + 7x + 4}$$

- a)  $q = 6x - 15.7, \quad r = 83.7x - 65.7$
- b)  $q = -6x + 15.7, \quad r = 83.7x + 65.7$
- c)  $q = 6x - 15.7, \quad r = 83.7x + 65.7$  (correct)
- d)  $q = 6x + 15.7, \quad r = 83.7x + 65.7$

```
>> a=[18,-5,-2,3];
>> b=[3,7,4];
>> [q,r]=deconv(a,b)

q =

    6.0000   -15.6667

r =

         0         0   83.6667   65.6667
```

© McGraw Hill LLC

## Plotting Polynomials

The function `polyval(a,x)` evaluates a polynomial at specified values of its independent variable  $x$ , which can be a matrix or a vector. The polynomial's coefficients of descending powers are stored in the array  $a$ . The result is the same size as  $x$ .

© McGraw Hill LLC

66

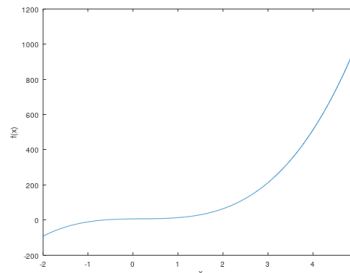
## Example of Plotting a Polynomial

To plot the polynomial  $f(x) = 9x^3 - 5x^2 + 3x + 7$  for  $-2 \leq x \leq 5$ , you type

```
>>a = [9,-5,3,7];
>>x = -2:0.01:5;
>>f = polyval(a,x);
>>plot(x,f),xlabel('x'),ylabel('f(x)')
```

OR

```
>> x=-2:0.01:5;
>> f=9*x.^3-5*x.^2+3.*x+7;
>> plot(x,f)
```



© McGraw Hill LLC

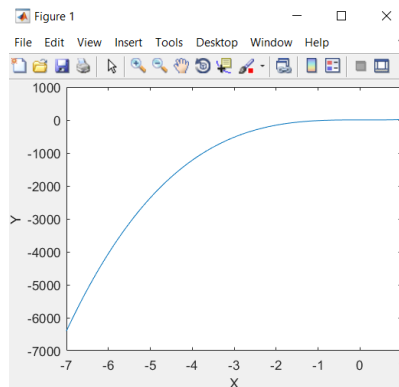
67

Q17- Plot the polynomial

$$y = 18x^3 - 5x^2 - 2x + 3$$

over the range  $-7 \leq x \leq 1$ .

```
>> a=[18,-5,-2,3];
>> x=-7:0.01:1;
>> f=polyval(a,x);
>> plot(x,f),xlabel('X'),ylabel('Y')
```



© McGraw Hill LLC

## Cell Array Functions

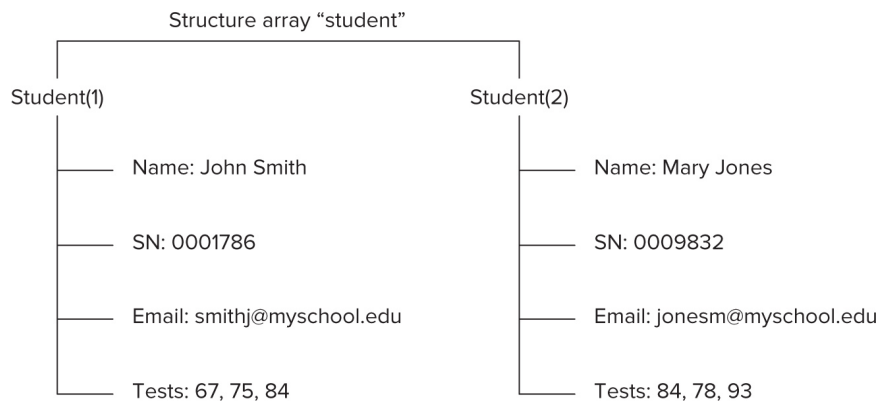
Function	Description
<code>C = cell(n)</code>	Creates an $n \times n$ cell array <code>C</code> of empty matrices.
<code>C = cell(n,m)</code>	Creates an $n \times m$ cell array <code>C</code> of empty matrices.
<code>celldisp(C)</code>	Displays the contents of cell array <code>C</code> .
<code>cellplot(C)</code>	Displays a graphical representation of the cell array <code>C</code> .

<https://www.mathworks.com/help/matlab/ref/cell.html>

© McGraw Hill LLC

69

## Arrangement of data in the structure array "student".



<https://tinyurl.com/2ubuhpt6>

© McGraw Hill LLC

70

## Structure Functions <sub>1</sub>

Function	Description
<code>names = fieldnames(S)</code>	Returns the field names associated with the structure array <code>S</code> as <code>names</code> , a cell array of strings.
<code>isfield(S, 'field')</code>	Returns 1 if 'field' is the name of a field in the structure array <code>S</code> , and 0 otherwise.
<code>isstruct(S)</code>	Returns 1 if the array <code>S</code> is a structure and 0 otherwise.

## Structure Functions <sub>2</sub>

Function	Description
<code>S = rmfield(S, 'field')</code>	Removes the field 'field' from the structure array <code>S</code> .
<code>S = struct('f1', 'v1', 'f2', 'v2', ...)</code>	Creates a structure array with the fields 'f1', 'f2', ... having the values 'v1', 'v2', ...



Because learning changes everything.®

[www.mheducation.com](http://www.mheducation.com)

---

© McGraw Hill LLC. All rights reserved. No reproduction or distribution without the prior written consent of McGraw Hill LLC.