Digital content of Mathematics

Ravish

[redacted]

1 Introduction to Differential Equations

A differential equation is an equation that involves one or more derivatives of an unknown function. Unlike algebraic equations, which involve only algebraic operations, differential equations include derivatives, making them more suitable for modeling processes that change continuously. For example, the equation describing the rate of change of a population over time is a differential equation.

2 First-Order Differential Equations

A first-order differential equation involves only the first derivative of the unknown function. An example is the separable differential equation: $\frac{dy}{dx} = x^2$.

Separation of variables is a technique used to solve first-order separable differential equations. For $\frac{dy}{dx} = x^2$, you can separate variables as $\frac{dy}{y} = x^2 dx$. Integrating both sides gives $\ln |y| = \frac{x^3}{3} + C$, where C is the constant of integration.

To solve a first-order linear differential equation, $y' + P(x)y = Q(x)$, you can use an integrating factor, denoted by $I(x)$. The integrating factor is $I(x) = e^{\int P(x)dx}$. Multiply both sides of the equation by the integrating factor to obtain $I(x)y' + P(x)I(x)y = I(x)Q(x)$. The left side can be simplified to $\frac{d}{dx}(I(x)y) = I(x)Q(x)$. Integrate both sides to find $I(x)y = \int I(x)Q(x)dx + C$. Solve for y to get $y = \frac{1}{I(x)} \left(\int I(x)Q(x)dx + C \right)$.

2.1 Differential Equation:

$\frac{dy}{dx} = x^2 y$

Solution: Separating variables, we get:

$$\frac{dy}{y} = x^2 dx$$

Integrating both sides gives:

$$\int \frac{1}{y} dy = \int x^2 dx$$

$$\ln|y| = \frac{x^3}{3} + C_1$$

Solving for y:

$$y = Ce^{\frac{x^3}{3}}$$

2.2 Differential Equation:

$\frac{dy}{dx} + 2xy = 0$

Solution: This is a first-order linear homogeneous ODE.

The integrating factor is $e^{\int 2xdx} = e^{x^2}$

Multiplying both sides by the integrating factor:

$$e^{x^2}\frac{dy}{dx} + 2xye^{x^2} = 0$$

Recognizing the left side as the derivative of (ye^{x^2}):

$$\frac{d}{dx}(ye^{x^2}) = 0$$

Integrating gives:

$$ye^{x^2} = C_2$$

Solving for y:

$$y = \frac{C_2}{e^{x^2}} = \frac{C_2}{\sqrt{x}}$$

2.3 Differential Equation:

$$(2xy + y^2)dx + (x^2 + 2xy)dy = 0$$

Solution: We need to check if this equation is exact by verifying if

$$\frac{\partial M}{\partial y} = \frac{\partial N}{\partial x}$$

.

$M = 2xy + y^2$, $\frac{\partial M}{\partial y} = 2x + 2y$
$N = x^2 + 2xy$, $\frac{\partial N}{\partial x} = 2x + 2y$

Since

$$\frac{\partial M}{\partial y} = \frac{\partial N}{\partial x}$$

, the equation is exact.

We find the potential function U such that

$$\frac{\partial U}{\partial x} = M$$

and

$$\frac{\partial U}{\partial y} = N$$

.

Integrating M with respect to x gives:

$$U = x^2y + \frac{1}{3}y^3 + h(y)$$

Differentiating U with respect to y to match with N gives:

$$\frac{\partial U}{\partial y} = x^2 + y^2 + h'(y) = x^2 + 2xy$$

$$h'(y) = 2xy$$

, integrating gives

$$h(y) = x^2y + C_3$$

So,

$$U = x^2y + \frac{1}{3}y^3 + x^2y + C_3$$

$$U = 2x^2y + \frac{1}{3}y^3 + C_3$$

The solution is given by

$$U = C$$

, where

$$C = 2x^2y + \frac{1}{3}y^3 + C_3$$

2.4 Differential Equation:

$$\frac{dy}{dx} = \frac{x^2 + y^2}{xy}$$

Solution: Let's make a substitution:

$$y = vx$$

Differentiating both sides with respect to x:

$$\frac{dy}{dx} = v + x\frac{dv}{dx}$$

Substitute this back into the differential equation:

$$v + x\frac{dv}{dx} = \frac{x^2 + v^2x^2}{xv}$$

Simplify and rearrange:

$$v\frac{dv}{dx} = \frac{x^2(1 + v^2)}{xv} - v$$

Divide by $v(1+v^2)$:

$$\frac{1}{v(1+v^2)}dv = \frac{x}{x^2+x^2v^2}dx$$

Integrate both sides:

$$\int \frac{1}{v(1+v^2)}dv = \int \frac{1}{x(1+v^2)}dx$$

This can be solved using partial fraction decomposition.
After integration, you'll get:

$$\ln|v| = \frac{1}{2}\ln|x^2+1| + C_4$$

Substitute back

$$v = \frac{y}{x}$$

:

$$\ln\left|\frac{y}{x}\right| = \frac{1}{2}\ln|x^2+1| + C_4$$

Taking the exponential of both sides:

$$\left|\frac{y}{x}\right| = e^{C_4}\sqrt{x^2+1}$$

Solving for y:

$$|y| = C_5\sqrt{x^2+1}$$

The general solution involves two constants, accounting for both positive and negative branches.

2.5 Differential Equation:

$$\frac{dy}{dx} - 2xy = y^2x^3$$

Solution: Divide the equation by y^2:

$$\frac{1}{y^2}\frac{dy}{dx} - 2x\frac{1}{y} = x^3$$

Make a substitution:

$$z = \frac{1}{y}$$

,

$$\frac{dz}{dx} = -\frac{1}{y^2}\frac{dy}{dx}$$

Substitute into the equation:

$$\frac{dz}{dx} + 2xz = -x^3$$

This is now a linear first-order ODE, which we solved earlier.
The integrating factor is

$$e^{\int 2xdx} = e^{x^2}$$

Multiply both sides by the integrating factor:

$$e^{x^2}\frac{dz}{dx} + 2xze^{x^2} = -x^3e^{x^2}$$

Recognizing the left side as the derivative of (ze^{x^2}):

$$\frac{d}{dx}(ze^{x^2}) = -x^3e^{x^2}$$

Integrating gives:

$$ze^{x^2} = -\frac{1}{4}x^2e^{x^2} + C_6$$

Substitute back

$$z = \frac{1}{y}$$

:

$$\frac{1}{y}e^{x^2} = -\frac{1}{4}x^2e^{x^2} + C_6$$

Solving for y:

$$y = \frac{1}{-\frac{1}{4}x^2 + C_6e^{-x^2}}$$

This can be further simplified as provided in the earlier response.

3 Second-Order Linear Differential Equations

A second-order linear differential equation involves the second derivative of the unknown function. An example is $y'' - 3y' + 2y = 0$.

The characteristic equation of a second-order linear homogeneous differential equation, $ay'' + by' + cy = 0$, is obtained by substituting $y = e^{rx}$ and solving the resulting quadratic equation $ar^2 + br + c = 0$. The solutions of this quadratic equation are the roots r_1 and r_2, which determine the form of the general solution: $y(x) = C_1e^{r_1x} + C_2e^{r_2x}$.

The complementary solution is the general solution of the associated homogeneous equation $(ay'' + by' + cy = 0)$ without considering any particular solution. The particular solution is a specific solution of the nonhomogeneous equation $(ay'' + by' + cy = g(x))$. The general solution of the nonhomogeneous equation is the sum of the complementary solution and the particular solution.

3.1 Differential Equation:

$$\frac{d^2y}{dx^2} + 3\frac{dy}{dx} + 2y = 0$$

Solution: The characteristic equation is:

$$r^2 + 3r + 2 = 0$$

Factoring the characteristic polynomial:

$$(r + 2)(r + 1) = 0$$

So, the characteristic roots are $r_1 = -2$ and $r_2 = -1$.
The general solution of the homogeneous equation is:

$$y(x) = C_1e^{-2x} + C_2e^{-x}$$

3.2 Differential Equation:

$$\frac{d^2y}{dx^2} - 4\frac{dy}{dx} + 4y = 0$$

Solution: The characteristic equation is:

$$r^2 - 4r + 4 = 0$$

Factoring the characteristic polynomial:

$$(r - 2)^2 = 0$$

So, the characteristic root is $r = 2$ with multiplicity 2.
The general solution of the homogeneous equation is:

$$y(x) = (C_1 + C_2x)e^{2x}$$

3.3 Differential Equation:

$$\frac{d^2y}{dx^2} + 6\frac{dy}{dx} + 9y = 0$$

Solution: The characteristic equation is:

$$r^2 + 6r + 9 = 0$$

Factoring the characteristic polynomial:

$$(r + 3)^2 = 0$$

So, the characteristic root is $r = -3$ with multiplicity 2.
The general solution of the homogeneous equation is:

$$y(x) = (C_1 + C_2x)e^{-3x}$$

3.4 Differential Equation:

$$\frac{d^2y}{dx^2} - 7\frac{dy}{dx} + 10y = 0$$

Solution: The characteristic equation is:

$$r^2 - 7r + 10 = 0$$

Factoring the characteristic polynomial:

$$(r - 5)(r - 2) = 0$$

So, the characteristic roots are $r_1 = 5$ and $r_2 = 2$.
The general solution of the homogeneous equation is:

$$y(x) = C_1e^{5x} + C_2e^{2x}$$

3.5 Differential Equation:

$$\frac{d^2y}{dx^2} - 2\frac{dy}{dx} + y = 0$$

Solution: The characteristic equation is:

$$r^2 - 2r + 1 = 0$$

Factoring the characteristic polynomial:

$$(r - 1)^2 = 0$$

So, the characteristic root is $r = 1$ with multiplicity 2.
The general solution of the homogeneous equation is:

$$y(x) = (C_1 + C_2x)e^x$$

3.6 Differential Equation:

$$\frac{d^2y}{dx^2} + 8\frac{dy}{dx} + 16y = 0$$

Solution: The characteristic equation is:

$$r^2 + 8r + 16 = 0$$

Factoring the characteristic polynomial:

$$(r + 4)^2 = 0$$

So, the characteristic root is $r = -4$ with multiplicity 2.
The general solution of the homogeneous equation is:

$$y(x) = (C_1 + C_2x)e^{-4x}$$

MATLAB

- The name MATLAB stands for MATrix LABoratory.
- MATLAB [1] is a high-performance language for technical computing.
- It integrates computation, visualization, and programming environment.
- Furthermore, MATLAB is a modern programming language environment: it has sophisticated data structures, contains built-in editing and debugging tools, and supports object-oriented programming

- MATLAB has many advantages compared to conventional computer languages (e.g., C, FORTRAN) for solving technical problems.
- MATLAB is an interactive system whose basic data element is an array that does not require dimensioning.
- It has powerful built-in routines that enable a very wide variety of computations.
- It also has easy to use graphics commands that make the visualization of results immediately available.
- Specific applications are collected in packages referred to as toolbox. There are toolboxes for signal processing, symbolic computation, control theory, simulation, optimization, and several other fields of applied science and engineering.

Features of MATLAB

Following are the basic features of MATLAB:

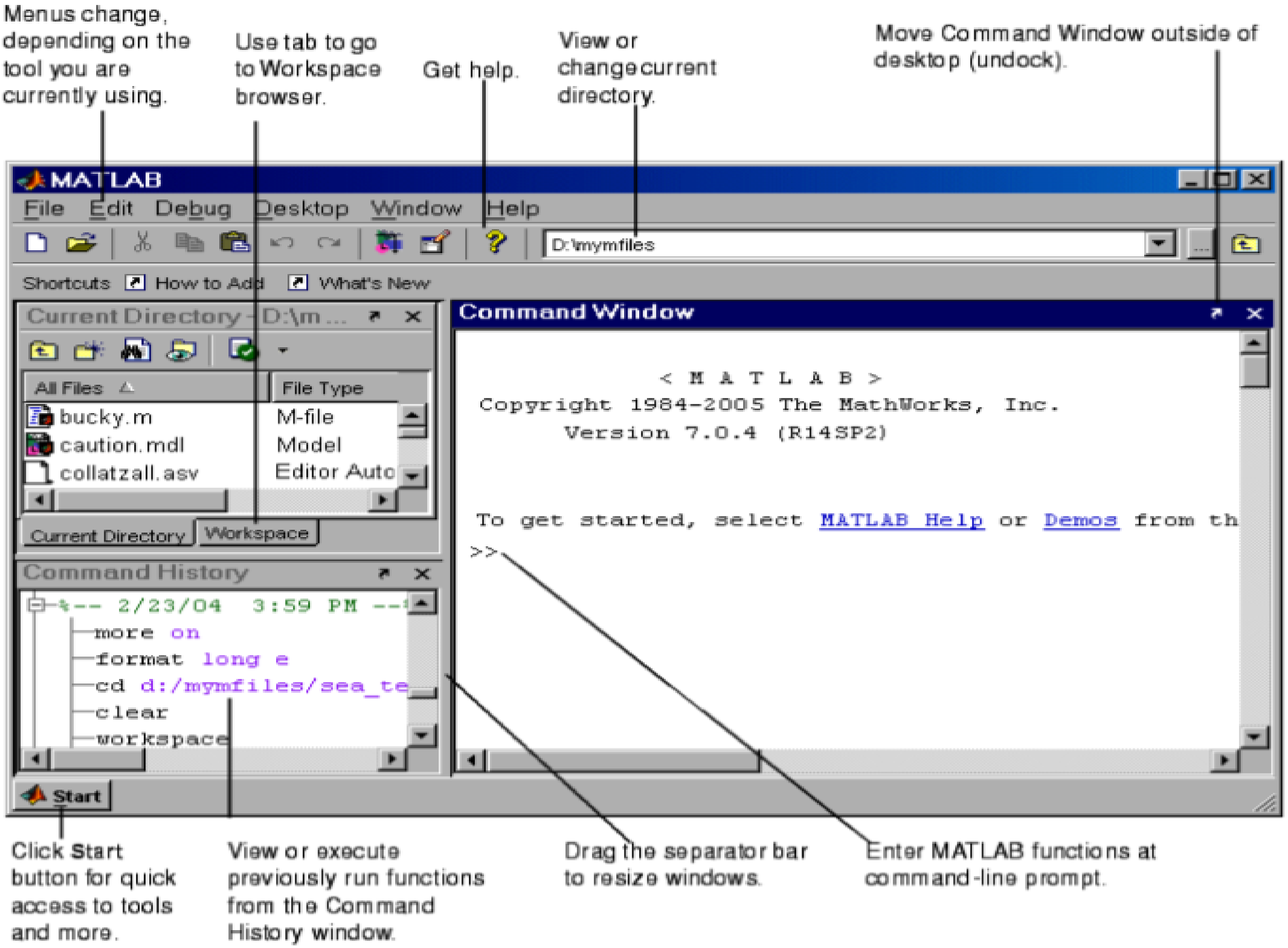
- It is a high-level language for numerical computation, visualization and application development.
- It also provides an interactive environment for iterative exploration, design and problem solving.
- It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
- It provides built-in graphics for visualizing data and tools for creating custom plots.
- MATLAB's programming interface gives development tools for improving code quality, maintainability, and maximizing performance.
- It provides tools for building applications with custom graphical interfaces.
- It provides functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET and Microsoft Excel.

Uses of MATLAB

- MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

1. signal processing and Communications
2. image and video Processing
3. control systems
4. test and measurement
5. computational finance
6. computational biology

- When you start MATLAB, a special window called the MATLAB desktop appears.
- The desktop is a window that contains other windows.
- The major tools within or accessible from the desktop are:
- The Command Window
- The Command History
- The Workspace
- The Current Directory
- The Help Browser
- The Start button



```
          < M A T L A B >
Copyright 1984-2005 The MathWorks, Inc.
       Version 7.0.4 (R14SP2)


To get started, select MATLAB Help or Demos from th
>>
```

```
%-- 2/23/04  3:59 PM --
   more on
   format long e
   cd d:/mymfiles/sea_te
   clear
   workspace
```

Menus change, depending on the tool you are currently using.

Use tab to go to Workspace browser.

Get help.

View or change current directory.

Move Command Window outside of desktop (undock).

Click Start button for quick access to tools and more.

View or execute previously run functions from the Command History window.

Drag the separator bar to resize windows.

Enter MATLAB functions at command-line prompt.

Figure 1.1: The graphical interface to the MATLAB workspace

```
>> 1+2*3
ans =7
>> x = 1+2*3
    x = 7
>>
```

Overwriting variable

- Once a variable has been created, it can be reassigned. In addition, if you do not wish to see the intermediate results, you can suppress the numerical output by putting a semicolon (;) at the end of the line. Then the sequence of commands looks like this:

```
>> t = 5;
>> t = t+1
 t = 6
```

Controlling the appearance of floating point number

- MATLAB by default displays only 4 decimals in the result of the calculations, for example −163.6667
- However, MATLAB does numerical calculations in double precision, which is 15 digits. The command format controls how the results of computations are displayed.

```
>> format short
>> x=-163.6667
```

- If we want to see all 15 digits, we use the command format long

```
>> format long
>> x= -1.636666666666667e+002
```

>> clear

- The command clear or clear all removes all variables from the workspace. This frees up system memory. In order to display a list of the variables currently in the memory, type

```
clear x                          % it will delete x, won't display anything
Clear                           % it will delete all variables in the workspace
                                  % peacefully and unobtrusively
```

>> who

while, whos will give more details which include size, space allocation, and class of the variables.

- The **whos command displays little more about the variables:**

1. Variables currently in memory
2. Type of each variables
3. Memory allocated to each variable
4. Whether they are complex variables or not

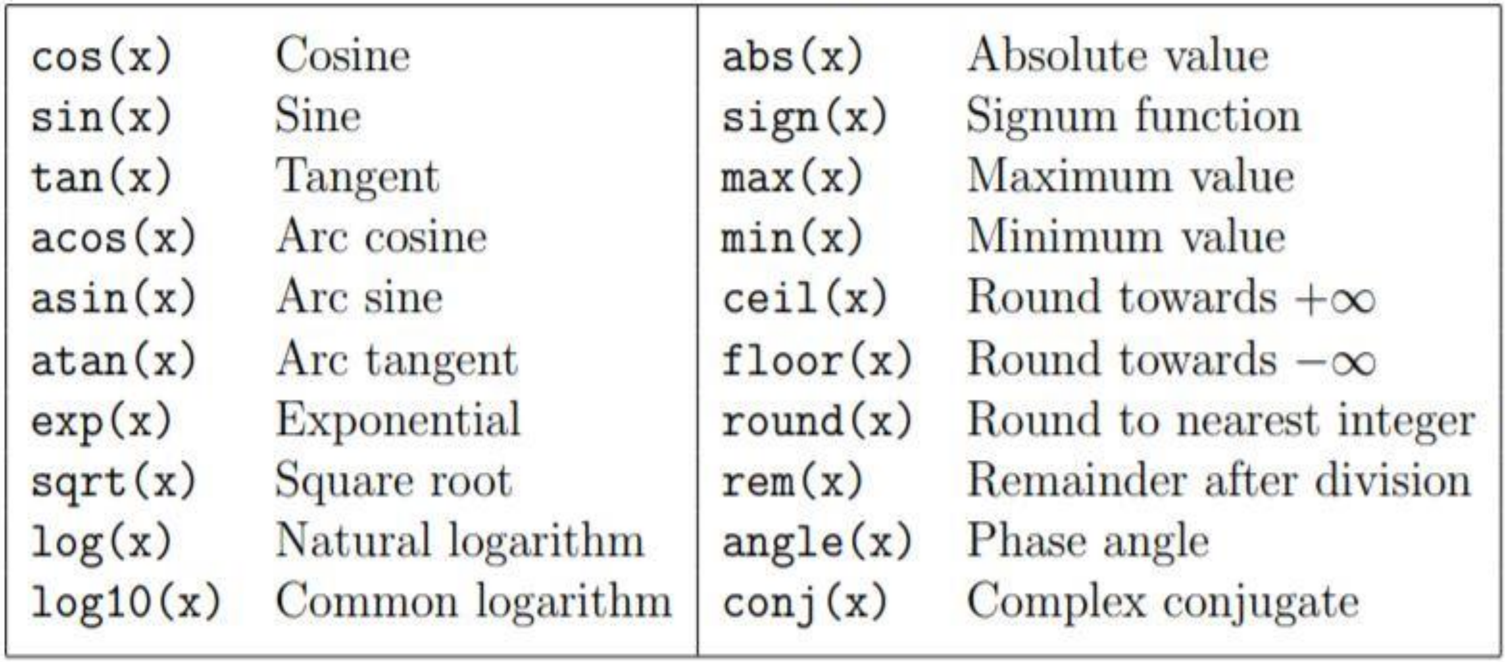
Entering multiple statements per line

- Entering multiple statements per line It is possible to enter multiple statements per line. Use commas (,) or semicolons (;) to enter more than one statement at once. Commas (,) allow multiple statements per line without suppressing output.

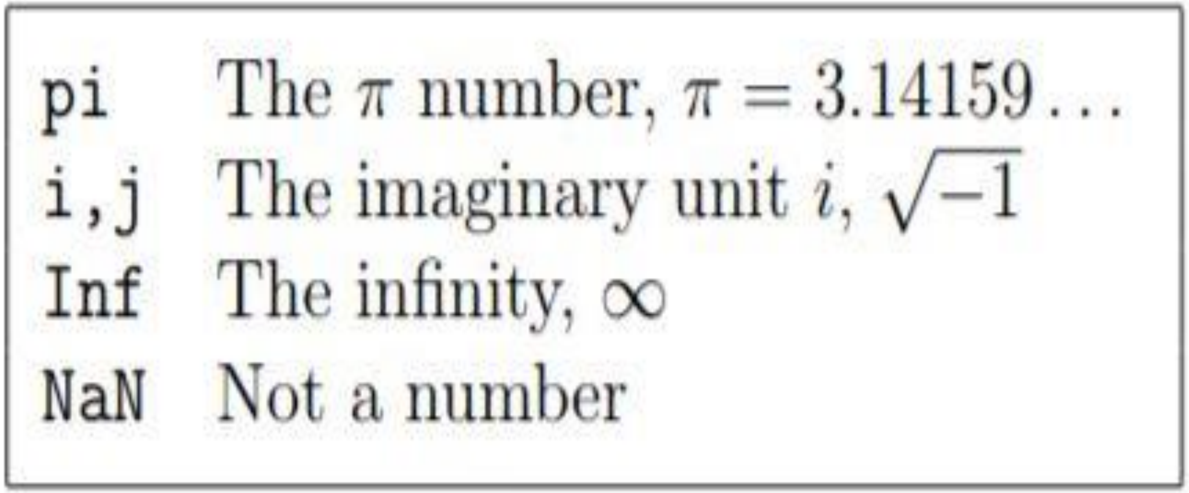
```
 >> a=7; b=cos(a), c=cosh(a)
b = 0.6570
c = 548.3170
```

Mathematical functions

- There is a long list of mathematical functions that are built into MATLAB. These functions are called built-ins.



`cos(x)`	Cosine	`abs(x)`	Absolute value
`sin(x)`	Sine	`sign(x)`	Signum function
`tan(x)`	Tangent	`max(x)`	Maximum value
`acos(x)`	Arc cosine	`min(x)`	Minimum value
`asin(x)`	Arc sine	`ceil(x)`	Round towards $+\infty$
`atan(x)`	Arc tangent	`floor(x)`	Round towards $-\infty$
`exp(x)`	Exponential	`round(x)`	Round to nearest integer
`sqrt(x)`	Square root	`rem(x)`	Remainder after division
`log(x)`	Natural logarithm	`angle(x)`	Phase angle
`log10(x)`	Common logarithm	`conj(x)`	Complex conjugate



`pi`	The π number, $\pi = 3.14159\ldots$
`i,j`	The imaginary unit i, $\sqrt{-1}$
`Inf`	The infinity, ∞
`NaN`	Not a number

```
>> a = 5; x = 2; y = 8;
>> y = exp(-a)*sin(x)+10*sqrt(y)
y = 28.2904

>> log(142)
ans = 4.9558

>> log10(142)
ans = 2.1523
```

```
>> sin(pi/4)
ans = 0.7071
>> exp(10)
ans = 2.2026e+004
```

Naming Variables

- Variable names consist of a letter followed by any number of letters, digits or underscore.
- MATLAB is case-sensitive**.**

Saving Your Work

- The save command is used for saving all the variables in the workspace, as a file with .mat extension, in the current directory.

For example,

```
save myfile
```

- You can reload the file anytime later using the load command.

```
load myfile
```

Basic plotting

- MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands.

```
>> x = [1 2 3 4 5 6];
 >> y = [3 -1 2 4 5 1];
>> plot(x,y)
>> x = 0:pi/100:2*pi;
>> y = sin(x);
>> plot(x,y)
```

Notes:

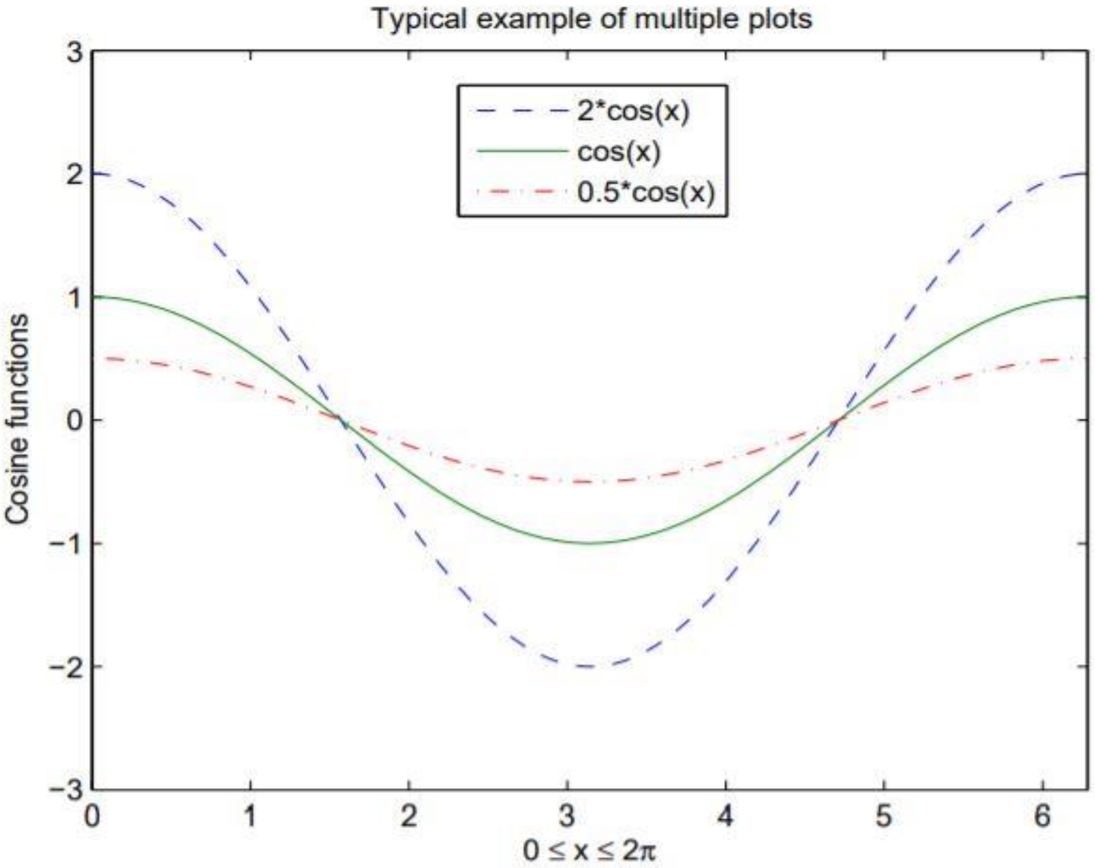
- 0:pi/100:2*pi yields a vector that
  - starts at 0,
  - takes steps (or increments) of π/100,
  - stops when 2π is reached.
- If you omit the increment, MATLAB automatically increments by 1

Adding titles, axis labels, and annotations

```
>> xlabel(’x = 0:2\pi’)
>> ylabel(’Sine of x’)
>> title(’Plot of the Sine function’)
```

Multiple data sets in one plot

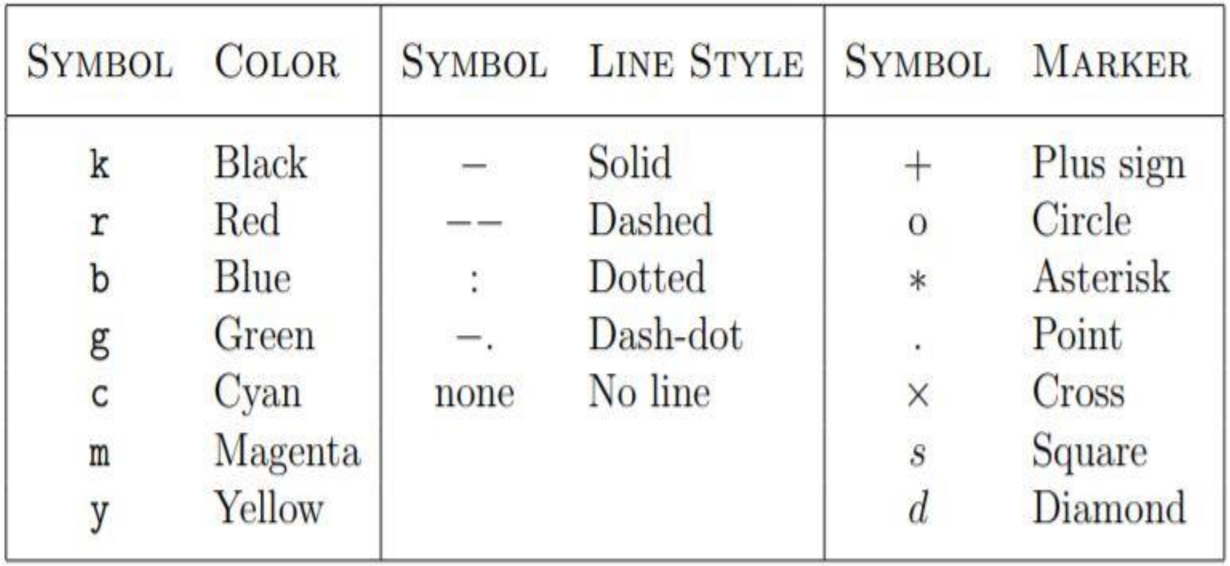
```
>> x = 0:pi/100:2*pi;
>> y1 = 2*cos(x);
>> y2 = cos(x);
>> y3 = 0.5*cos(x);
>> plot(x,y1,’--’,x,y2,’-’,x,y3,’:’)
>> xlabel(’0 \leq x \leq 2\pi’)
>> ylabel(’Cosine functions’)
>>legend(’2*cos(x)’,’cos(x)’,’0.5*cos(x)’)
>> title(’Typical example of multiple plots’)
>> axis([0 2*pi -3 3])
```


Typical example of multiple plots
2*cos(x)
cos(x)
0.5*cos(x)
Cosine functions
3
2
1
0
−1
−2
−3
0
1
2
3
4
5
6
0 ≤ x ≤ 2π

Specifying line styles and colors

- plot(x,y,'style_color_marker')
- where style_color_marker is a triplet of values from Table

Table 2.3: Attributes for `plot`



SYMBOL	COLOR	SYMBOL	LINE STYLE	SYMBOL	MARKER
k	Black	–	Solid	+	Plus sign
r	Red	––	Dashed	o	Circle
b	Blue	:	Dotted	*	Asterisk
g	Green	–.	Dash-dot	.	Point
c	Cyan	none	No line	×	Cross
m	Magenta			*s*	Square
y	Yellow			*d*	Diamond

- >> help Command
- >> lookfor inverse
- >> help sqrt
- >> doc plot

Matrices

- Matrices are the basic elements of the MATLAB environment.
- A matrix is a two-dimensional array consisting of m rows and n columns. Special cases are column vectors (n = 1) and row vectors (m = 1).

- an array of dimension 1 ×n is called a row vector, whereas an array of dimension m × 1 is called a column vector.
- The elements of vectors in MATLAB are enclosed by square brackets and are separated by spaces or by commas.
- For example, to enter a row vector, v, type
- `>> v = [1 4 7 10 13]`
- `v = 1 4 7 10 13`
- Column vectors are created in a similar way, however, semicolon (;) must separate the components of a column vector,
- `>> w = [1;4;7;10;13]`
- `w = 1 4 7 10 13`
- On the other hand, a row vector is converted to a column vector using the transpose operator. The transpose operation is denoted by an apostrophe or a single quote (’).

```
>> w = v’
w = 1
      4
      7
     10
     13
```

to access blocks of elements, we use MATLAB’s colon notation (:)

```
>> v(1:3)
 ans = 1 4 7
>> v(3:end)
 ans = 7 10 13
```

>> v(:) produces a column vector, whereas writing

>> v(1:end) produces a row vector.

To enter a matrix A, such as,

$$A = \begin{matrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{matrix}$$

type,

```
>> A = [1 2 3; 4 5 6; 7 8 9]
```

Often we must deal with matrices or vectors that are too large to enter one element at a time.

For example, suppose we want to enter a vector x consisting of points (0, 0.1, 0.2, 0.3, · · · , 5).

We can use the command

```
>> x = 0:0.1:5;
```

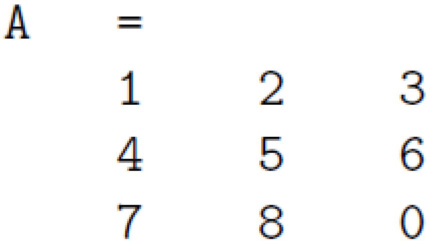
The row vector has 51 elements.

Matrix indexing

- The element of row i and column j of the matrix A is denoted by A(i,j). Thus, A(i,j) in MATLAB refers to the element Aij of matrix A.
- The first index is the row number and the second index is the column number. For example, A(1,3) is an element of first row and

third column. Here, A(1,3)=3.

- Correcting any entry is easy through indexing. Here we substitute A(3,3)=9 by
- A(3,3)=0. The result is
- >> A(3,3) = 0



```
A =
     1     2     3
     4     5     6
     7     8     0
```

Linear spacing

- On the other hand, there is a command to generate linearly spaced vectors: linspace.
- It is similar to the colon operator (:), but gives direct control over the number of points. For example,
- y = linspace(a,b)
- generates a row vector y of 100 points linearly spaced between and including a and b.
- y = linspace(a,b,n)
- generates a row vector y of n points linearly spaced between and including a and b. This is useful when we want to divide an interval into a number of subintervals of the same length.
- For example,
- >> theta = linspace(0,2*pi,101)
- divides the interval [0*; 2¼] into 100 equal subintervals, then creating a vector of 101 elements.*

Colon operator in a matrix

- The colon operator can also be used to pick out a certain row or column. For example, the statement A(m:n,k:l specifies rows m to n and column k to l)
- >> A(2,:)
- ans = 4 5 6 is the second row elements of A. The colon operator can also be used to extract a sub-matrix from a matrix A.
- >> A(:,2:3)
- A row or a column of a matrix can be deleted by setting it to a null vector, [ ].
- >> A(:,2)=[]

- To extract a submatrix B consisting of rows 2 and 3 and columns 1 and 2 of the matrix A, do the following
- >> B = A([2 3],[1 2])
- To interchange rows 1 and 2 of A, use the vector of row indices together with the colon operator.
- >> C = A([2 1 3],:)
- • A(:,j) is the jth column of A,
- while • A(i,:) is the ith row,
- and • A(end,:) picks out the last row of A.

- To delete a row or column of a matrix, use the empty vector operator, [ ].
- >> A(3,:) = []
- Third row of matrix A is now deleted. To restore the third row, we use a technique for creating a matrix
- >> A = [A(1,:);A(2,:);[7 8 0]]

- Dimension To determine the dimensions of a matrix or vector, use the command size. For example,
- >> size(A)
- Or more explicitly with,
- >> [m,n]=size(A)

Transposing a matrix

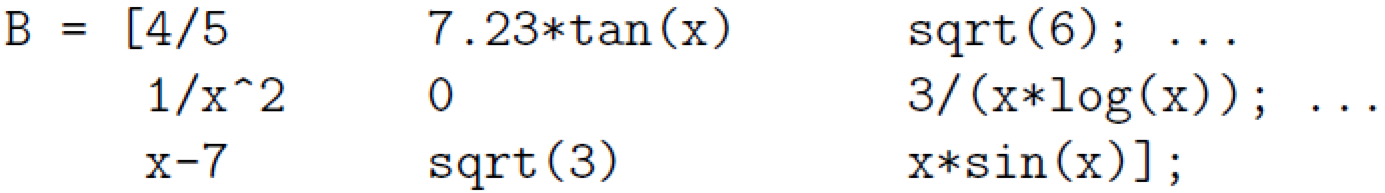
- The transpose operation is denoted by an apostrophe or a single quote (’). It flips a matrix about its main diagonal and it turns a row vector into a column vector. Thus,
- >> A’

Creating a sub-matrix

- To extract a *submatrix B consisting of rows 2 and 3 and columns 1 and 2 of the matrix A,* do the following
- >> B = A([2 3],[1 2])
- To interchange rows 1 and 2 of A, use the vector of row indices together with the colon operator.
- >> C = A([2 1 3],:)

Continuation

- If it is not possible to type the entire input on the same line, use consecutive periods, called
- an ellipsis *: : :, to signal continuation, then continue the input on the next line.*

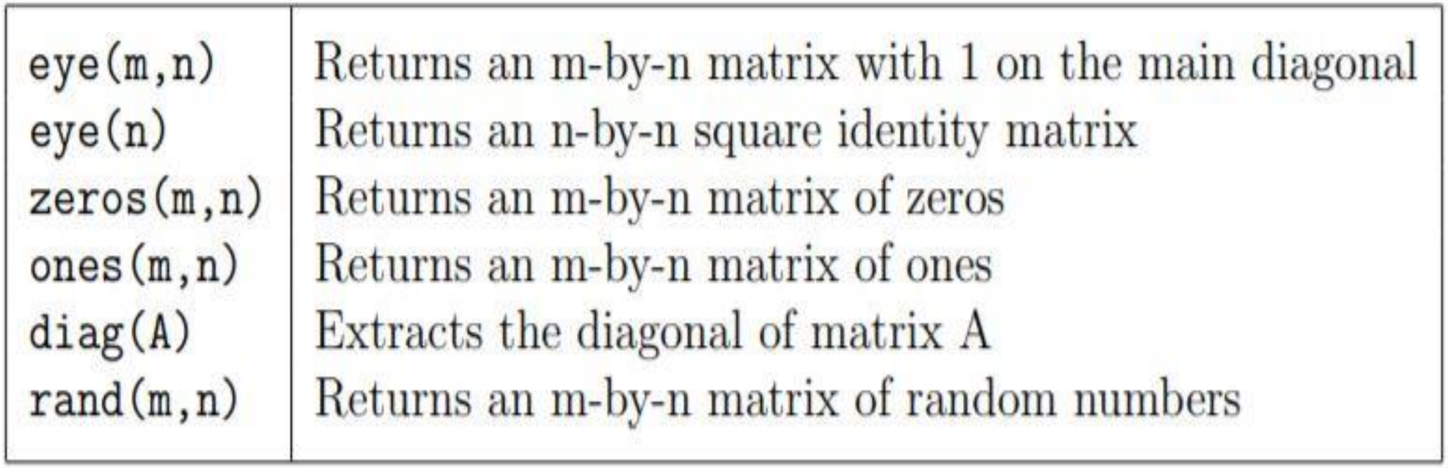


```
B = [4/5      7.23*tan(x)      sqrt(6); ...
      1/x^2   0                3/(x*log(x)); ...
      x-7     sqrt(3)          x*sin(x)];
```

Matrix generators

- The matrix of zeros, the matrix of ones, and the identity matrix are returned by the functions zeros, ones, and eye, respectively.

Table 2.4: Elementary matrices



`eye(m,n)`	Returns an m-by-n matrix with 1 on the main diagonal
`eye(n)`	Returns an n-by-n square identity matrix
`zeros(m,n)`	Returns an m-by-n matrix of zeros
`ones(m,n)`	Returns an m-by-n matrix of ones
`diag(A)`	Extracts the diagonal of matrix A
`rand(m,n)`	Returns an m-by-n matrix of random numbers

```
>> b=ones(3,1)
 b = 1
     1
     1
```

- >> eye(3)
- >> c=zeros(2,3)

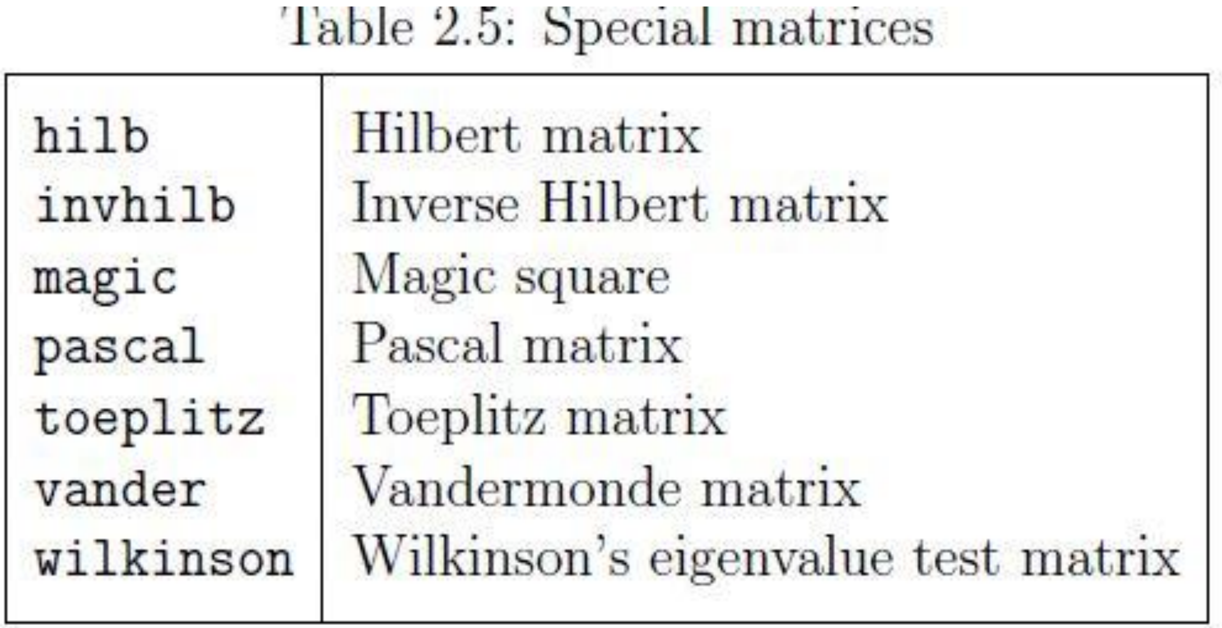
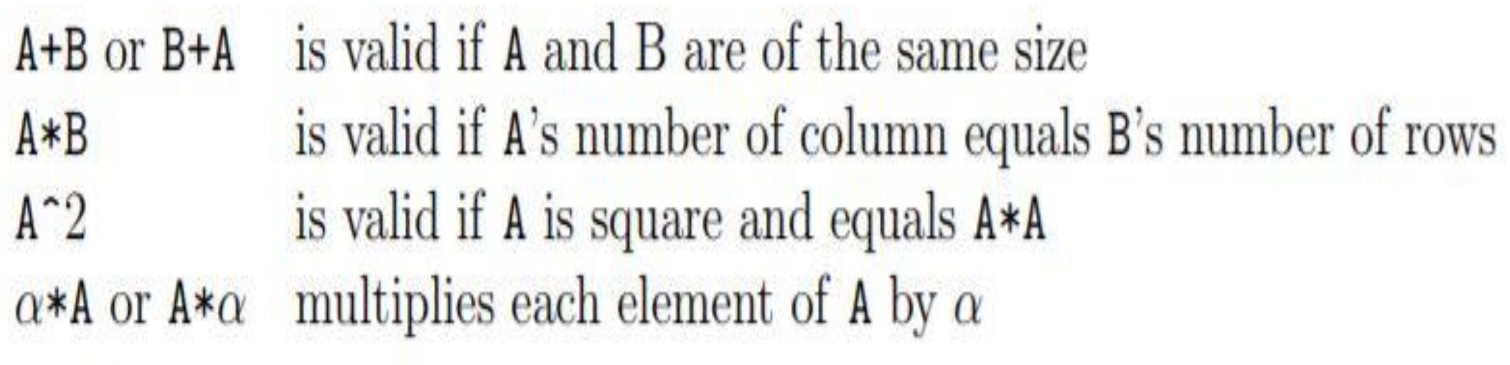


Table 2.5: Special matrices

`hilb`	Hilbert matrix
`invhilb`	Inverse Hilbert matrix
`magic`	Magic square
`pascal`	Pascal matrix
`toeplitz`	Toeplitz matrix
`vander`	Vandermonde matrix
`wilkinson`	Wilkinson's eigenvalue test matrix

Matrix arithmetic operations

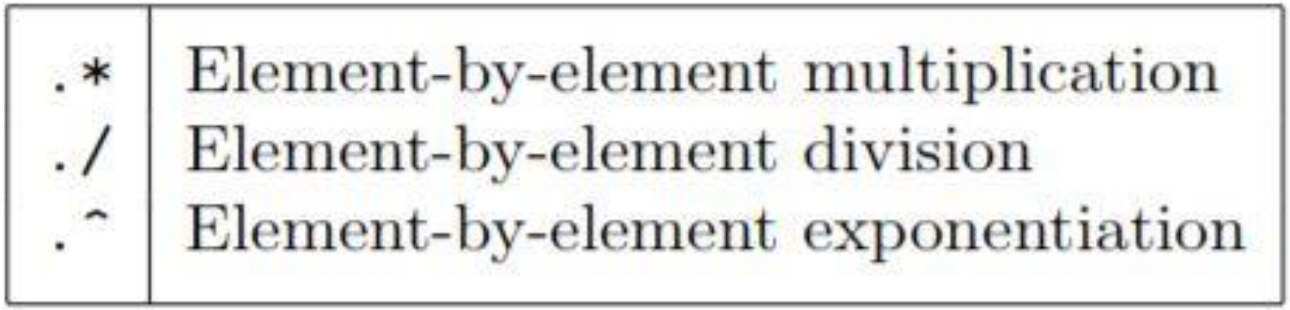
- As we mentioned earlier, MATLAB allows arithmetic operations: +, −, ∗, and ˆ to be carried out on matrices.
-



A+B or B+A	is valid if A and B are of the same size
A*B	is valid if A's number of column equals B's number of rows
A^2	is valid if A is square and equals A*A
α*A or A*α	multiplies each element of A by α

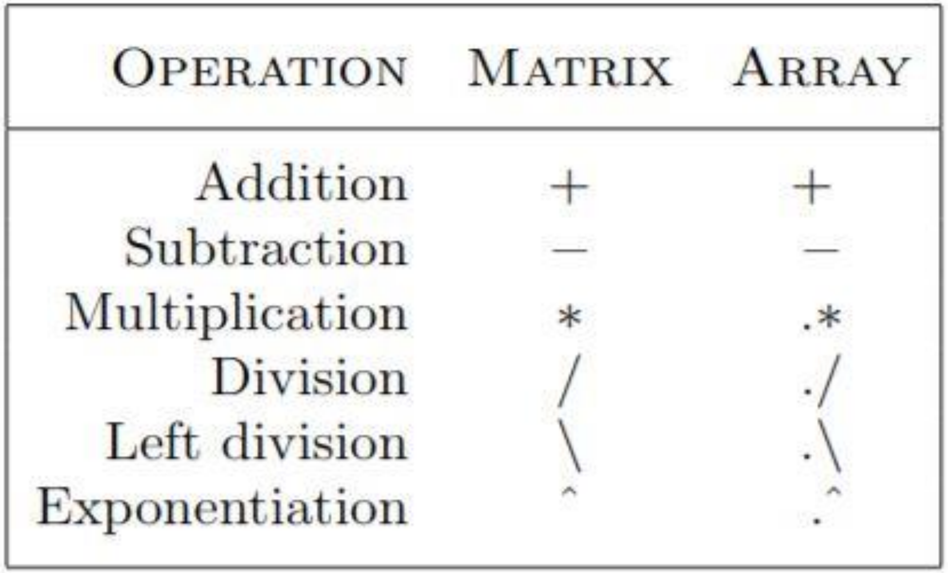
Array arithmetic operations

- On the other hand, array arithmetic operations or array operations for short, are done element-by-element.
- The period character, ., distinguishes the array operations from the matrix operations.
- However, since the matrix and array operations are the same for addition (+) and subtraction (−), the character pairs (.+) and (.−) are not used.



.*	Element-by-element multiplication
./	Element-by-element division
.^	Element-by-element exponentiation

- >> C = A.*B
- >> A.^2



OPERATION	MATRIX	ARRAY
Addition	+	+
Subtraction	−	−
Multiplication	*	.*
Division	/	./
Left division	\	.\
Exponentiation	^	.^

vectors U and V with elements $U = [u_i]$ and $V = [v_j]$.

$$\begin{array}{rll} U.*V & \text{produces} & [u_1v_1\ u_2v_2 \ldots\ u_nv_n] \\ U./V & \text{produces} & [u_1/v_1\ u_2/v_2 \ldots\ u_n/v_n] \\ U.\hat{}V & \text{produces} & [u_1^{v_1}\ u_2^{v_2} \ldots\ u_n^{v_n}] \end{array}$$

Solving linear equations

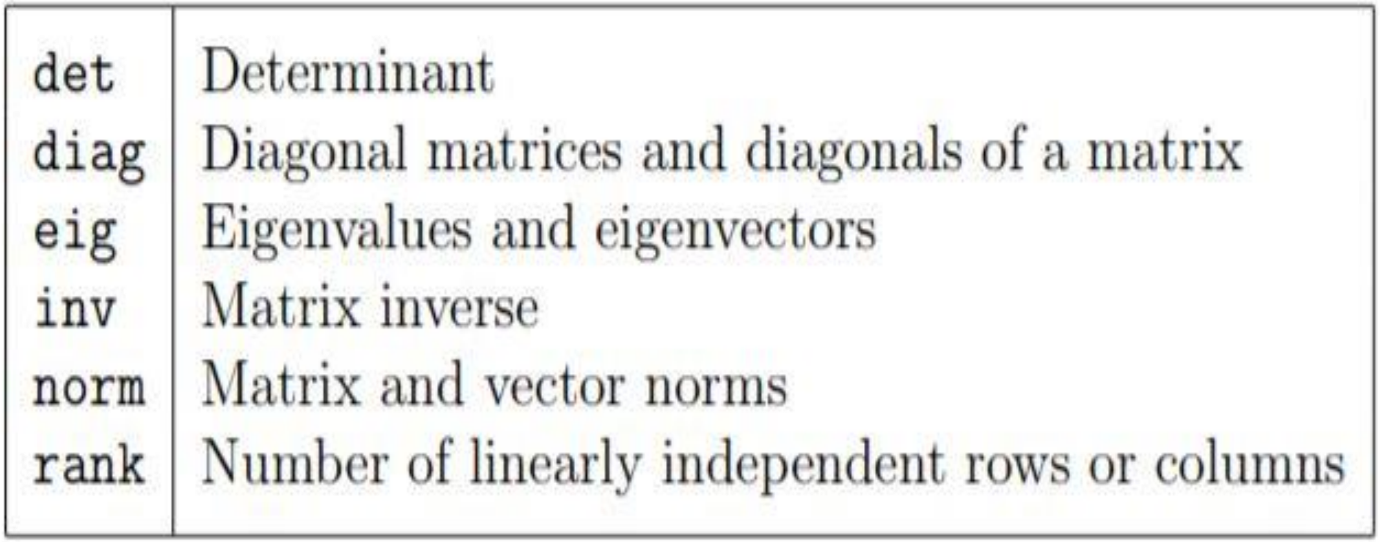
- With matrix notation, a system of simultaneous linear equations is written
- Ax = b
- This equation can be solved for x using linear algebra.
- The result is x = A^{-1} b. There are typically two ways to solve for x in MATLAB:
- 1. The first one is to use the matrix inverse, inv.

```
>> A = [1 2 3; 4 5 6; 7 8 0];
>> b = [1; 1; 1];
 >> x = inv(A)*b
```

- The second one is to use the backslash (\)operator.

```
>> A = [1 2 3; 4 5 6; 7 8 0];
 >> b = [1; 1; 1];
>> x = A\b
```

Matrix functions



det	Determinant
diag	Diagonal matrices and diagonals of a matrix
eig	Eigenvalues and eigenvectors
inv	Matrix inverse
norm	Matrix and vector norms
rank	Number of linearly independent rows or columns