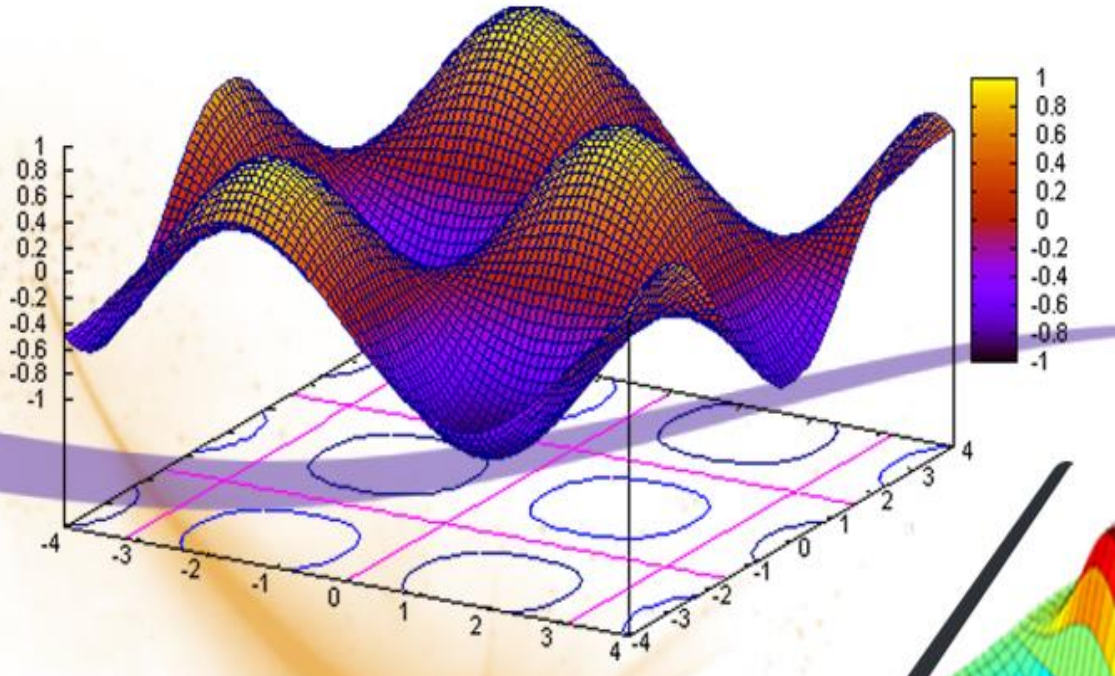
MATLAB FOR ALL ENGINEERS

MATLAB Program for Engineering Implementations Examples and Programming Layout



By

Saeed J. Almalowi

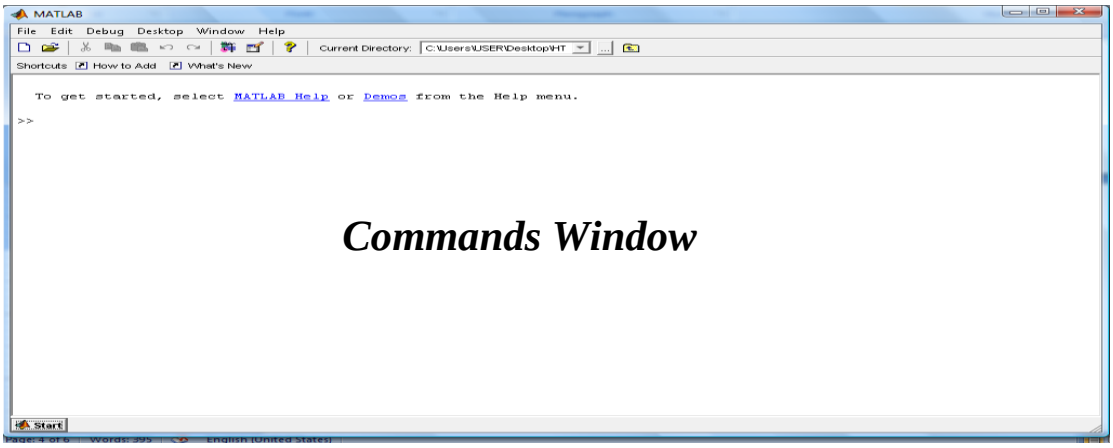
CHAPTER#1 MATLAB Windows

1.1 Starting Windows

Two types of windows which are important to be known by users: command window and editor window/ or M-file window (script).

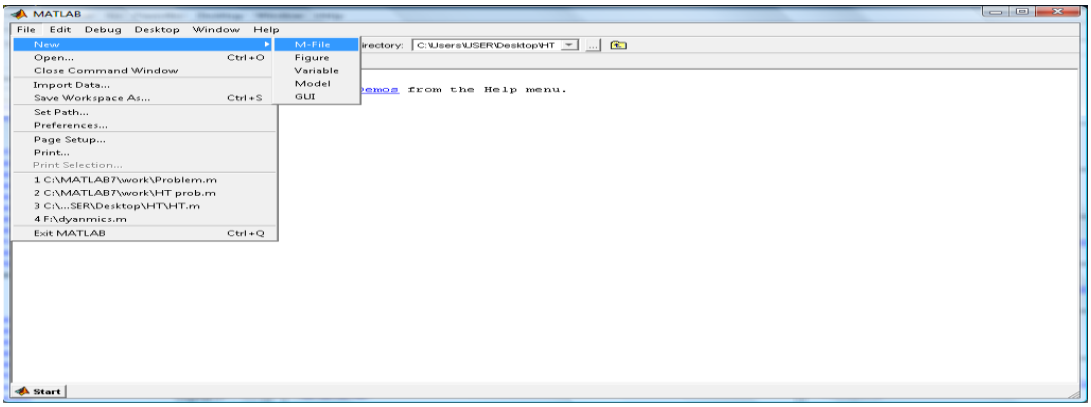
1.2. Commands Window

It allows users to correct their program's errors and helps them to figure out which types of errors programmers might have in their programs.

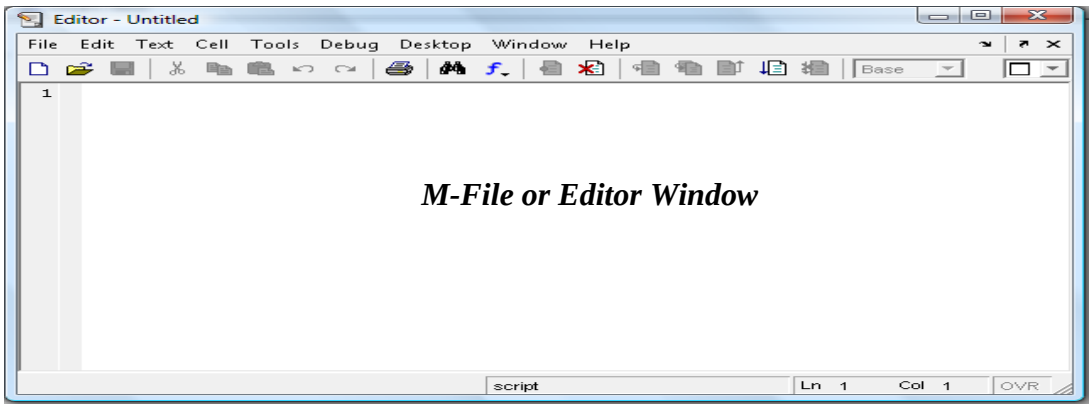


1.3. Editor Windows or M-File Windows

This window is used to build up a program. It is called M-File or script file .A programmer should be "save as" his file before starting



Save as an M-file and see this window



After opening MATLAB window, you might start your program or writing your commands.

2.1. A mathematical and ***MATLAb*** Functions

MATLAB OBJECTIVE

MATLAB program software is used to
- solve algebraic equations,
- plot mathematical functions,
- solve differential equations,
- compute the integrals, and
- Generate numerical solution of ODEs..Etc.

A MATHEMATICAL ***MATLAB*** Logarithms and Functions

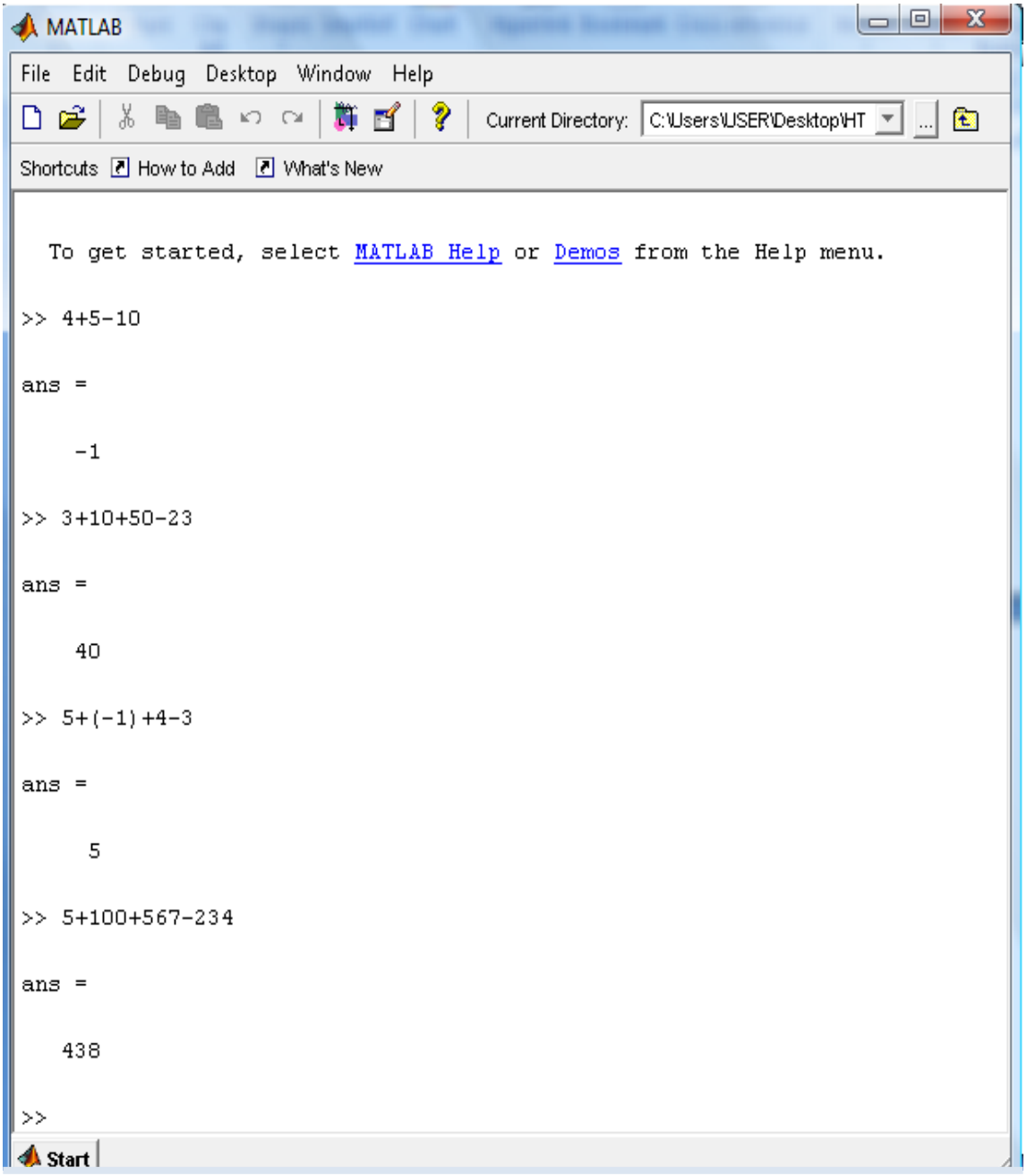
Mathematic Function Form	MATLAB Form
Square root $\sqrt{x}$	sqrt(x)
Exponential $e(x)$	exp(x)
logarithmic $\ln(x)$	log(x)
Logarithmic tenth $\log_{10}(x)$	Log10(x)
Sine angle ($\sin(\theta)$)	$\sin(\theta)$
Cosine angle (cos(ɵ))	$\cos(\theta)$
Pie π	pi
Tan inverse angle $\tan^{-1}(\theta)$	$a\tan(\theta)$
Subtract -	-
Adding +	+
Power	^
Division ÷	/
Multiple	*

MATLAB logic operators will be discussed in the following chapter.

Example (1): Add the following mathematical calculations

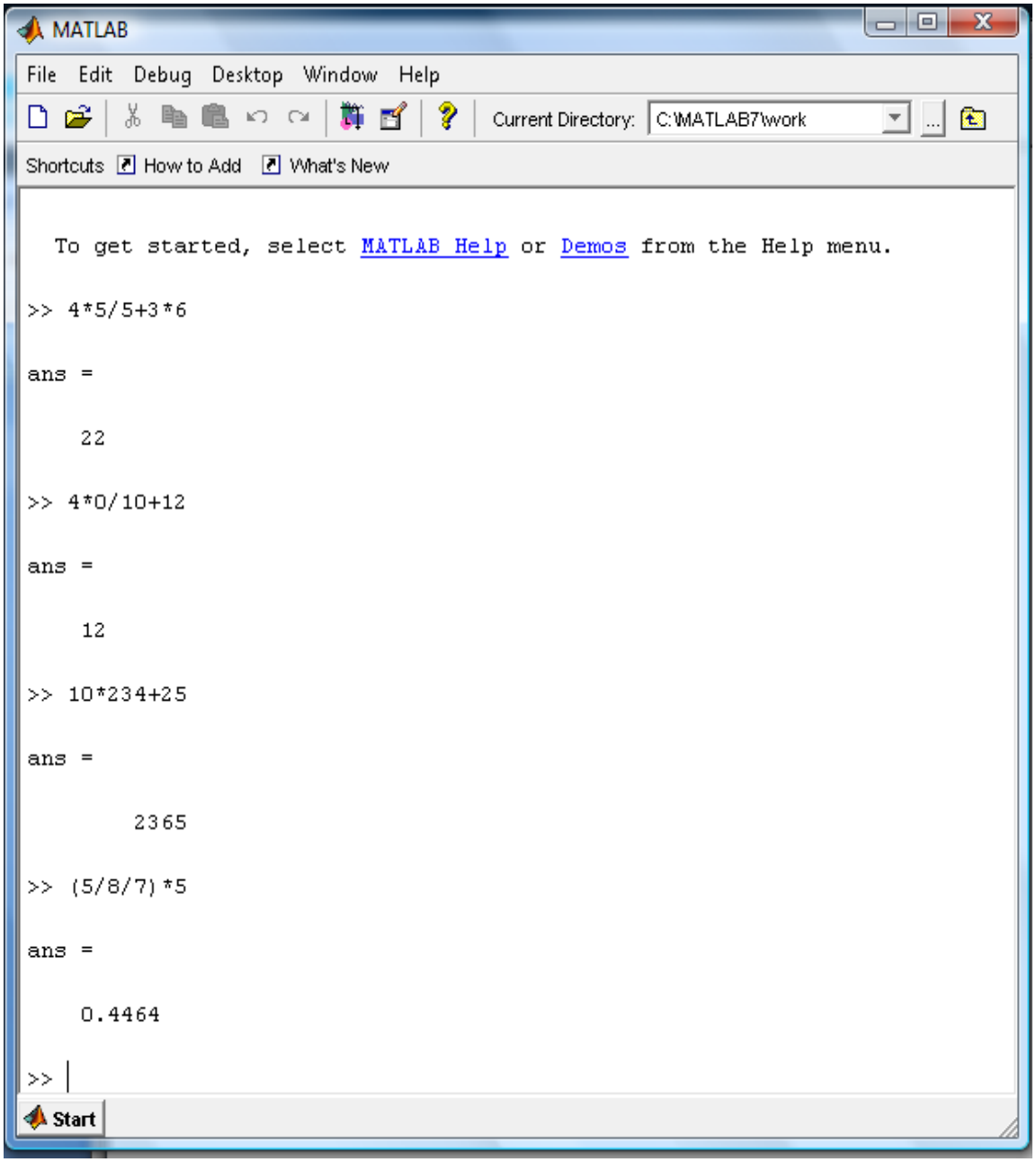
a- 4+5-10
b- 3+10+50-23
c- 5+ (-1)+4-3
d- 5+100+567-234

Ans.



Example (2): Estimate the following mathematical calculations

a- 4×5÷5+3×6
b- 4×0÷10+12
c- 10×234+25
d-(5÷8÷7)×5



Example (3): Estimate the following calculations

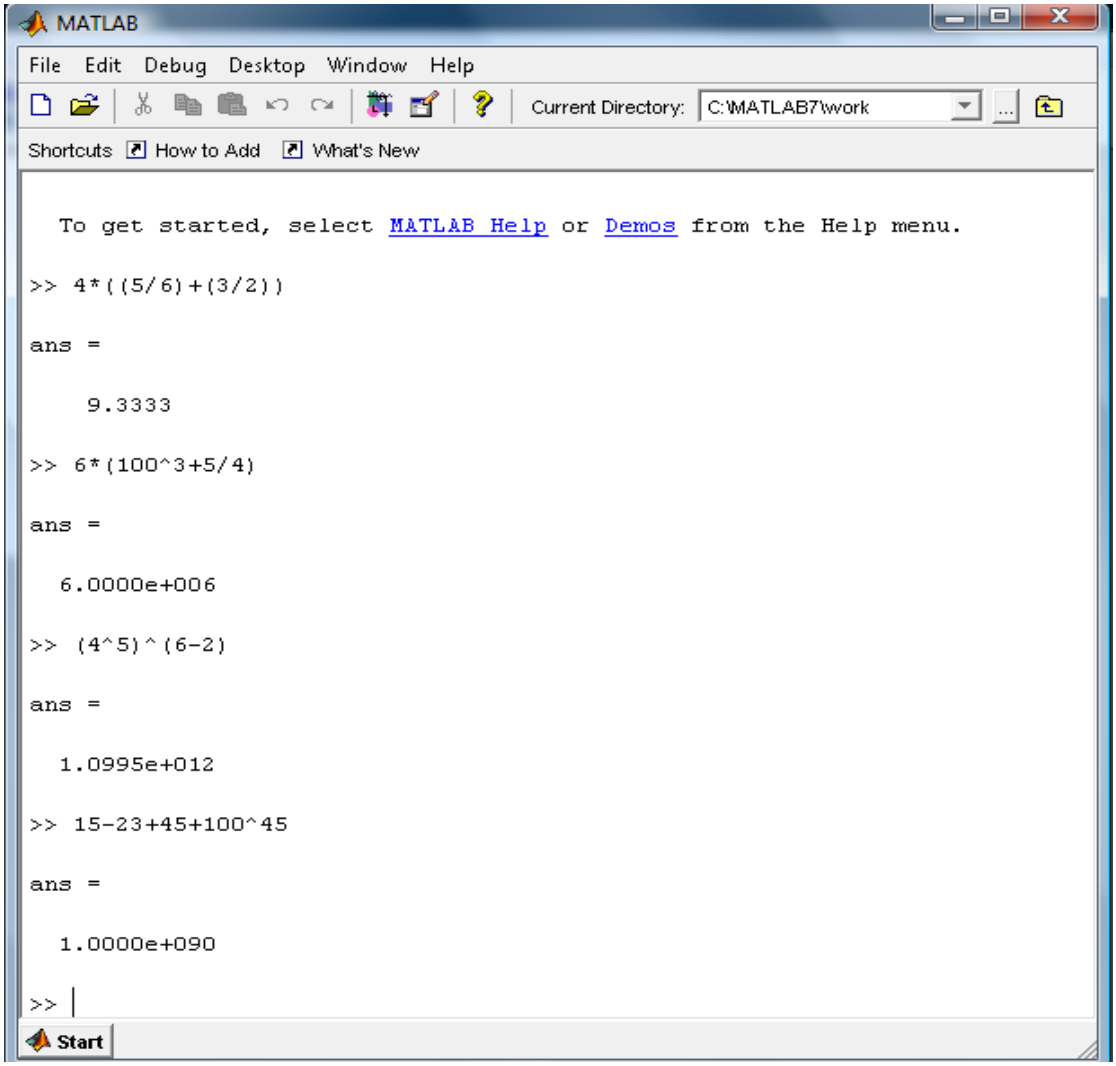
a- $4\times((5\div6)+(3\div2))$

b- $6\times(100^3+5\div4)$

c- $(4^5)^{6-2}$

d- $15-23+45+100^{45}$

Ans.



By hand

$4\times((5\div6)+(3\div2))$

First of all, this operation is calculated from inside the parenthesis from left to right. Then multiply by the outside number.

Step #: ($5\div6$) =(0.833)

Step#2: $(3\div2)$ = (1.5)

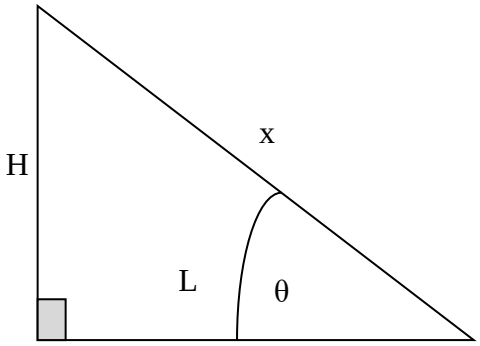
Step#3: (Step#1+Step#2) = (0.833+1.5)=(2.333)

Step#4: $4\times$(Step#3) = $4\times(2.333)$ = 9.333

2.4. Basic Triangle Functions

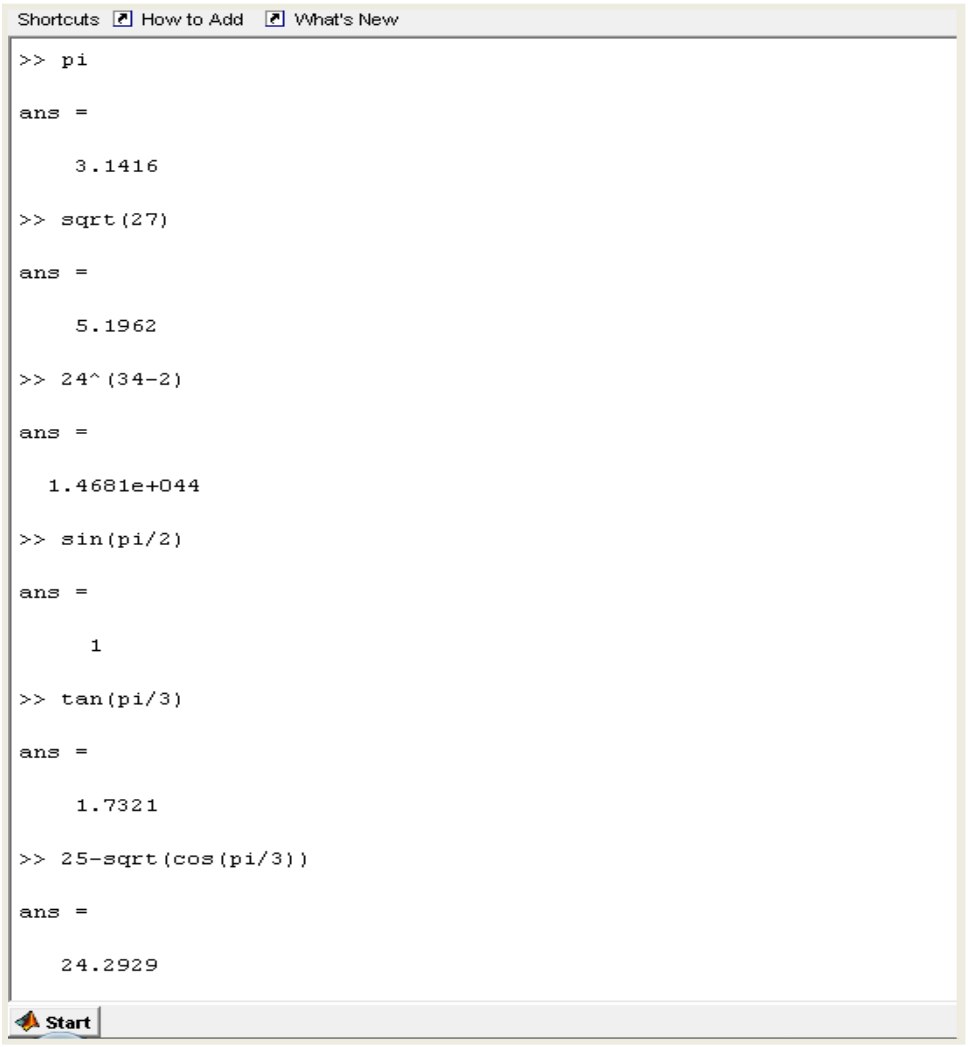
Right hand triangle

$$sin\theta = \frac{H}{x} \qquad cos\theta = \frac{L}{x} \qquad tan\theta = \frac{H}{L}$$



Example (4): Estimate the basic triangle expressions and the other mathematical expressions

a- π
b- $\sqrt{27}$
c- $24^{(34-2)}$
d- sin(90)
e- tan(45)
f- $25-\sqrt{\cos(60)}$
Ans.



2.5. Vectors and Matrices

The vectors are a part of matrices. They are structured either in rows or columns. Therefore, the matrices are a group of rows and columns of vectors. Showing rows and vectors by using MATLAB have different commands, so in this section will be learning the matrices structures.

Example (5): Create the column vector with three elements

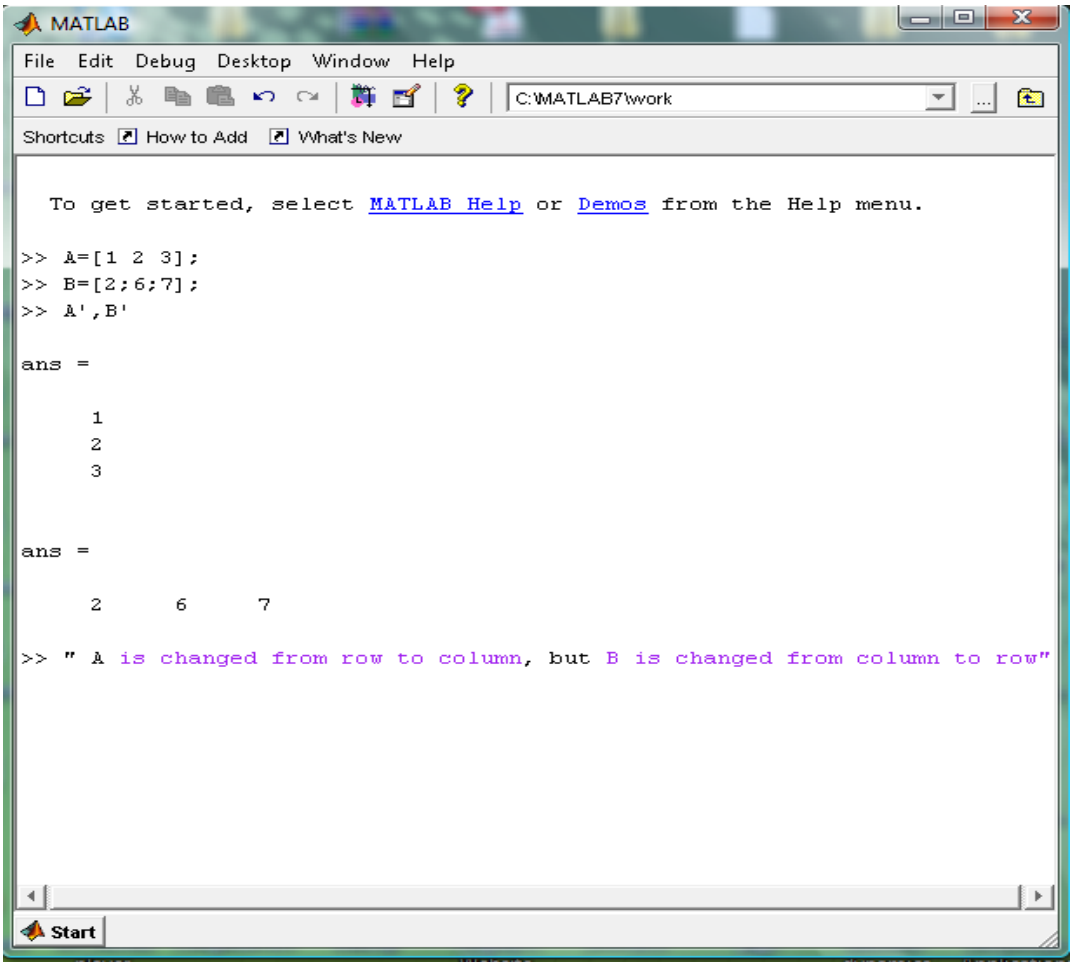
Ans.

$$A=\begin{bmatrix}1\\2\\3\end{bmatrix}$$

>>A=[1; 2 ; 3]
ans=
1
2
3

Example (6): Create the row vector with three elements

Ans.
B=[1 2 3]
>>B=[1 , 2, 3]
ans=
1 2 3



2.5.1. Transpose the Vector

The column vectors can be turned into the row vectors by using " A′" . As we know a vector which has elements in a row will be in the horizontal, but a vector which has elements in column will be in the vertical.

A is the vector in one column and three rows. $A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$

B is the Vector in one row and three columns. $B = [1 \ 3 \ 4]$

Therefore, matrices have number of rows and columns.

Square Matrices (SM)

C is the square matrix because it has three rows and three columns.

$$C = \begin{bmatrix} 1 & 3 & 4 \\ 5 & 6 & 8 \\ 10 & 12 & 14 \end{bmatrix}$$

The reverse of this matrix will be " C′"

A square matrix means that the numbers of rows are equal to the numbers of columns.

Example (7): Find the reverse vector for the following vectors

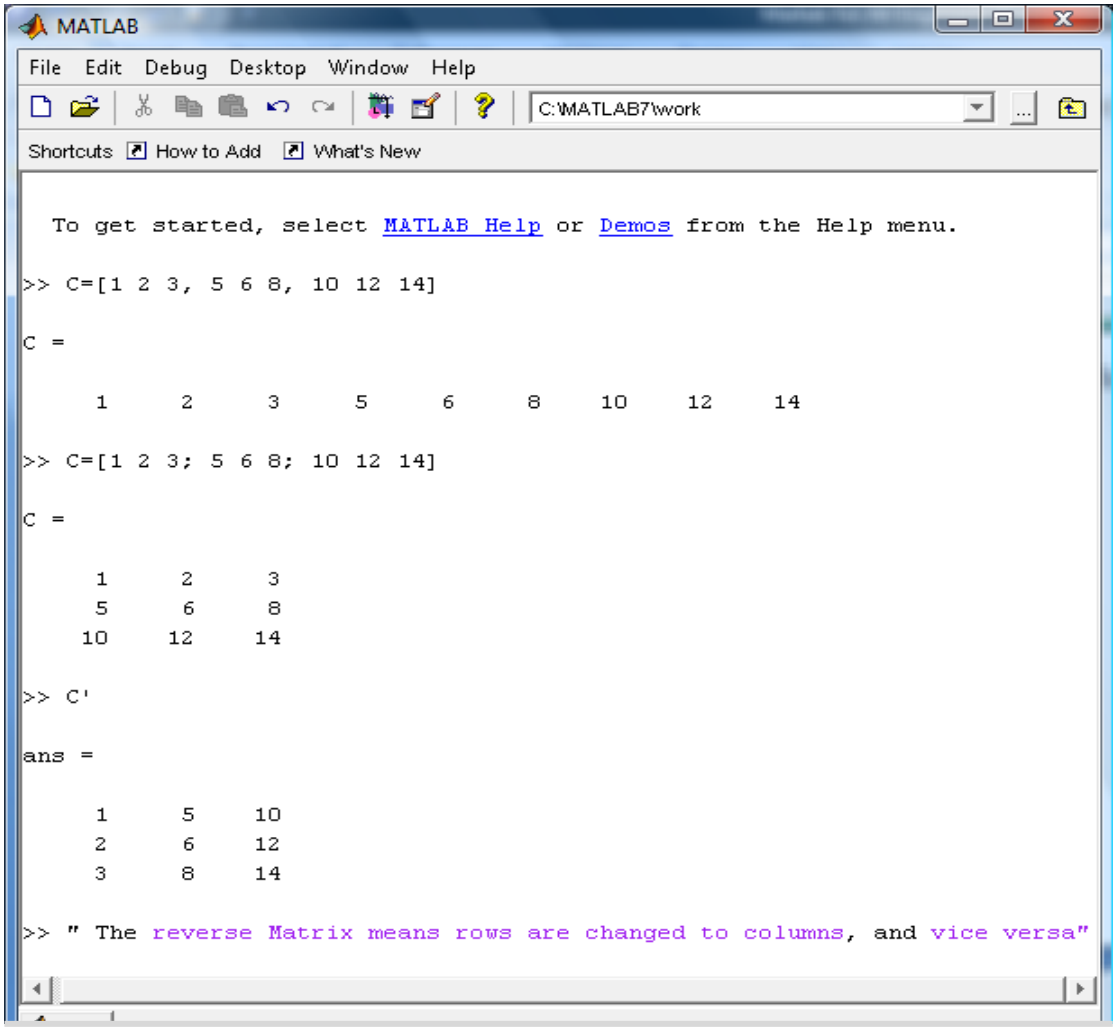
$A = [1 \;\; 2 \;\; 3], B = \begin{bmatrix} 2 \\ 6 \\ 7 \end{bmatrix}$

Ans.

Example (8): create the following matrix, and then find the reverse of the matrix, too.

$$C = \begin{bmatrix} 1 & 3 & 4 \\ 5 & 6 & 8 \\ 10 & 12 & 14 \end{bmatrix}$$

Ans.



Notice: see in this window the colon makes the matrix in row, but the semi- colon makes the matrix in rows and columns as it is required in the example(8).

Rectangular Matrices (RM)

RM is the number of columns more than the numbers of rows or vice versa.

$$\begin{bmatrix} 1 & 2 & 3 & 5 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 8 & 6 \end{bmatrix}$$ This is called the rectangular matrix.

Diagonal Matrices (DM)

A first element of the first row is 1, the second element of the second row is 1, and the third element of the third row is 1 for SM.

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

Unit Matrices (UM)

All elements are one.

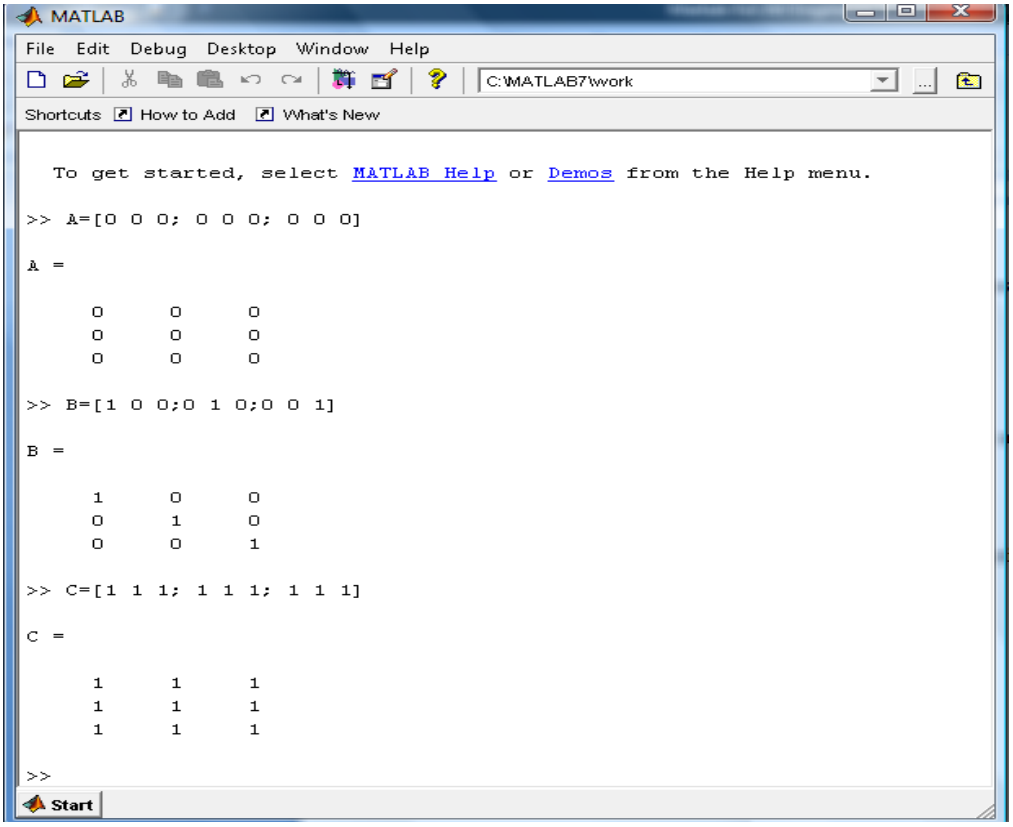
$$\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

Zero Matrices (ZM)

All the elements of this type of matrices are zeros.

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Examples (9) create the zero, diagonal, and unit matrices by using MATLAB command window.



2.6 Vectors Rules and Adding of Matrices, Subtracting of Matrices, and multiplication of Matrices

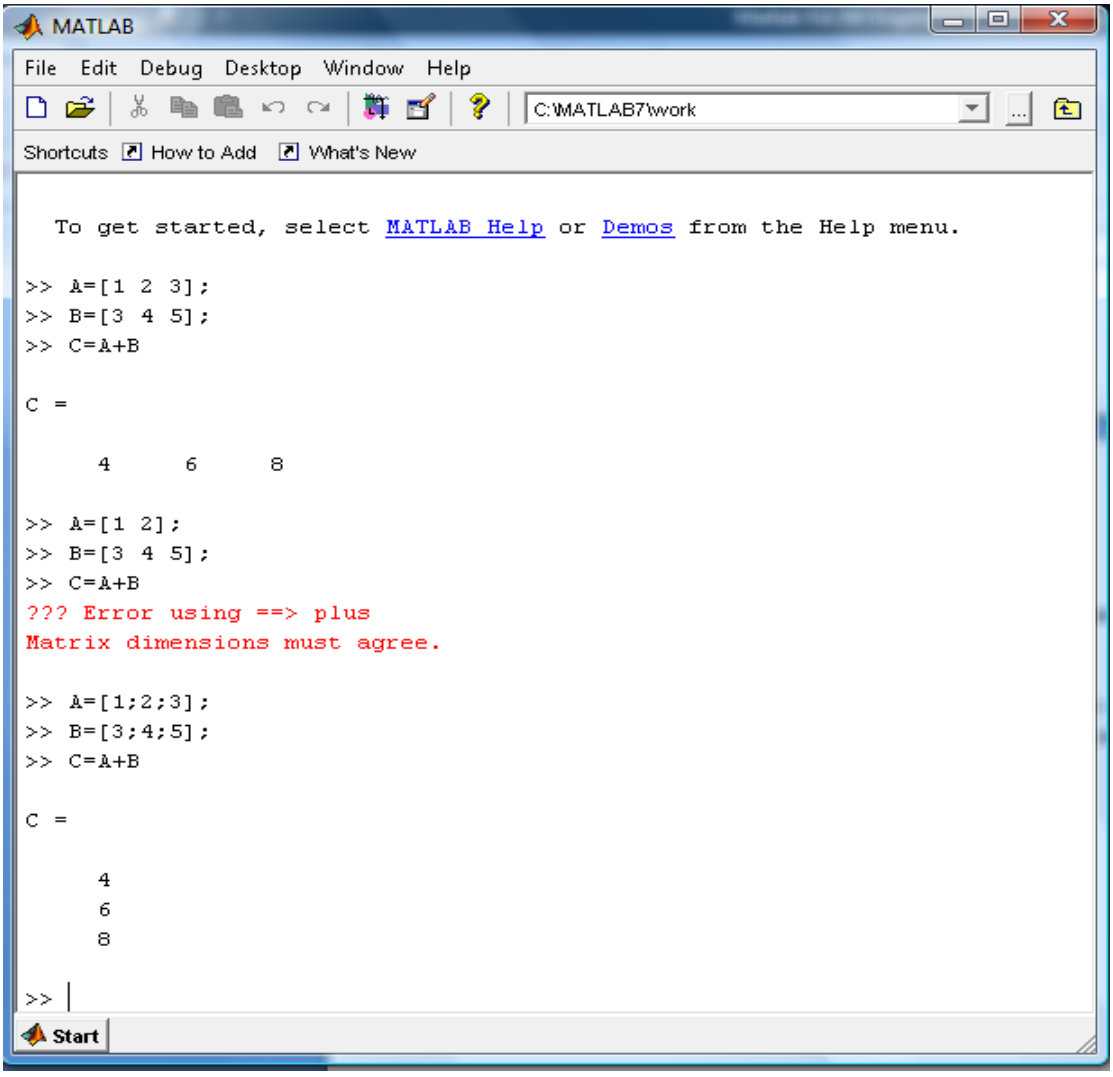
2.6.1. Adding and subtracting vectors

In order to add vectors or subtract vectors, they must have must same of rows and columns. We cannot add a vector which has two rows with a vector which has three rows.

Example (10): Add the following vectors
1. A= [1 2 3], B= [3 4 5], C=A+B
2. A= [1 2], B=[3 4 5], C=A+B

3.

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = A + B$$



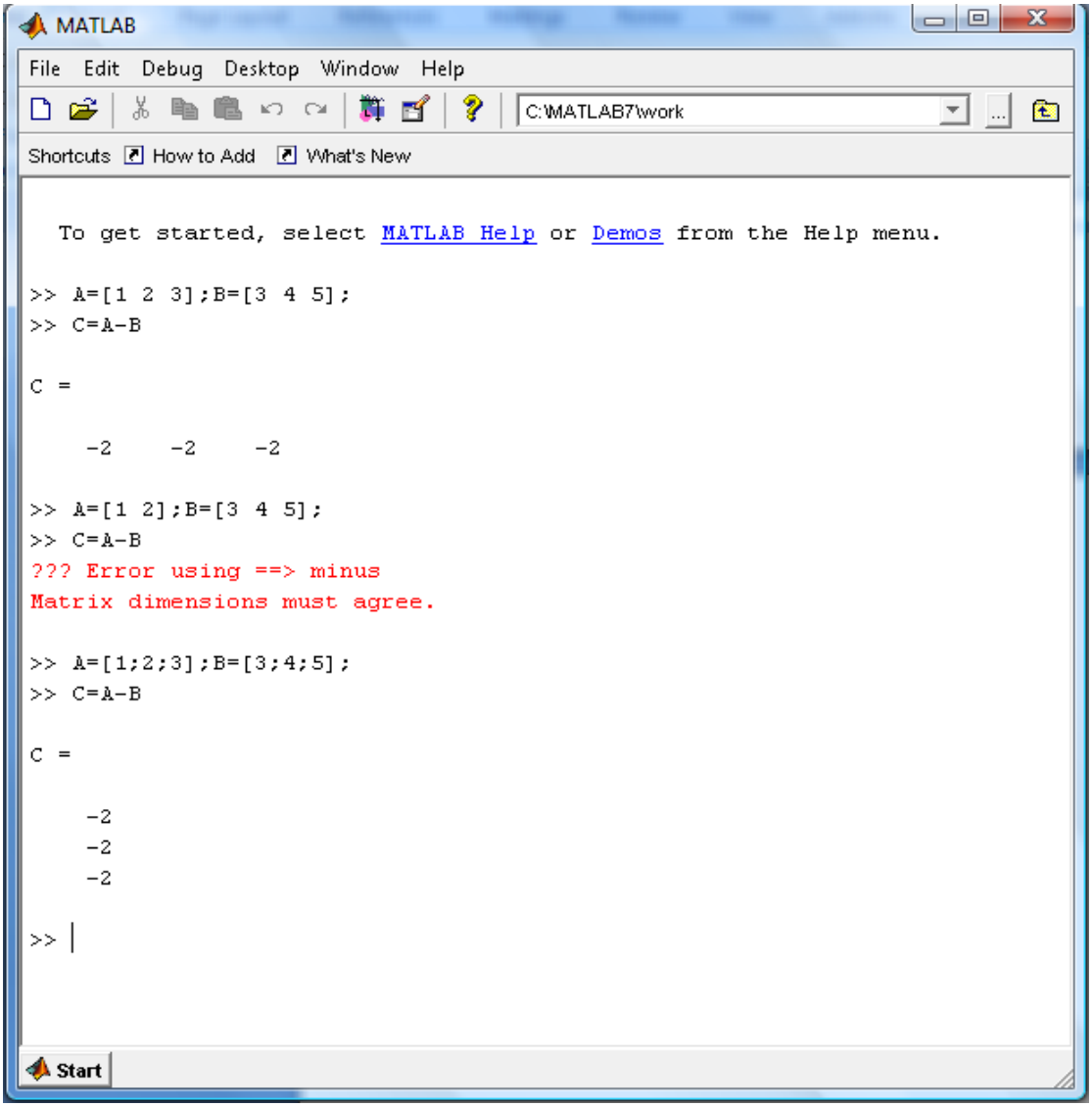
***Example (11)*:** Subtract the following vectors

1. A= [1 2 3], B= [3 4 5], C=A-B
2. A= [1 2], B=[3 4 5], C=A-B

3.

$$A = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}, B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = A - B$$

Ans.



2.6.2 Multiplication and division vectors

N is the number of rows, and m is the number of columns.
3×2 means N=3 and m=2; therefore, the N×m can be multiplied in each other's if N in the first vector is equal to m in the second vector.

Explanation:

$$A = [1\ 2\ \ 3], B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, N_A = m_B \ or \ N_B = m_A$$

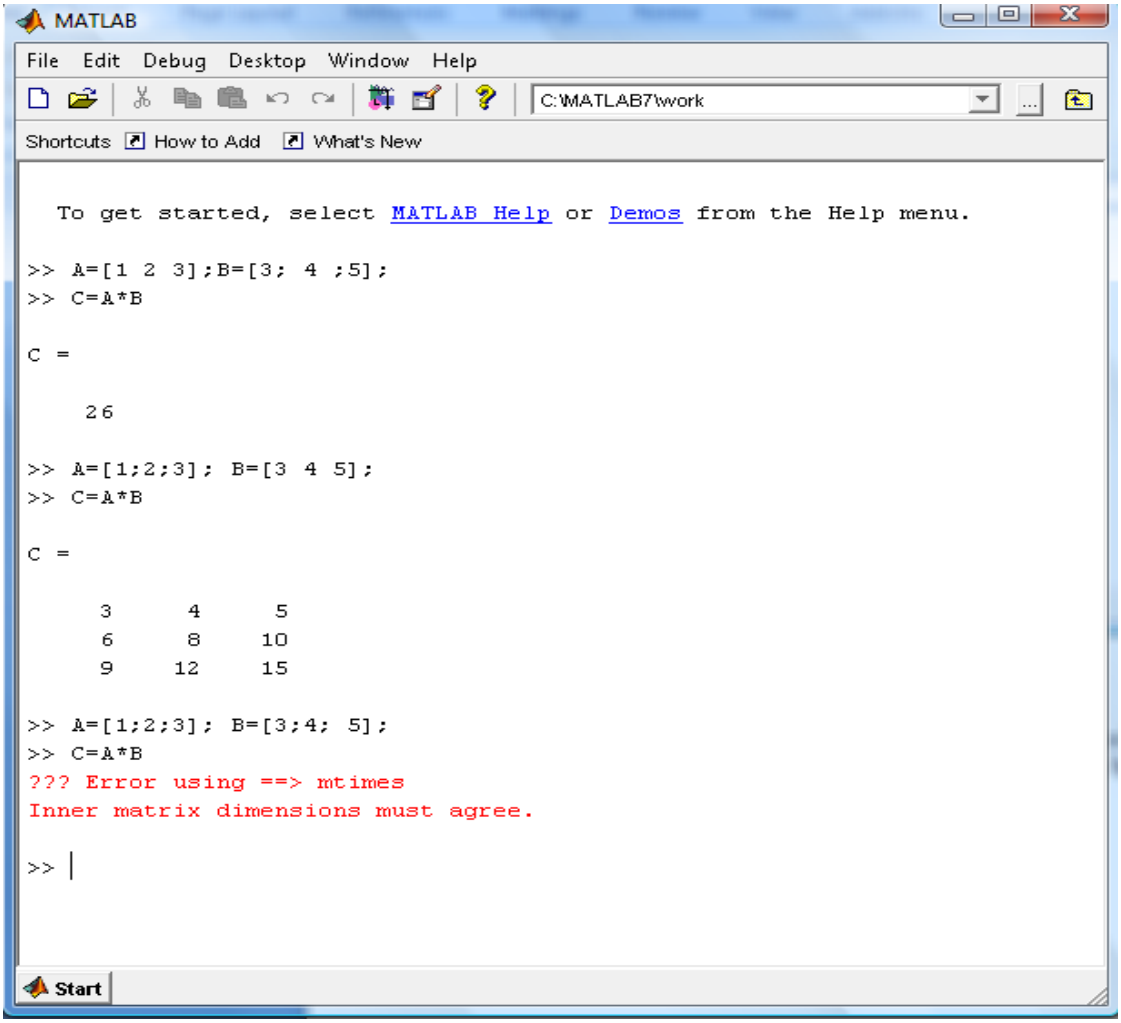
In the first vector N×m= (1×3), but in the second vector is opposite N×m=(3×1)
Consequently, $N_A = m_B$ and vice versa. (Rule#2)

Notice:
A one row multiplied by a one column, the result will be a one value.
A one column multiplied by a one row, the result will be a matrix.

Example (12): Find the multiplication for the following vectors.

$$A = [1\ 2\ \ 3], B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = BxA$$

$$A = [1\ 2\ \ 3], B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = AxB$$



The rules of the division are not the same as the rules of the multiplications.
$N_A=N_B$ and $m_A=m_B$

There is another type of commands which can be used to flip or reshape a matrix.

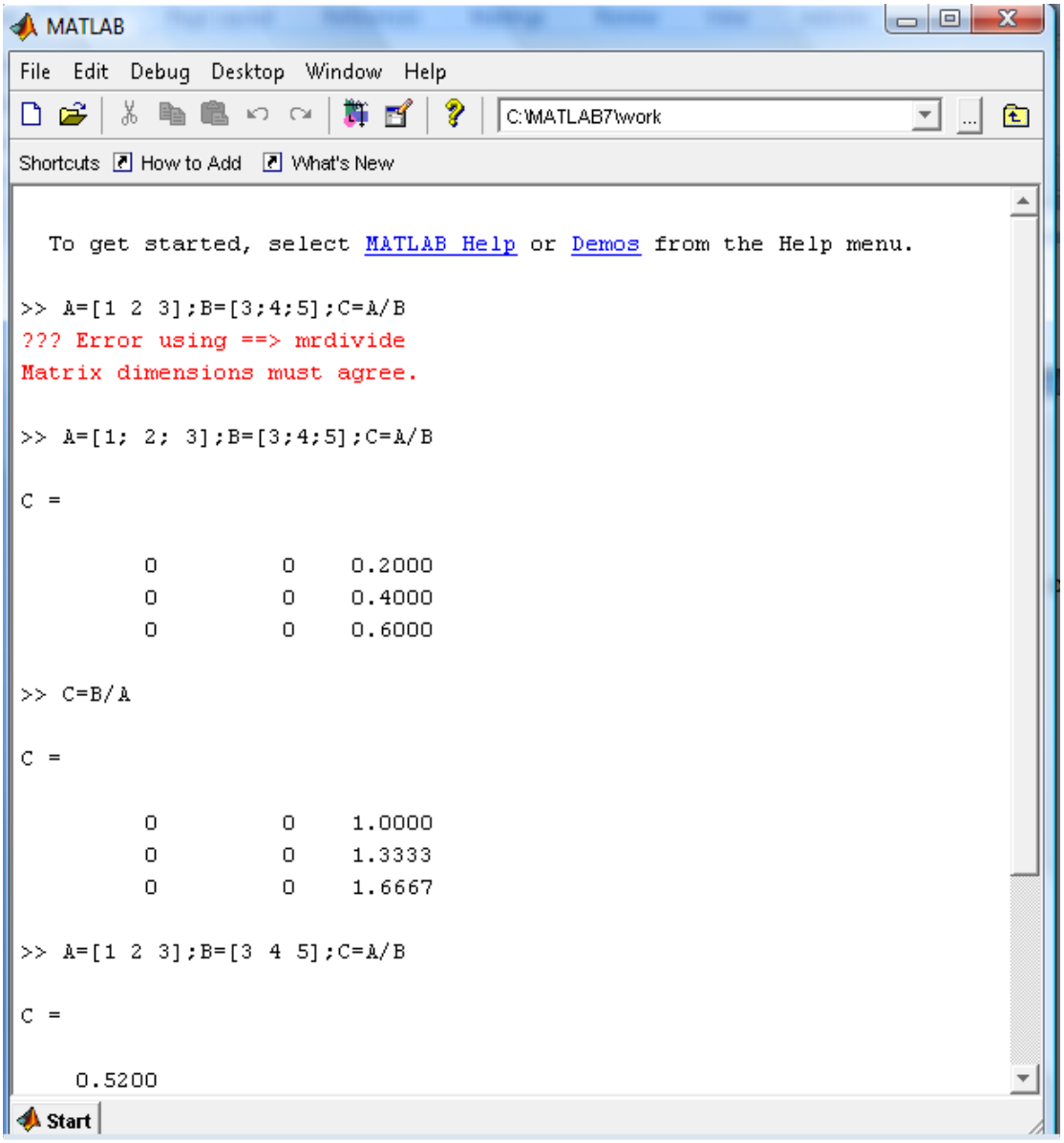
MATLAB commands	Meaning
C=reshape(A,N,m)	To returns the m-by-n matrix C whose elements are taken column-wise from A
flipud(A)	To flip a matrix from up to bottom.
fliprl(A)	To flip a matrix from right to left.
fliplr(A)	To flip a matrix from left to right.

Example (13): Divide the following vectors:

$$A = [1\ 2\ \ 3], B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = A/B$$

$$A = [1\ 2\ \ 3], B = \begin{bmatrix} 3 \\ 4 \\ 5 \end{bmatrix}, C = B/A$$

Ans.



2.6.3. The matrices Rules

Matrices have different rules which are based on type of matrix and type of the mathematical operation (+, *,- , /).

2.6.3.1. Adding and subtracting matrices

The two matrices must be the same type (SM-SM, SM+SM, RM-RM, RM+RM….etc)

2.6.3.1 Multiplication and division of matrices

The multiplication of matrices has different rules. We can multiply a row vector in a matrix, but we cannot multiply a matrix in a row vector

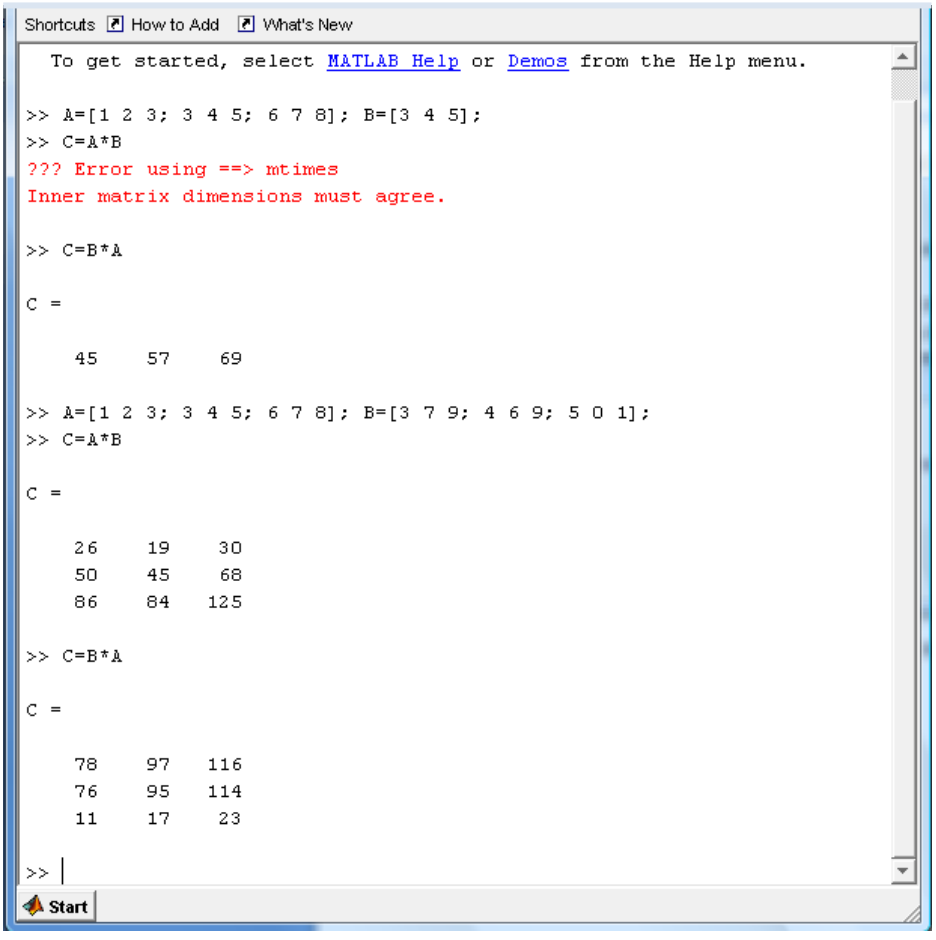
We can multiply SM in SM, and we cannot multiply RM in SM, we cannot multiply SM in RM, as well.

Examples (14): find the following expressions

$$A=\begin{bmatrix}1&2&3\\3&4&5\\6&7&8\end{bmatrix}, B=\begin{bmatrix}3\\4\\5\end{bmatrix}, C=A\times B, C=B\times A$$

$$A=\begin{bmatrix}1&2&3\\3&4&5\\6&7&8\end{bmatrix}, B=\begin{bmatrix}3&7&9\\4&6&9\\5&0&1\end{bmatrix}, C=A\times B, C=B\times A$$

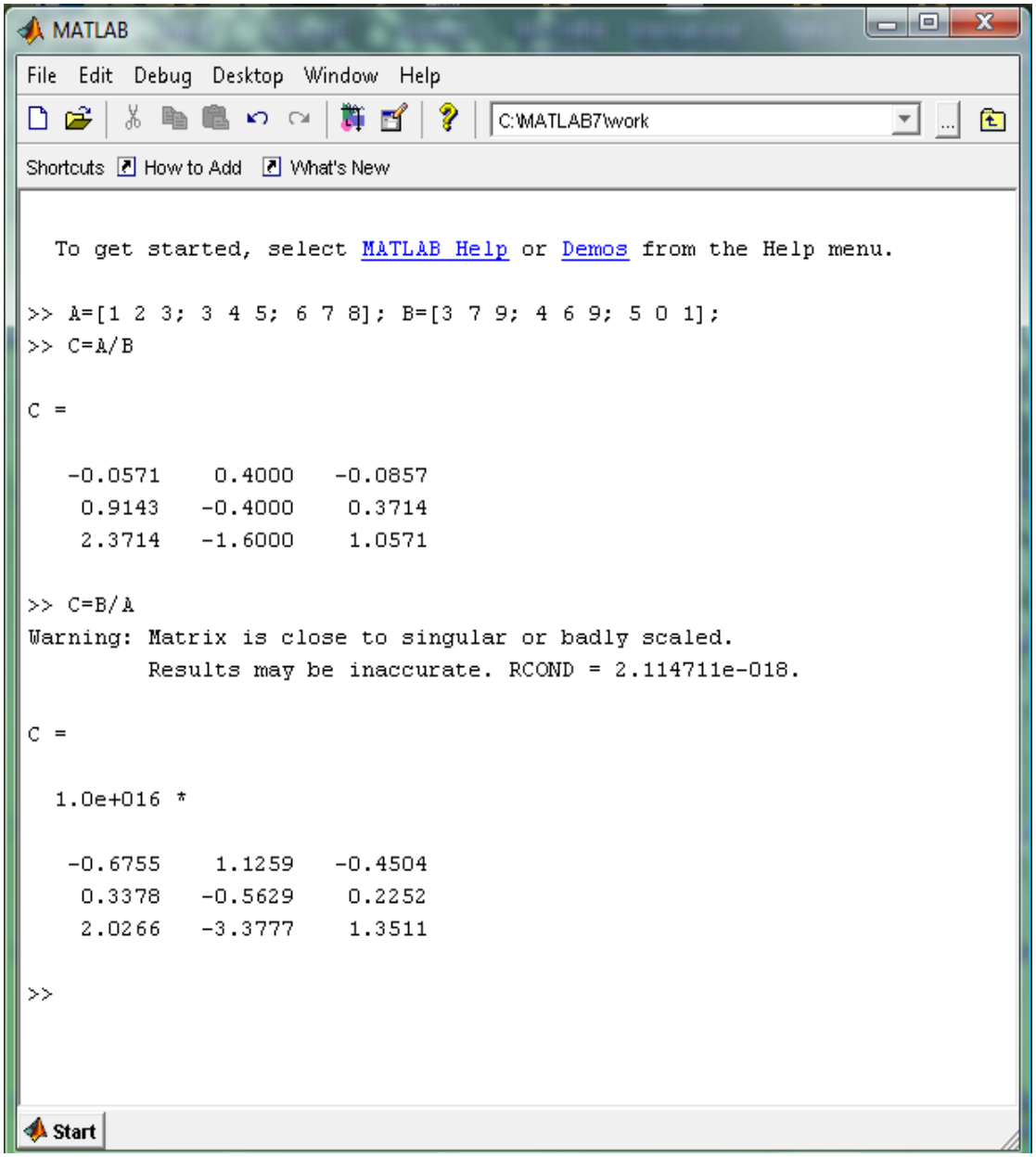
Ans.



Division of Two Matrices

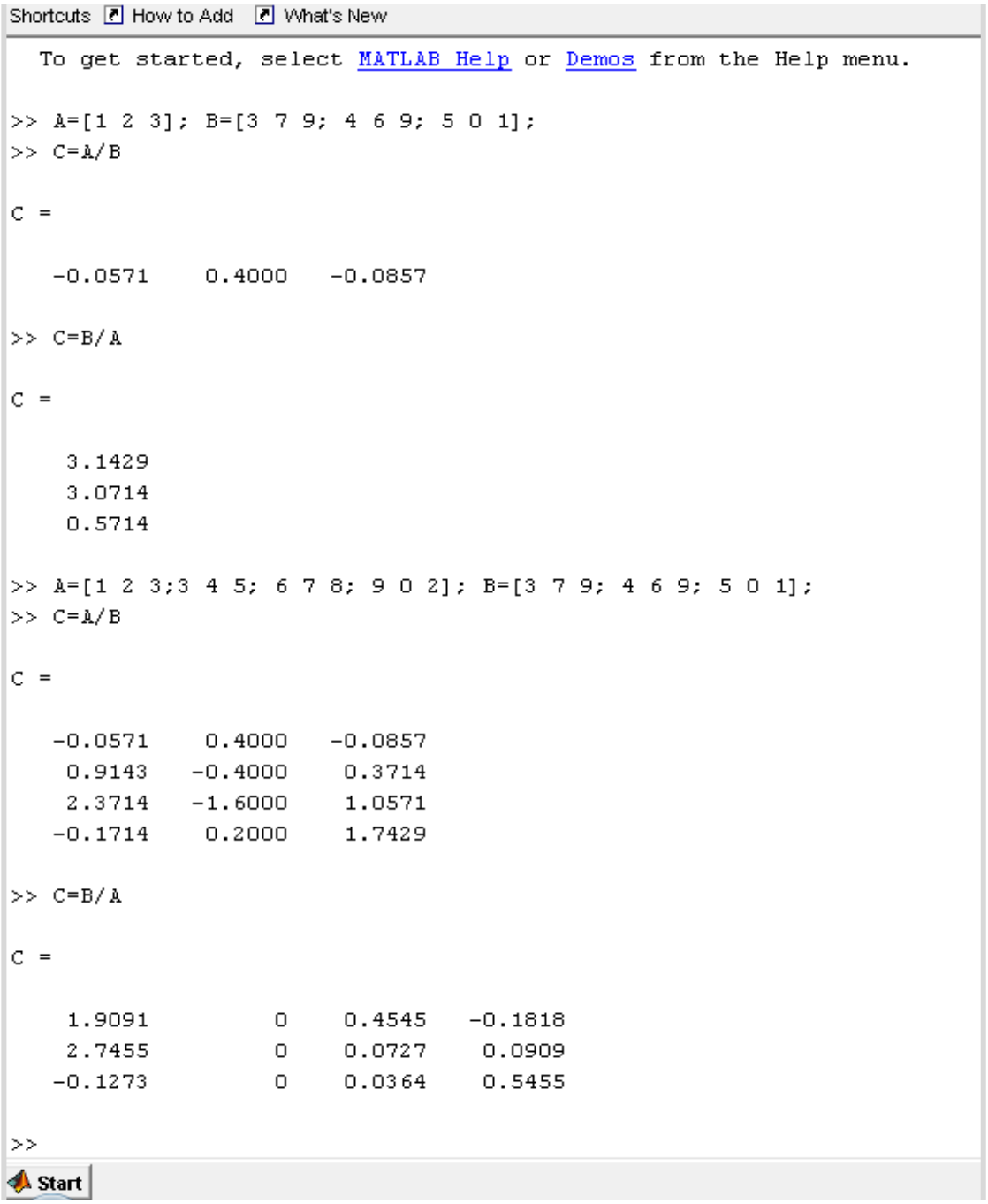
We can divide two matrices under certain rules. We can divide the same type of matrices, for instance, RM/RM, SM/SM….etc.

Example (15):



We can divide a vector on a matrix and vice versa. Further, we can divide RM on SM and vice versa. All those rules should be considered by a programmer before he/ she creates his/her program. A professional programmer must know all of those rules to design any type of program. MATLAB helps a programmer to figure out his mistakes. The mistakes will show up in the command window.

Tracing any column or row in a matrix A
A (1,:) takes the first column in a matrix A.
A(n,:) takes “n” values of a ”n” row.
A(:,n) takes ” n” values of a “ n” column.



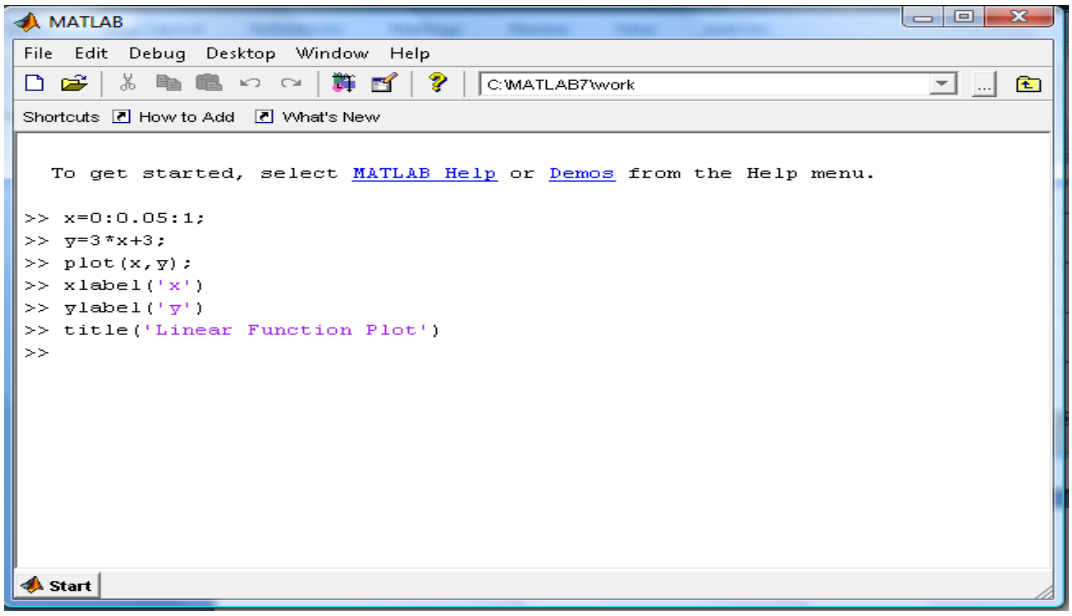
Notice: We can divide one element on any type of matrix. Also, we can multiply one element by a matrix.

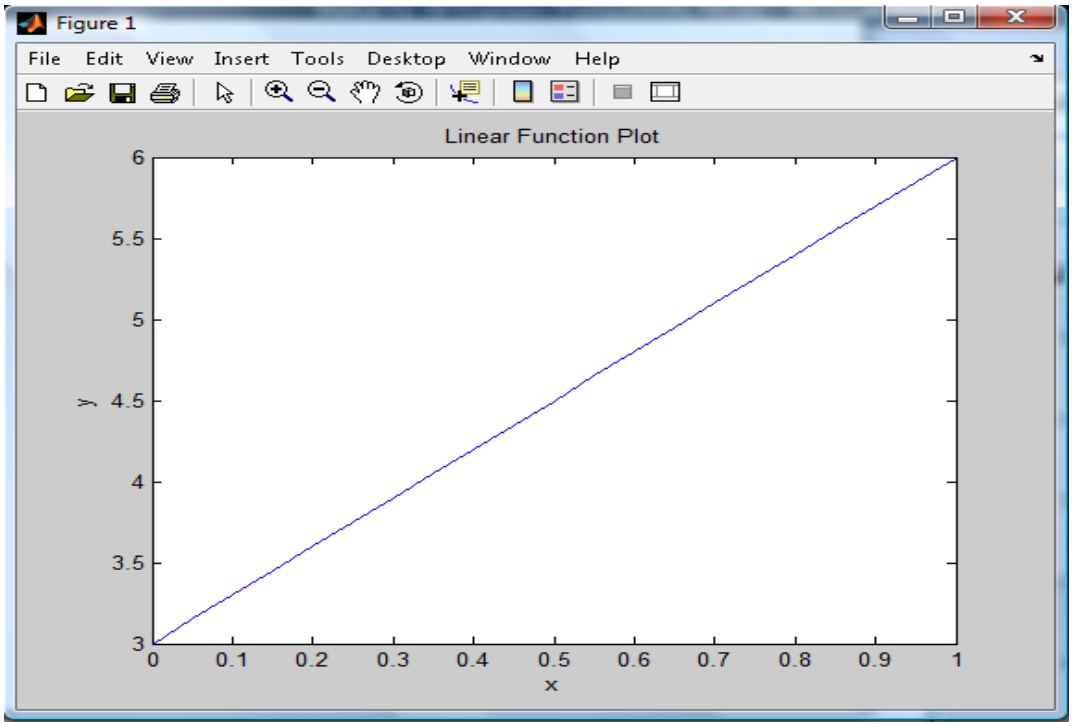
CHAPTER#3: Plots and Functions

3.1. Plots of Mathematical Functions

Plots are important part which helps us to understand more about a problem. All engineers use those kinds of plots to figure out a solution. Plots should have x-label, y-label, and title. Therefore, in this chapter we will learn how to draw plot and add all labels and titles for any types of plots. Further, we will try to draw all types of plots.

3.1.2. Line Plot (command: plot
Example (16)



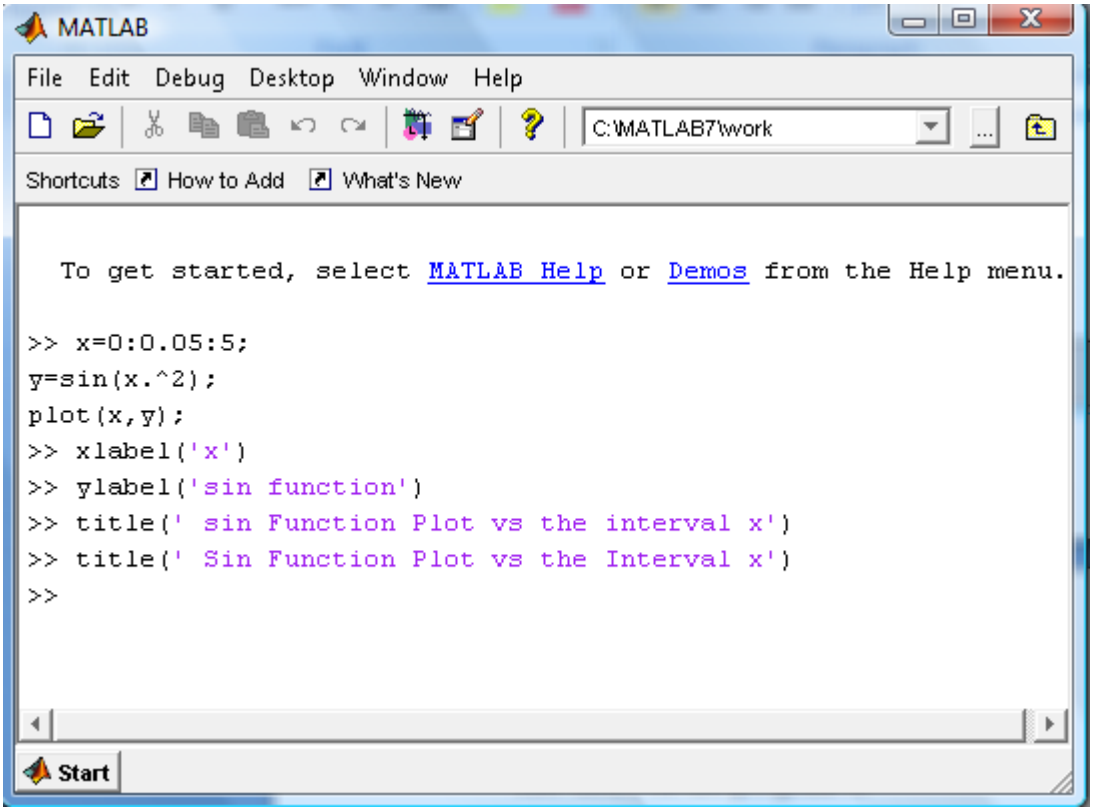


The entire of the program is

```
x=0:0.05:1;
y= 3*x+3;
plot(x,y);
xlabel('x');
ylabel('y');
title('Linear Equation Plot')
```

For practicing, you are able to copy this program and put it on the command window or M-file window.

Example (17)


MATLAB
File Edit Debug Desktop Window Help
C:\MATLAB7\work
Shortcuts How to Add What's New
To get started, select MATLAB Help or Demos from the Help menu.
>> x=0:0.05:5;
y=sin(x.^2);
plot(x,y);
>> xlabel('x')
>> ylabel('sin function')
>> title(' sin Function Plot vs the interval x')
>> title(' Sin Function Plot vs the Interval x')
>>
Start

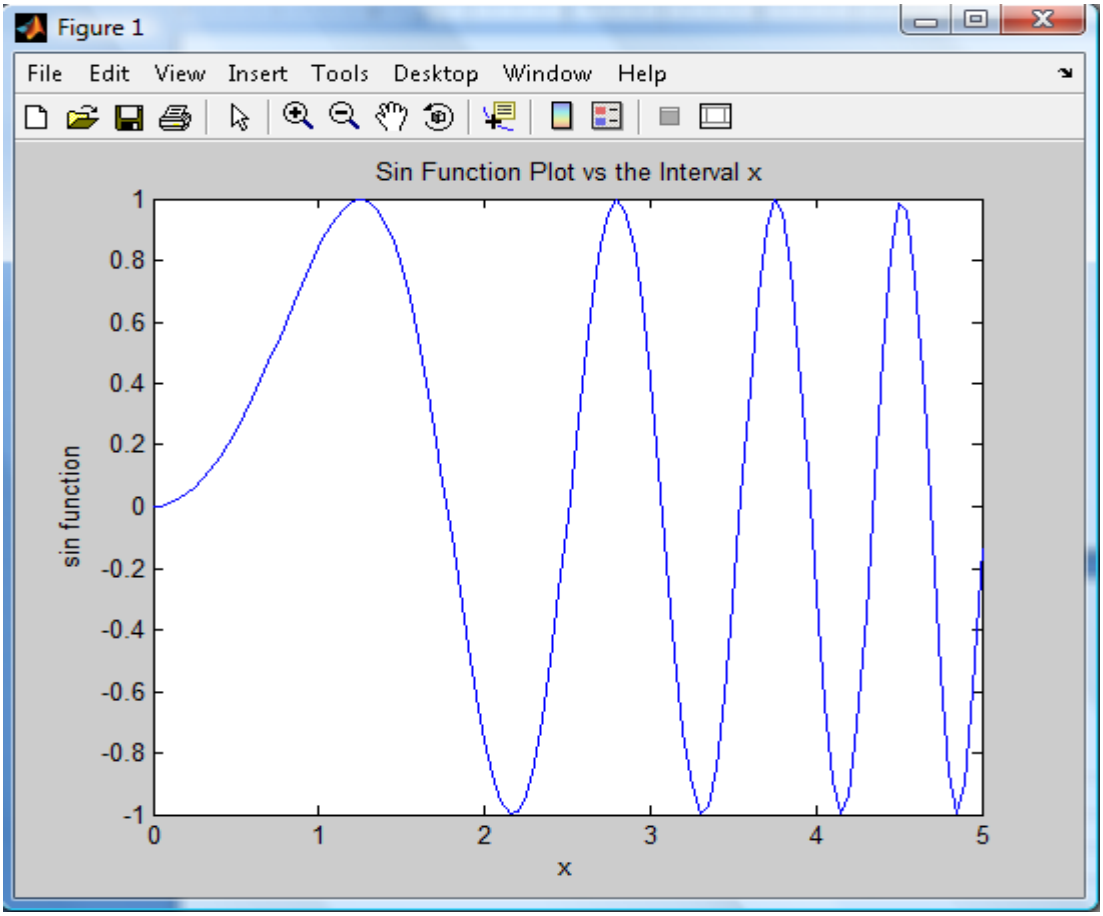
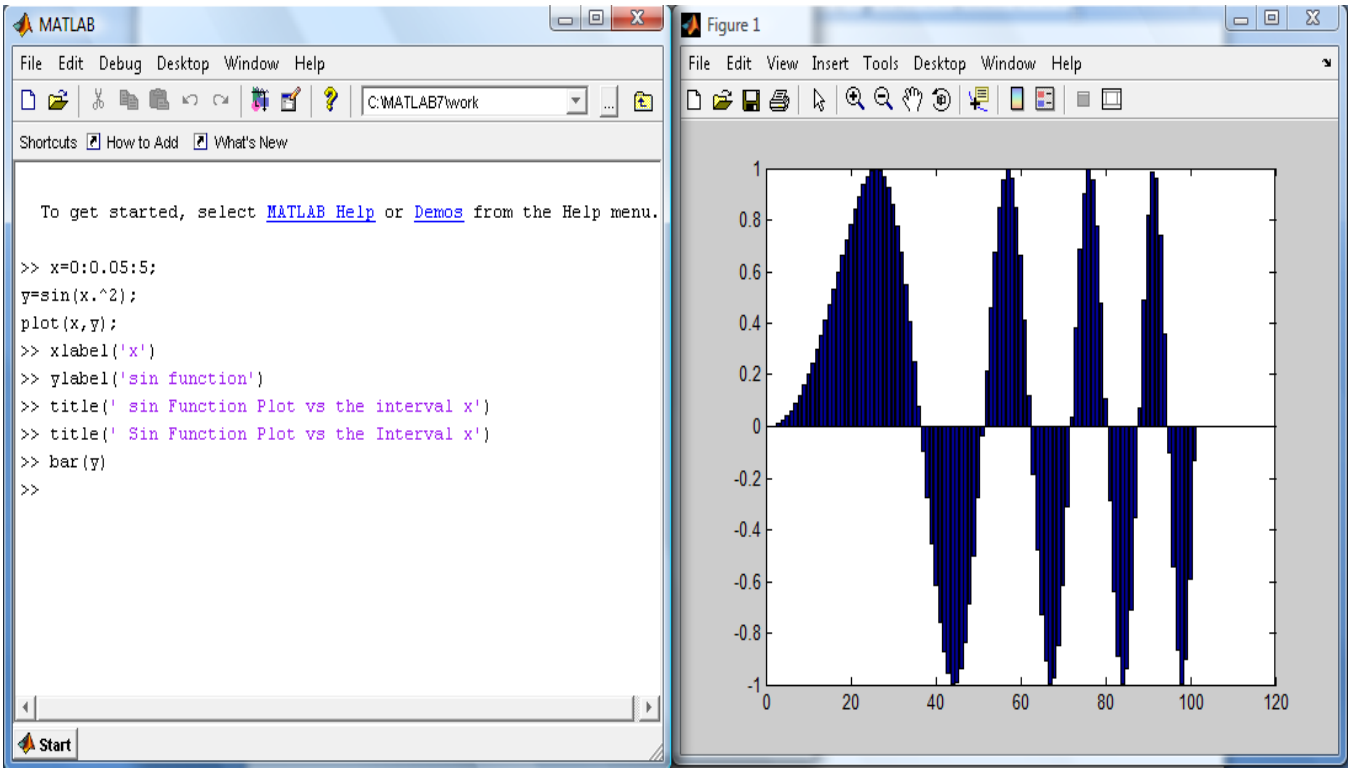


Figure 1
File Edit View Insert Tools Desktop Window Help
Sin Function Plot vs the Interval x
sin function
x
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
1
2
3
4
5

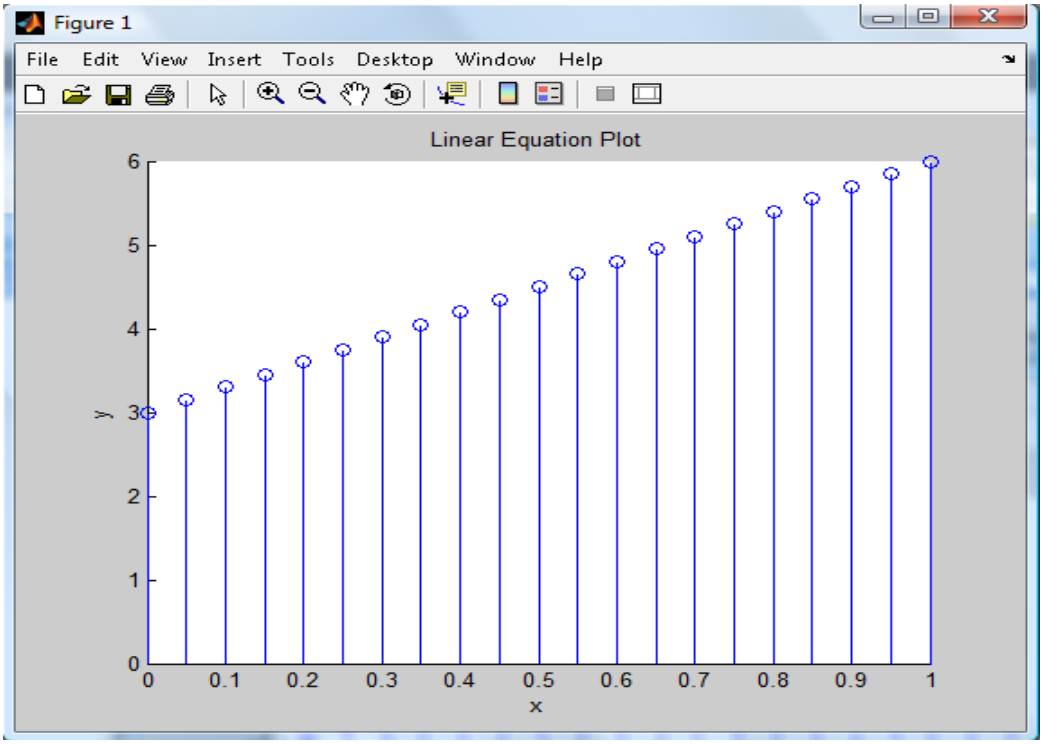
3.1.2. Bar Plot(command: bar)

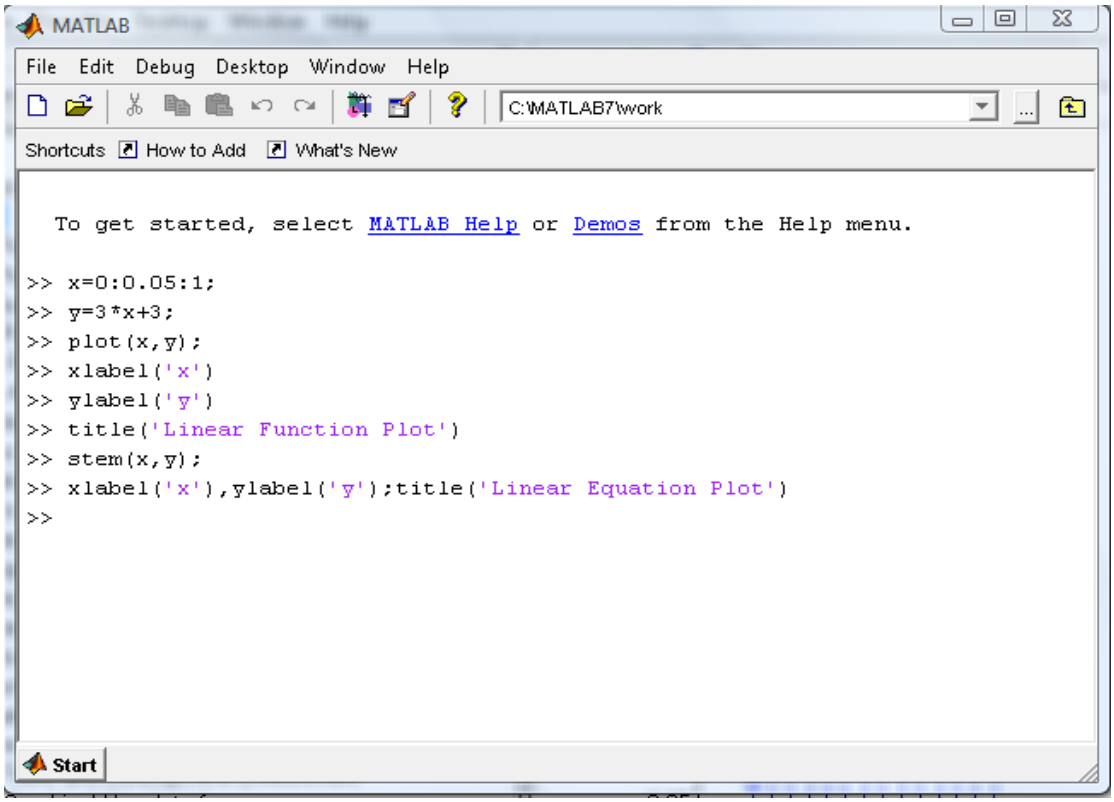
A bar plot is the one type of a plot which is used to show statistical measuring or other usages.
Example (18)



3.1.3. Stem plot (command: stem)

Example (19)



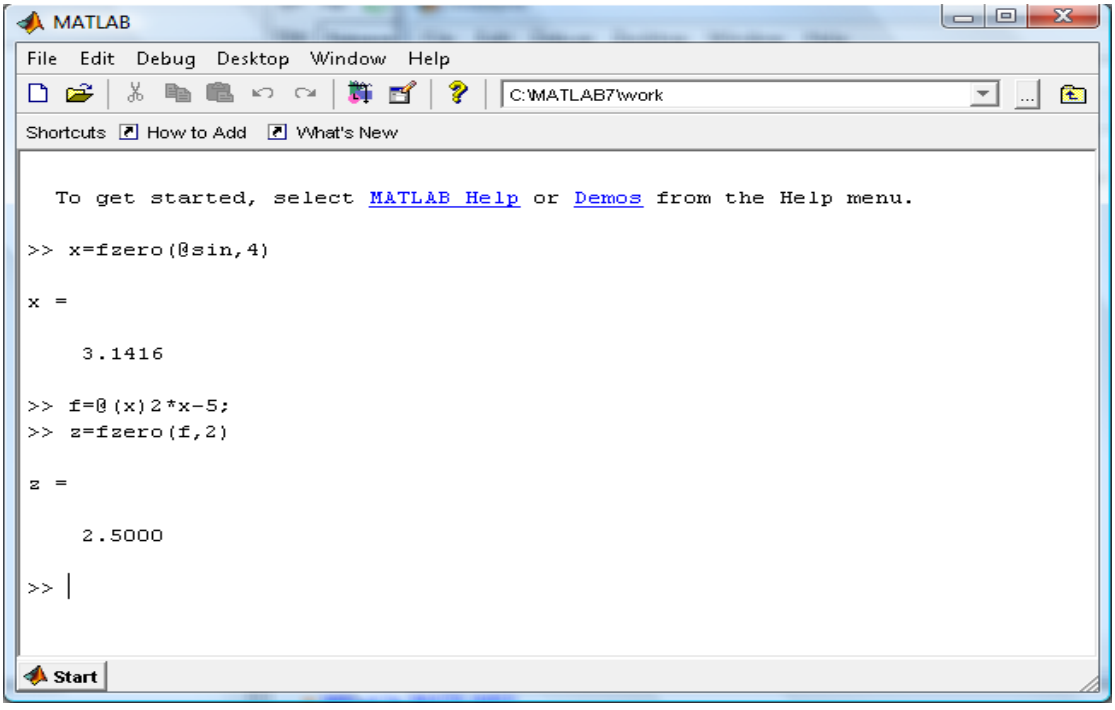


3.2. Linear Equations Functions with one variable (command: fzero (function))

Mathematical equations have different ways to be evaluated. First degree function with one variable is easy to solve.

Explanation:
3x-6=0
x=6/3=2

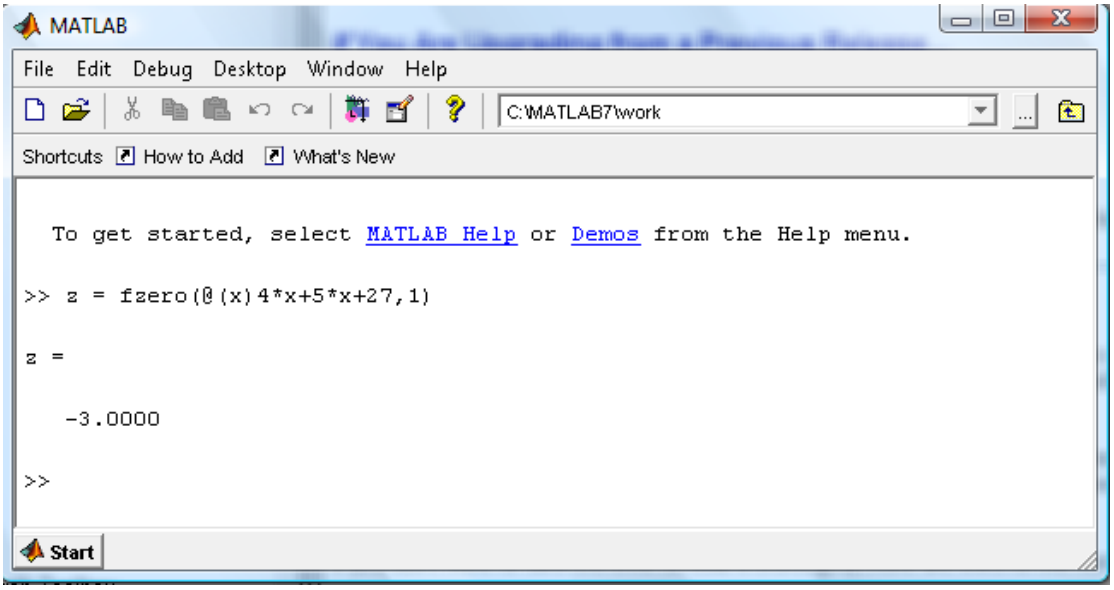
Example (19): Calculate π by finding the zero of the sine function near 4, and also find 2x-5=0
Ans.



Notice: for the near value, you can choose the near value for zero. For example, 2x-5=0, we can choose 0, 1,2…etc. and we will get the same answer.

Example (20) : Evaluate the solution for the following function
4x+5x+27=0

```
>> z = fzero(@(x)4*x+5*x+27,1)

z =

-3.0000
```



3.2.1a Second Degree Functions (command: root)

Solving the second degree equations are by using a quadratic 2nd order equation
$ax^2+bx+c=0$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

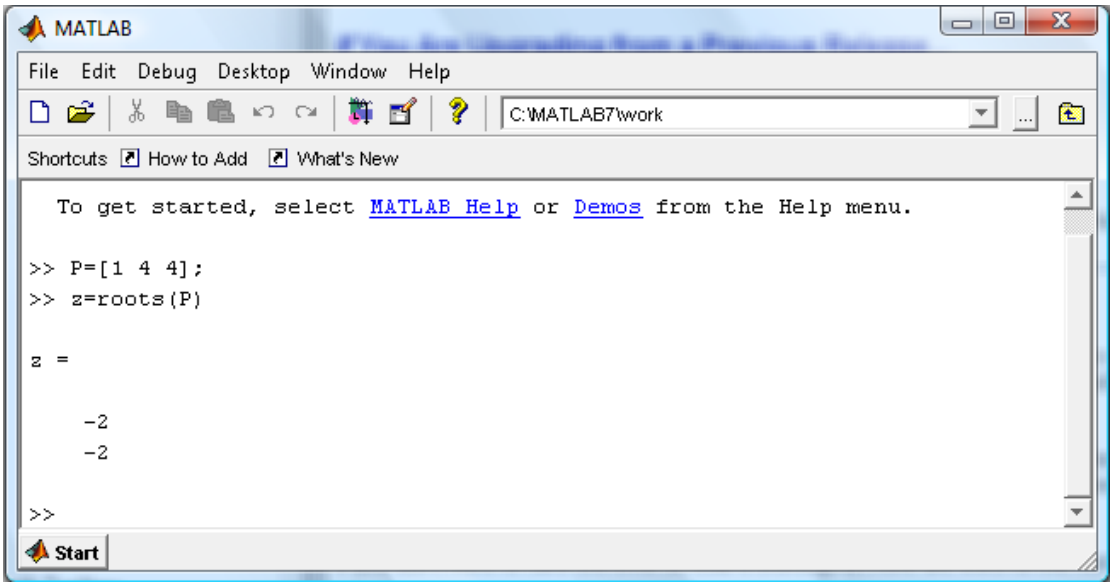
Explanations:

$x^2+4x+4=0$, (x+2)(x+2)=0, so x =-2,-2, two values, multiplication of both of them is equal to 4 and adding both of them is equal to 4.
By using quadratic 2nd order solution formula

$$x = \frac{-4 \pm \sqrt{4^2 - 4 \times 1 \times 4}}{2} = -2, -2$$

By MATLAB

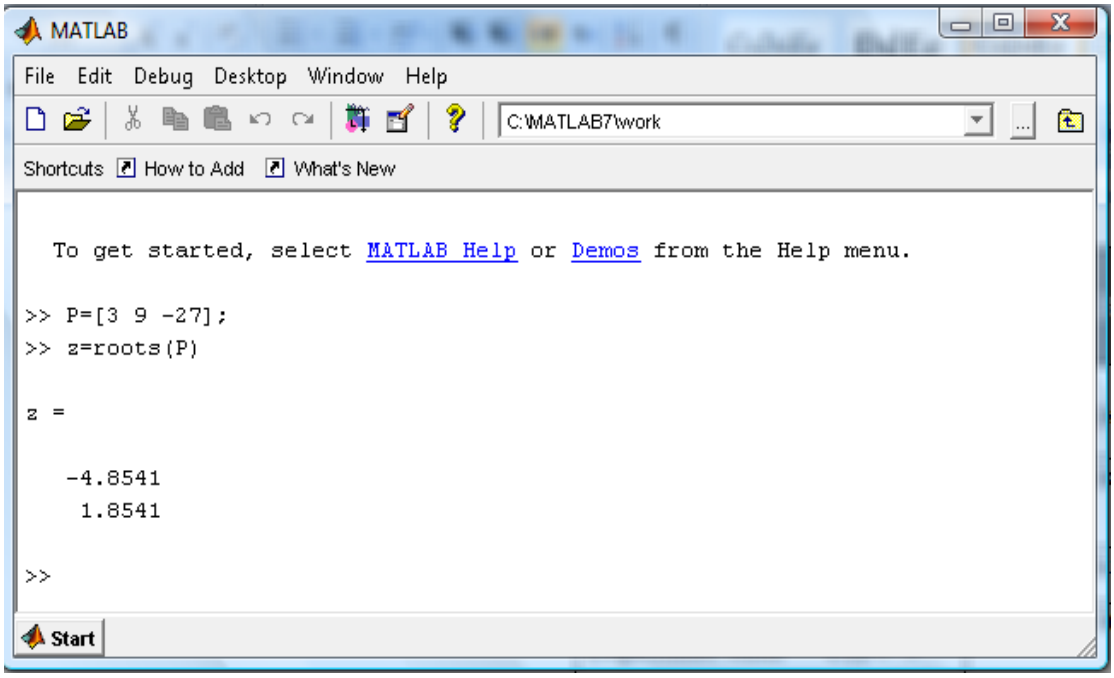
A "P" is the factors of unknown "x", P=[1 4 4], and the command which helps us to solve this type of equations is roots.



Example (21): Estimate the square roots of $3x^2+9x-27$.

Ans.

P= [3 9 -27];
z = roots(P)



3.2.1b. Linear equations with two and three variables

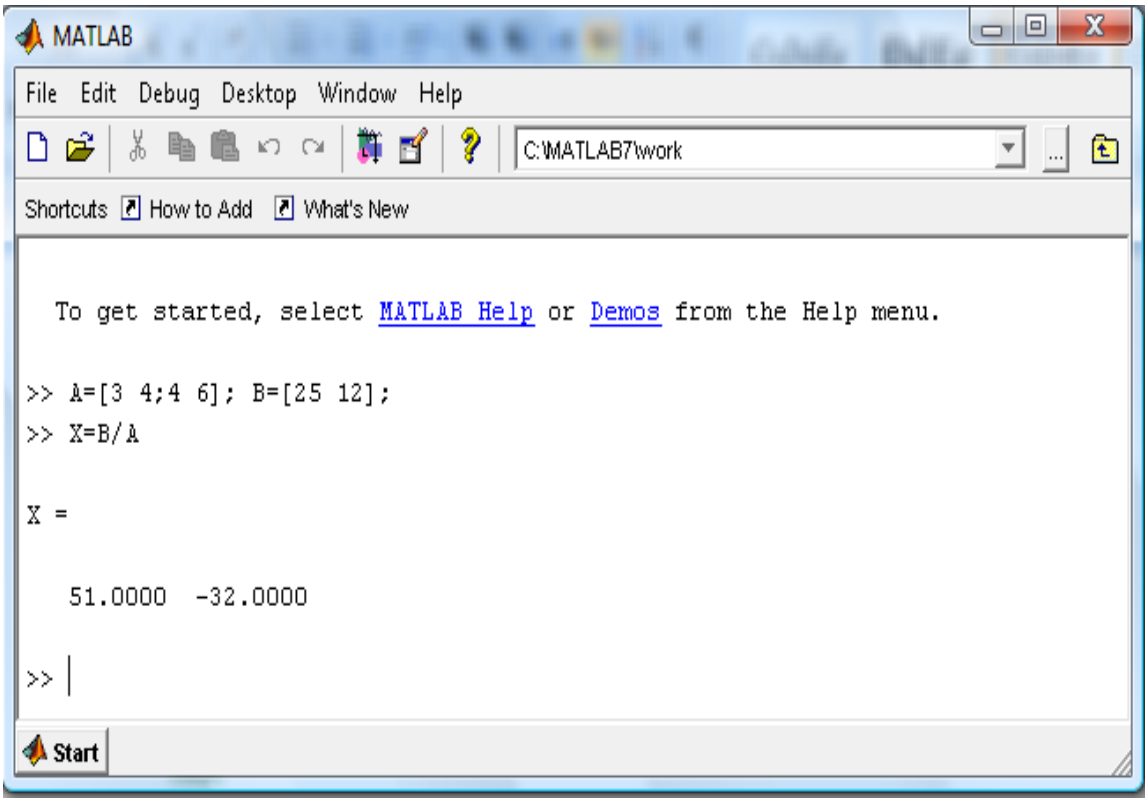
Linear equations might have two or three variables (x,y,z), so we should know how to solve theose types of equations.

3.2.2a Linear Equations with two variables

Example (22): Estimate the values of x and y for the following equations

3x+4y=25
4x+6y=12

Ans.



Explanations:

$AX = B$

$ax + by = k$

$cx + dy = m$

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix} = \begin{vmatrix} k \\ m \end{vmatrix}$$

3.2.2b. Linear Equations with three variables

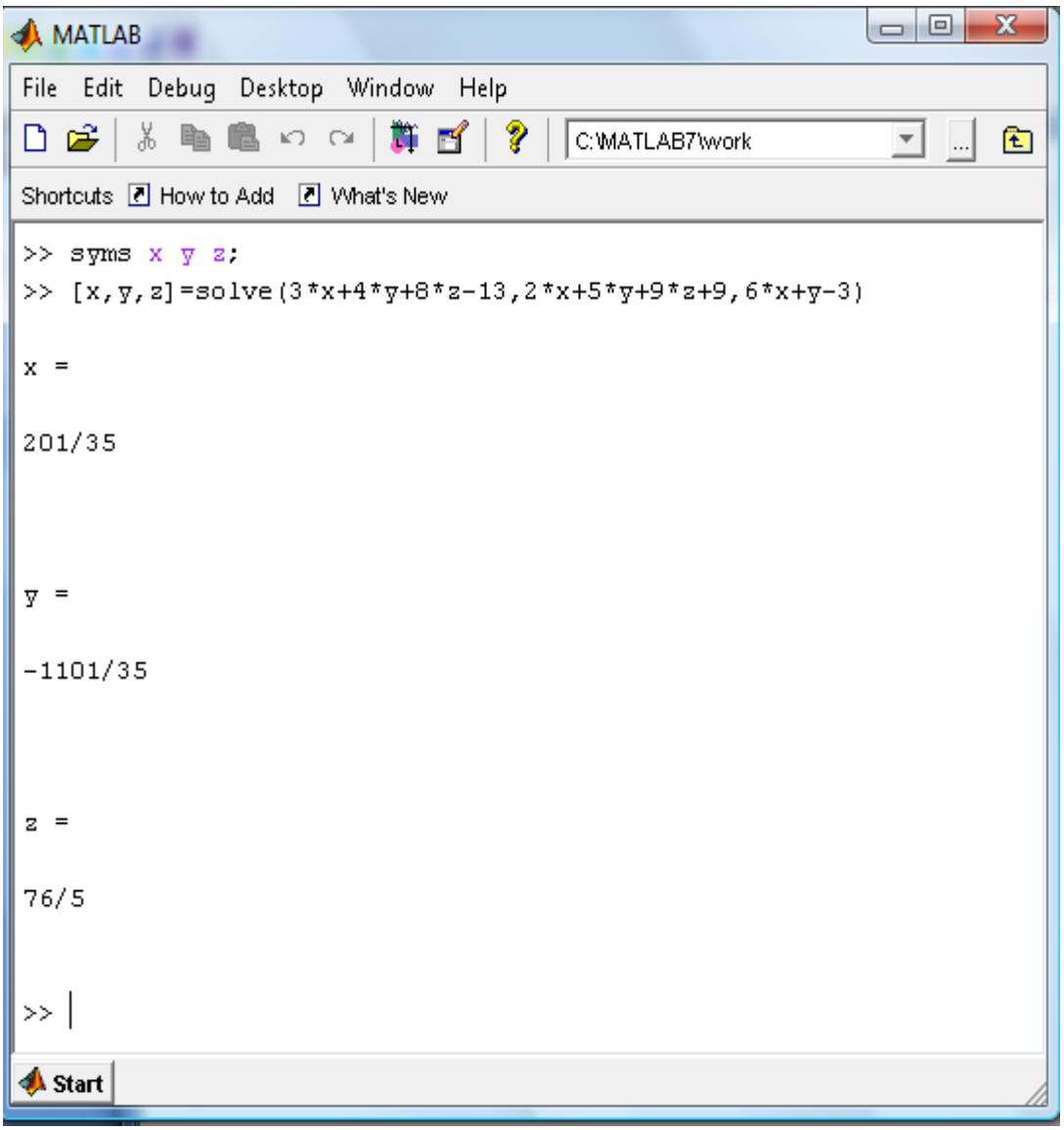
Example (23): Find the solutions of the following equations

3x+4y+8z = 13
2 x + 5y+9z = -9
6 x+y =3

To solve these equations, we have two methods. 1st method by using matrices AX=B and the 2nd method by using solve and syms commands

>> syms x y z;
>> [x,y,z]= solve (eq1,eq2,eq3)

Ans.



Notice: the values of x, y, and z are respectively.

3.4. Nonlinear Equations with, one variable, two variables and three variables.

A mathematical equation or function has different degree, for instance, 1st degree, 2nd degree, 3rd degree or nth degree. Therefore, we will mention all of those types of degrees.

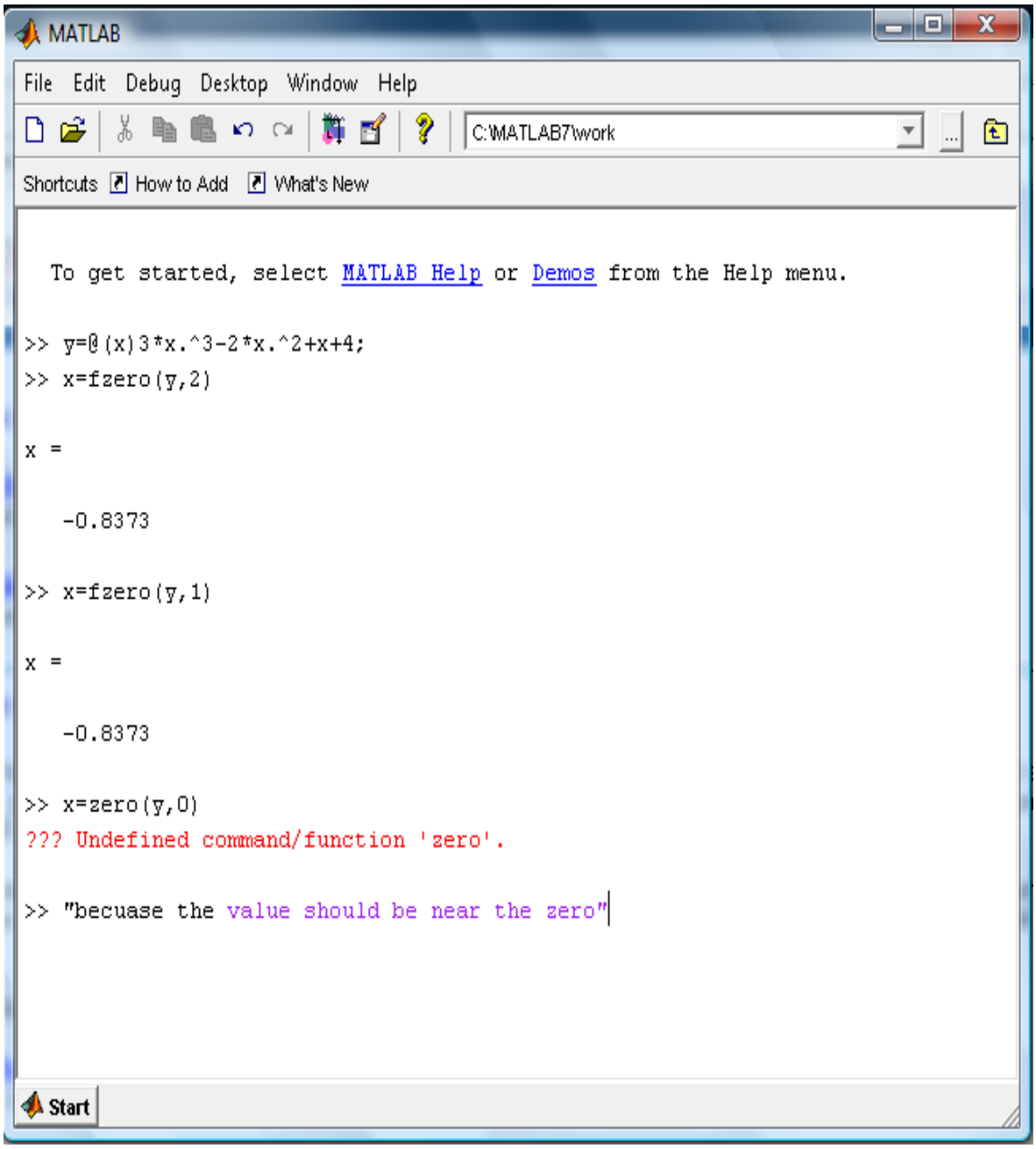
3.4.1a. Nonlinear Equations with One Variable

$Y= ax^3+bx^2+cx+d$, we must find the value of x by using fzero command.

Example (24): Find the value of x for the following equation

$Y= 3x^3-2x^2+x+4$

Ans.



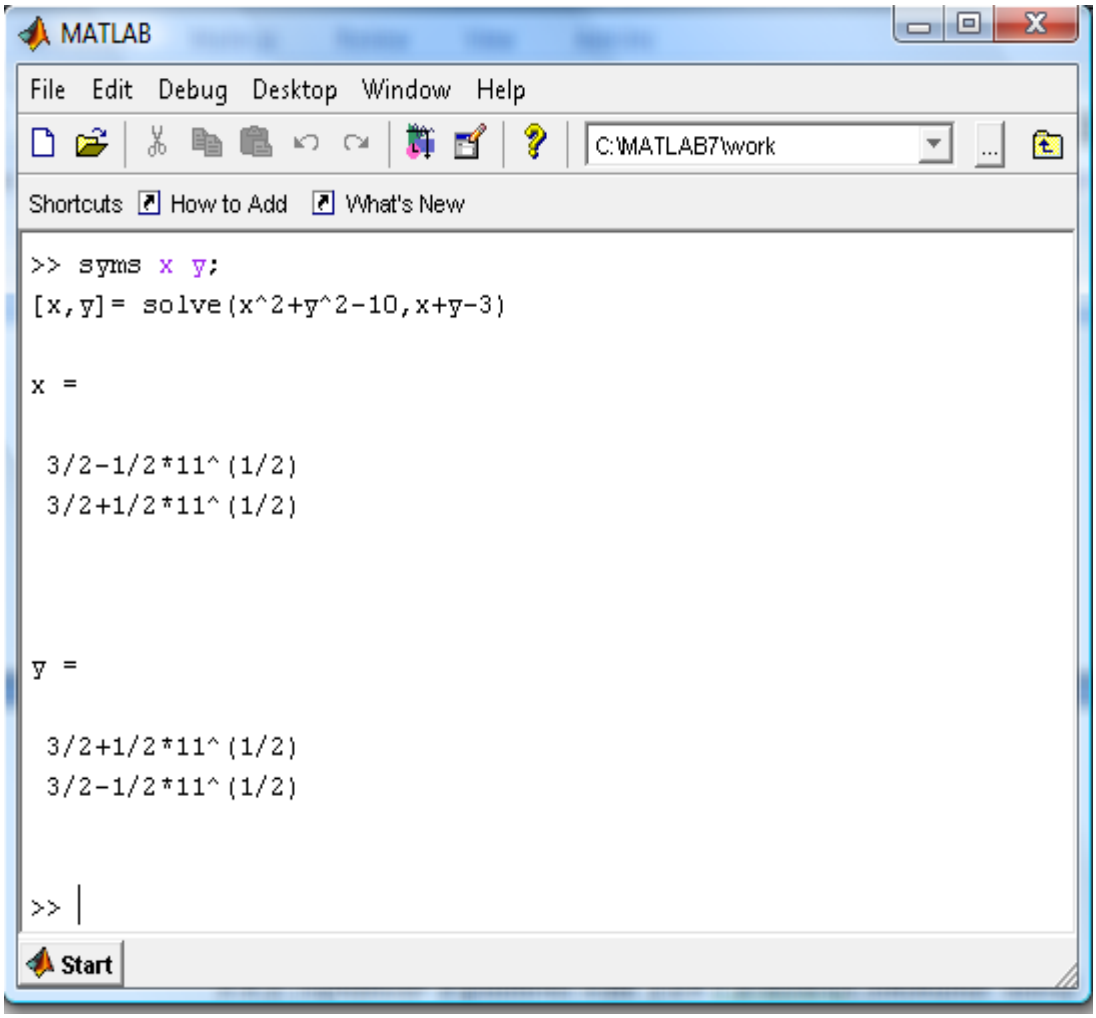
Checking if the answer is correct!?
Substitute in the original equation if the equation is equal to zero the value of the x is correct. Try by yourself.

3.4.1b. Nonlinear Equations with Two Variables (Command: syms, solve)

This section explains how to solve systems of equations using symbolic (syms) math and solve command.
$x^2+y^2=a$
$x+y=b$

Example (25): Find the solution of those mathematical functions
$x^2+y^2= 10$
$x+y =3$

Ans.



3.4.1c. Nonlinear Equations with three Variables (command: syms, solve)

Example (26): Determine the solution for the following equations

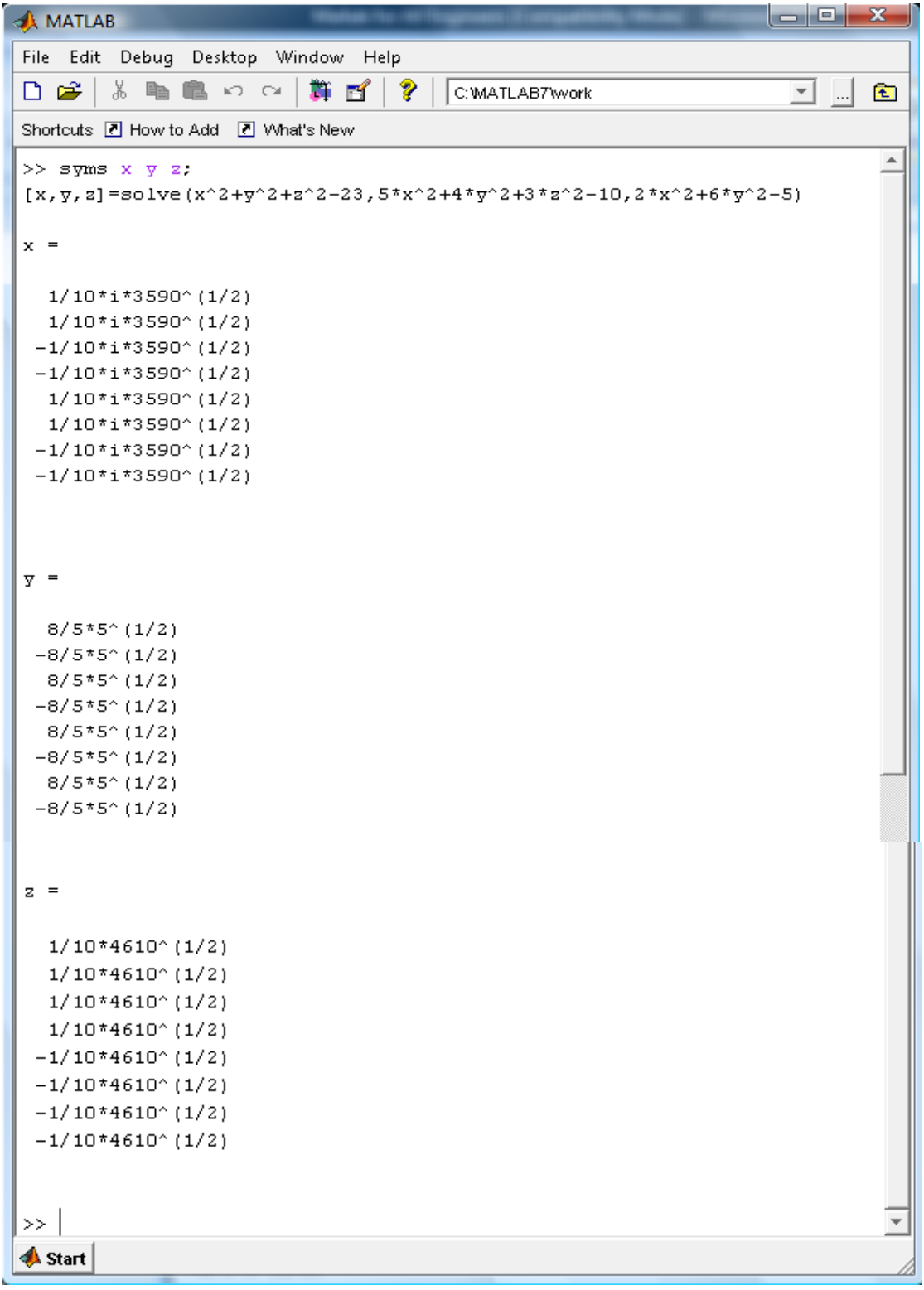
$x^2+y^2+z^2=23$

$5x^2+4y^2+3z^2=10$

$2x^2+6y^2=5$

Ans. it is awkward to solve those types of equations by hand, so we have to use a computer program to solve them. Also, we can use some types of calculation machine which might help students to solve those types of equations.
We will solve those equations by using "syms" and "solve" command. The solution of those equations does not take time to be solved with MATLAB, it takes a few seconds!!.

Ans



3.4. The Trending Line and curve fitting

Command: **"**grid on" is used to draw a graphical sheet.

The trending curve and curve fitting is used if we have a data or if we need to create an equation for a group of data or to fit some data in reported or plotted data.

CHAPTER #7: Numerical Solution Using MATLAB -Finite Difference and FEM for Thermo-fluids and other applications

The Diffusion Models

In this chapter, the one and two-dimensional diffusion model will be mentioned either in steady state or transient using different numerical methods. MATLAB has different "ode" command such as ode45. Those commands are available and applicable in any version of MATLAB This chapter will cover the solution numerical techniques using finite difference method (FDM) as well as using Finite element method (FEM). In some examples, we will validate our code using the analytical solution.

7.1.1.One Dimension Steady State Model

$$\frac{\partial^2 \varphi}{\partial x^2} = 0 \tag{7.1}$$

Here, φ is known as the dependent variable which might represent temperature, velocity, or species.

The equation (7.1) can be solved analytically as

$$\varphi = ax + b \tag{7.2}$$

Where a and b are constants, and they can be evaluated from the boundary conditions.

Example (54): Find the temperature distribution of the 1D steady state bounded slab numerically and analytically at x=0, T=100^{o}C and at x=L, T=40^{o}C.

$$\frac{\partial^2 T}{\partial x^2} = 0$$

$$T = ax + b$$

Using normalization as

$$\theta = \frac{T - T_h}{T_h - T_c},\ \mathrm{X} = \frac{\mathrm{x}}{\mathrm{L}}, \frac{\partial^2 \theta}{\partial X^2} = 0$$

The pervious temperature distribution becomes:

$$\theta(0) = 1, \theta(1) = 0$$

After applying the boundary conditions, the equation becomes

$$\theta = -X + 1 \tag{7.3}$$

The numerical solution of this type of problem

Using central finite difference

$$\frac{\partial^2 \theta}{\partial X^2} = 0$$

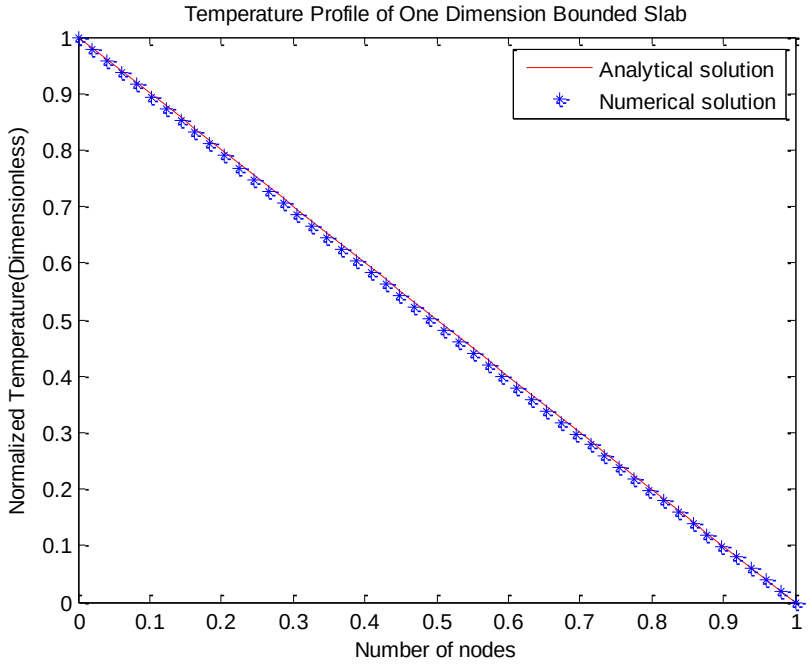
$$\frac{\theta(i+1) - 2\theta(i) + \theta(i-1)}{\Delta X^2} = 0$$

$$\theta(i) = \frac{\theta(i+1) + \theta(i-1)}{2}$$

Using MATLAB to solve the analytical and numerical problem as

```
% 1D heat diffusion
% 1.Setting up the boundary condition
% 2.Calculating of the analytical solution
% 3.Using Central Finite Difference
% 4.Iterative numerical Tech. has been used
clear all
clc
Nx=50; % Number of nodes in x-direction
X=0:1/(Nx-1):1;
% Initialization and Number of iterations
Theat=zeros(Nx);
mstep=1000;
% 1. Boundary Conditions
Theta(1)=1;  % Normalized temperature
Theta(Nx)=0;
% 2. Analytical Solution
for i=1:Nx
    Thetaa(i)=-X(i)+1;
end
for kk=1:mstep

    for i=2:Nx-1
        Theta(i)=0.5*(Theta(i-1)+Theta(i+1));
    end
end
plot(X,Thetaa,'r')
hold on
plot(X,Theta,'*')
hold off
legend('Analytical solution','Numerical solution')
title('Temperature Profile of One Dimension Bounded Slab')
xlabel('Number of nodes');
ylabel('Normalized Temperature(Dimensionless)')
```

The above code is used to solve one dimension steady state heat diffusion problem.
It can be modified to be used for transient one dimension heat diffusion problem.



Figure(7.1). illustrates the numerical and the analytical solution of 1D diffusion steady state

One dimension steady state model with heat generation

$$k\frac{\partial^2 T}{\partial x^2} = -\dot{q} \qquad (7.4)$$

After doing normalization the equation (7.4) becomes

$$\frac{\partial^2 \theta}{\partial X^2} = -\frac{\dot{q} L^2}{k(Th - Tc)}$$

The left hand side terms represents the normalized heat generation Q

$$\frac{\partial^2 \theta}{\partial X^2} = -Q$$

If we assume Q=0.5 (dimensionless values) , the numerical solution becomes as following

Solving the pervious equation (7.4) using numerical and analytical can be done here

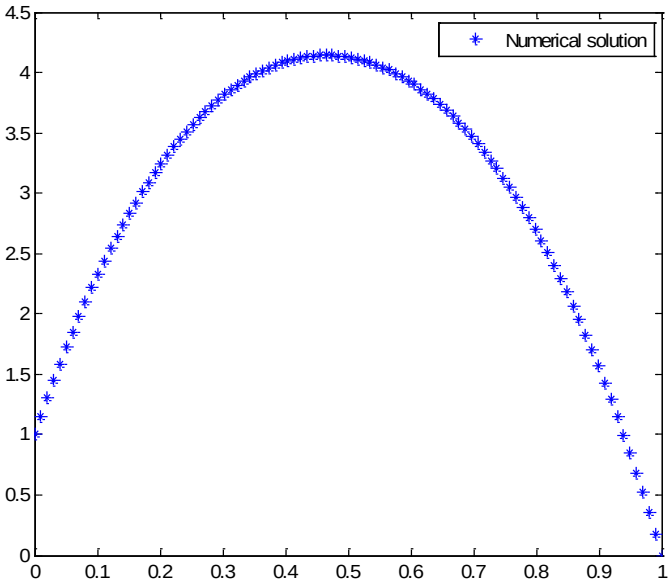
Numerical Solution

$$\frac{\theta(i+1) - 2\theta(i) + \theta(i-1)}{\Delta X^2} = -Q$$

$$\theta(i) = \frac{1}{2} Q \Delta X^2 + \frac{1}{2}(\theta(i+1) + \theta(i-1))$$

Using MATLAB to solve the problem numerically

```
% 1D heat diffusion" Heat generation"
% 1.Setting up the boundary condition
% 2.Calculating of the analytical solution
% 3.Using Central Finite Difference
% 4.Iterative numerical Tech. has been used
clear all
clc
Nx=100; % Number of nodes in x-direction
X=0:1/(Nx-1):1;
% Initialization and Number of iterations
Theta=zeros(Nx);
mstep=1000;
dx=1/(Nx-1)
% 1. Boundary Conditions
Theta(1)=1;  % Normalized temperature
Theta(Nx)=0;
Q=0.5;       % Normalized heat generation
for kk=1:mstep

    for i=2:Nx-1
        Theta(i)=0.5*Q*dx+0.5*(Theta(i-1)+Theta(i+1));
    end
end

plot(X,Theta,'*')

legend('Numerical solution')
title( 'Temperature Profile of One Dimension Bounded Slab with heat Generation')
xlabel('Number of nodes');
ylabel('Normalized Temperature(Dimensionless)')
```



Figure(7.2). illustrates the numerical solution of 1D diffusion steady state model with heat generation

7.1.2. One Dimensional Transient Diffusion Model

One dimensional diffusion transient equation can be written as

$$k\frac{\partial^2\varphi}{\partial x^2}=\rho c\frac{\partial\varphi}{\partial t} \tag{7.5}$$

Here, φ is known as the dependent variable which might represent temperature, velocity, or species. The diffusion process becomes faster as the κ increases and vice versa where, κ is either thermal diffusivity or kinematic viscosity. For scaling purpose τ is the time scale and L is the scale of length; therefore

$$\frac{\partial^2\varphi}{\partial x^2}=\frac{\rho c}{k}\frac{\partial\varphi}{\partial t}$$

From equation 7.5 the length scale is proportional to $L=\sqrt{\alpha\tau}$

Equation 5.1 has second derivative in space; so that, the equation needs two boundaries conditions to be solved and also it needs the initial condition.

Example (55) one –dimensional momentum diffusion for a fluid confined between two parallel plates are due to the motion of the upper lid can be written as

$$\upsilon\frac{\partial^2 u}{\partial x^2}=\frac{\partial u}{\partial t} \tag{7.6}$$

From the equation 7.6, the rate of momentum diffusion relies on the viscosity of the fluid and the distance between two parallel plates.

$$\frac{F}{A}=\mu\frac{\partial u}{\partial y}$$

The finite Difference approximation can be applied (forward for time and central differences in space) here easily:

$$\frac{u^{t+1}(i)-u^t(i)}{\Delta t}=\upsilon\frac{u(i+1)-2u(i)+u(i-1)}{\Delta x^2}$$

$$u^{t+1}(i)=\frac{\Delta t}{\Delta x^2}\upsilon(u^t(i+1)-2u^t(i)+u^t(i-1))+u^t(i) \tag{7.7}$$

For the stability condition (using explicit scheme), the coefficients in the right hand side must be greater than zero. Therefore, to satisfy this condition we should select

$$\Delta t\le\frac{\Delta y^2}{2\upsilon}$$

Where, $\upsilon=\frac{\mu}{\rho}$, For scaling:

The equation 7.7 becomes:

$$U^{\tau+1}(i) = U^{\tau}(1-\tau) + \tau\frac{U^{\tau}(i+1)+U^{\tau}(i-1)}{2}$$

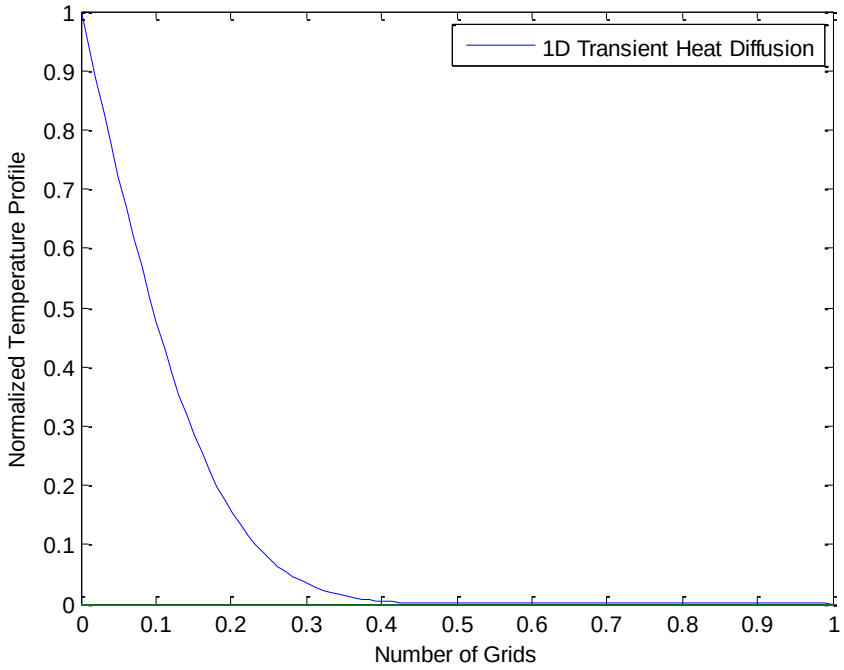
Example (56). Find the temperature distribution of 1D diffusion transient heat transfer. The boundary conditions are at x=0 T=100C, and x=L is T=40C and initial temperature is zero.

$$\frac{\theta^{t+1}(i)-\theta^{t}(i)}{\Delta\tau} = \frac{\theta^{t}(i+1)-2\theta^{t}(i)+\theta^{t}(i-1)}{\Delta X^{2}}$$

$$\Delta\tau = \frac{\Delta t}{\alpha L^{2}}, \quad \alpha = \frac{k}{\rho c}$$

$$\theta^{t+1}(i) = \frac{\Delta\tau}{\Delta X^{2}}[\theta^{t}(i+1)-2\theta^{t}(i)+\theta^{t}(i-1)]+\theta^{t}(i)$$

```
% 1. one dimensional tansient
% 2. Number of nodes in x direction
% 3. Initialization
% 4. setting up the initial and BCs
% 5. Total number of iteration
Nx=100;
Theta=zeros(Nx);  % Temperature distribution at future time
Thetao=zeros(Nx); % Temperature distribution at current time
% Setting the BCs
Thetao(1)=1;
Thetao(Nx)=0;
Theta(1)=1;
Theta(Nx)=0;
% Total number of iterations
Tau=0.00001;
DX=1/(Nx-1);
mstep=1000;
X=0:1/(Nx-1):1;
for kk=1:mstep
    for i=2:Nx-1
        Theta(i)=Tau/DX.^2*(Thetao(i+1)-2*Thetao(i)+Thetao(i-1))+Thetao(i);
    end

    % For update
    Thetao=Theta;
end
plot(X,Theta);
legend('1D Transient Heat Diffusion')
xlabel('Number of Grids');ylabel('Normalized Temperature Profile')
```



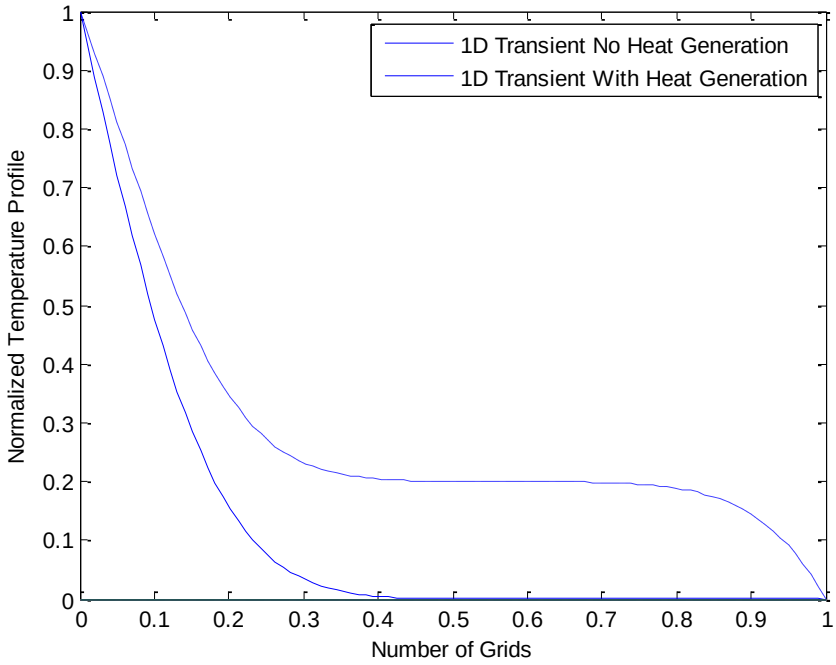
Figure(7.3). illustrates the numerical solution of 1D diffusion transient model

Here, the heat generation can be added as we have done in the steady state 1D diffusion problem.

The governor equation will be :

$$\theta^{t+1}(i) = \frac{\Delta\tau}{\Delta X^2}[\theta^t(i+1) - 2\theta^t(i) + \theta^t(i-1)] + \theta^t(i) + \Delta\tau Q$$

Where Q is the normalized heat generation as it is defined in the previous example.
If the heat generation is increased, it will affect one the temperature distribution over the 1D slab.



Figure(7.4). Illustrates the numerical solution of 1D diffusion transient model with and w/o heat generation.

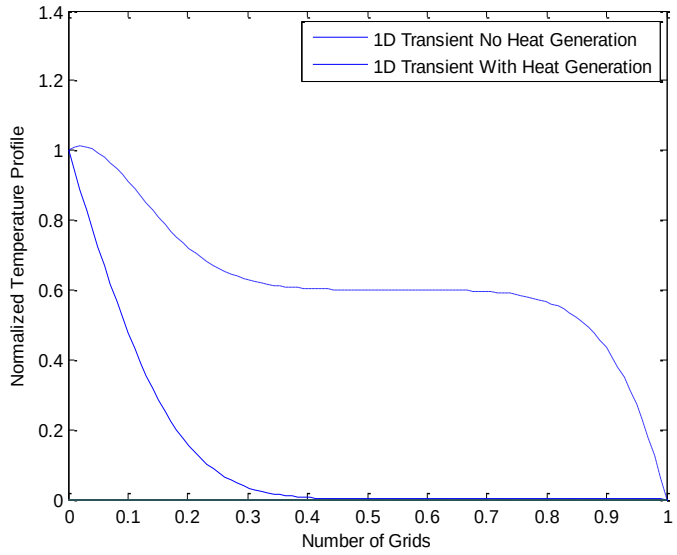
```
% 1. one dimensional transient
% 2. Number of nodes in x direction
% 3. Initialization
% 4. setting up the initial and BCs
% 5. Total number of iteration
Nx=100;
Theta=zeros(Nx);  % Temperature distribution at future time
Thetao=zeros(Nx); % Temperature distribution at current time
% Setting the BCs
Thetao(1)=1;
Thetao(Nx)=0;
Theta(1)=1;
Theta(Nx)=0;
% Setting BCs for
Theta1=zeros(Nx);  % Temperature distribution at future time
Thetao1=zeros(Nx); % Temperature distribution at current time
% Setting the BCs
Thetao1(1)=1;
Thetao1(Nx)=0;
Theta1(1)=1;
Theta1(Nx)=0;
% Total number of iterations
Tau=0.00001;
DX=1/(Nx-1);
mstep=1000;
X=0:1/(Nx-1):1;
Q=20;
for kk=1:mstep
    for i=2:Nx-1
      Theta(i)=Tau/DX.^2*(Thetao(i+1)-2*Thetao(i)+Thetao(i-1))+Thetao(i);
    end
    for i=2:Nx-1
      Theta1(i)=Tau/DX.^2*(Thetao1(i+1)-2*Thetao1(i)+Thetao1(i-1))+Thetao1(i)+Tau*Q;
    end

    % For update temperature
    Thetao=Theta;
    Thetao1=Theta1;
end
plot(X,Theta,'b');
hold on
plot(X,Theta1,'--');
legend('1D Transient No Heat Generation','1D Transient With Heat Generation')
hold off
xlabel('Number of Grids');ylabel('Normalized Temperature Profile')
```

In case of increasing the heat generation 3 times, the temperature distribution will jump up:

The governing equation will be:

$$\theta^{t+1}(i) = \frac{\Delta\tau}{\Delta X^2}[\theta^t(i+1) - 2\theta^t(i) + \theta^t(i-1)] + \theta^t(i) + \Delta\tau Q$$

Selecting the time step and number of grids are important to be sure the code is stable.



--- With heat generation
___ No heat generation

Figure(7.5). The temperature distribution of 1D transient with and w/o heat generation (Q=60)

7.2.1. Two dimensional steady state diffusion

For isotropic material when $k_x=k_y$ (in two dimensions). The steady state heat transfer can be governed by

$$\frac{\partial^2 T}{\partial x^2}+\frac{\partial^2 T}{\partial y^2}=0 \tag{7.8}$$

In case of a material is non-isotropic when $k_x \neq k_y$. The steady state heat transfer can be governed by :

$$\frac{\partial^2 T}{\partial x^2}+\frac{k_y}{k_x}\frac{\partial^2 T}{\partial y^2}=0 \tag{7.9}$$

After normalizing the two-dimensional equation (7.9) yields

$$\frac{\partial^2 \theta}{\partial X^2}+\frac{k_y}{k_x}(\frac{L}{H})^2\frac{\partial^2 \theta}{\partial Y^2}=0$$

The material is uniform in every orientation (isotropic)

$$\frac{\partial^2 \theta}{\partial X^2}+(\frac{L}{H})^2\frac{\partial^2 \theta}{\partial Y^2}=0 \tag{7.10}$$

Discretizing the equation (7.10) using central finite difference yields

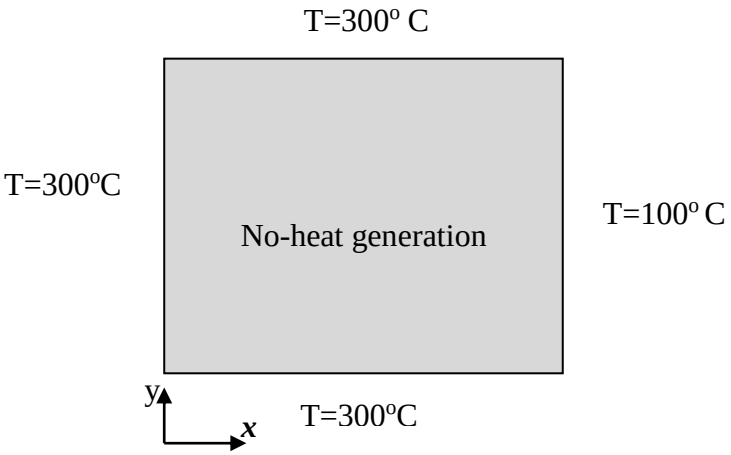
$$2(\frac{1}{\Delta X^2}+\frac{1}{\Delta Y^2})\theta(i,j)=\frac{1}{\Delta X^2}(\theta(j,i+1)+\theta(j,i-1))+\frac{AR^2}{\Delta Y^2}(\theta(j+1,i)+\theta(j-1,i))$$

$$\theta(i,j)=\frac{1}{2(\frac{1}{\Delta X^2}+\frac{1}{\Delta Y^2})}\{\frac{1}{\Delta X^2}(\theta(j,i+1)+\theta(j,i-1))+\frac{AR^2}{\Delta Y^2}(\theta(j+1,i)+\theta(j-1,i))\}$$

Example (57): Find the temperature profile for the following square block in case

a. Isotropic material

b. Non-isotropic material



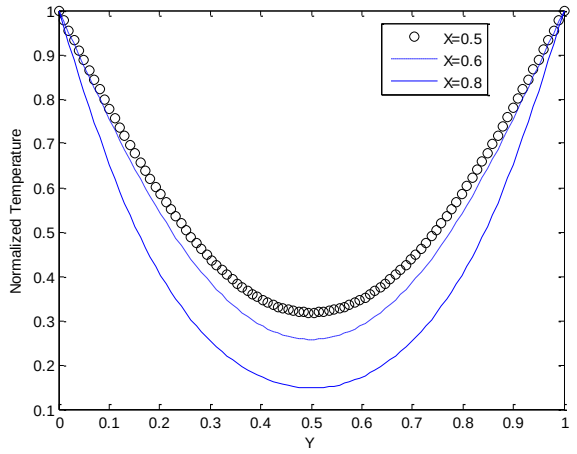
$$\theta=\frac{T-Tc}{T_h-T_c}, X=\frac{x}{L}, Y=\frac{y}{H}$$

Using central finite difference

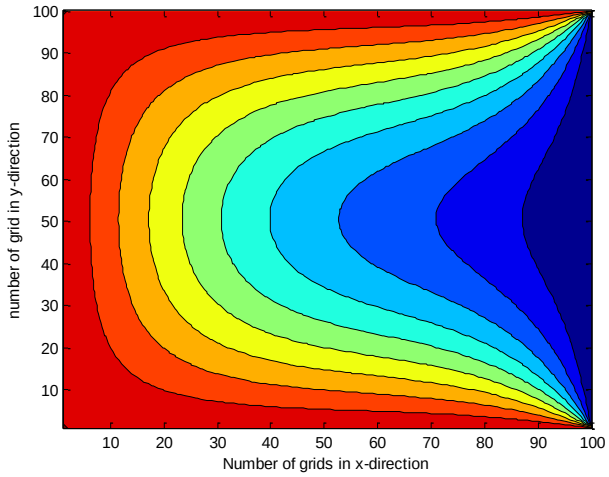
$$\theta(i,j)=\frac{1}{2(\frac{1}{\Delta X^2}+\frac{AR^2}{\Delta Y^2})}\{\frac{1}{\Delta X^2}(\theta(j,i+1)+\theta(j,i-1))+\frac{AR^2}{\Delta Y^2}(\theta(j+1,i)+\theta(j-1,i))\}$$

```
%1. 2D Diffusion model
%2. Setting numbre of grids in x and y-direction
%3. Setting the BCs
%4. Initialization of the Domain
%5. Iterative loop
Nx=100;
Ny=100;
%:::::::::::::::::::::::::::
X=0:1/(Nx-1):1;
Y=0:1/(Ny-1):1;
Theta=zeros(Nx,Ny);
% Setting up the BCs
Theta(:,1)=1;  % Left BCs
Theta(:,Nx)=0; % Right BCs
Theta(1,:)=1;  % Lower BCs
Theta(Ny,:)=1; % Upper BCs
%:::::::::::::::::::::::::::::::
mstep=1000; % Total Number of iterations
dX=1/(Nx-1);% The distance between two adjacent node in x-direction
dY=1/(Ny-1);% The distance between two adjacent nodes in y-direction
AR=Nx/Ny;       % Aspect ratio
for kk=1:mstep

    for i=2:Nx-1
        for j=2:Ny-1
            Theta(j,i)=1/(2*(1/dX.^2+1/dY.^2))*(1/dX.^2*(Theta(j,i+1)+Theta(j,i-1))...
                +AR.^2/dY.^2*(Theta(j+1,i)+Theta(j-1,i)));
        end
    end
end
figure(1)
contourf(Theta);
xlabel('Number of grids in x-direction');
ylabel('number of grid in y-direction');
figure(2)
plot(Y,Theta(:,50),'ko')
```

```
hold on
plot(Y,Theta(:,60),'--')
plot(Y,Theta(:,80))
hold off
xlabel('Y');ylabel('Normalized Temperature')
legend('X=0.5','X=0.6','X=0.8')
```

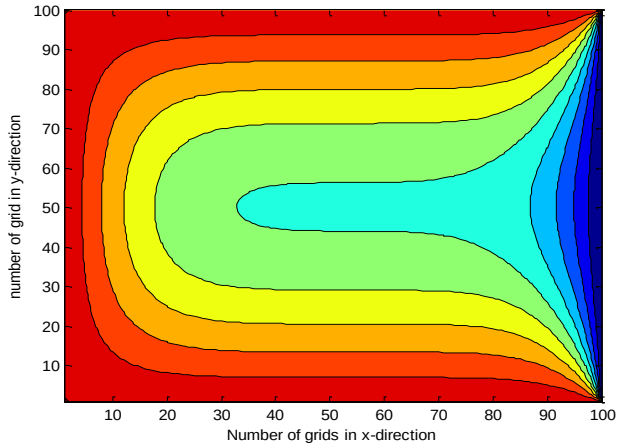


Figure(7.6). The temperature distribution at various cross-section of 2D steady state for isotropic material



Figure(7.7). The temperature distribution contour of 2D steady state for isotropic material

For non-isotropic material



Figure(7.8). The temperature distribution contour of 2D steady state for non-isotropic material

Non-isotropic governing after discretizing will be

$$\theta(i,j)=\frac{1}{2(\frac{1}{\Delta X^2}+\frac{AR^2}{\Delta Y^2}\times K)}\{\frac{1}{\Delta X^2}(\theta(j,i+1)+\theta(j,i-1))+\frac{AR^2}{\Delta Y^2}\times K(\theta(j+1,i)+\theta(j-1,i))\}$$

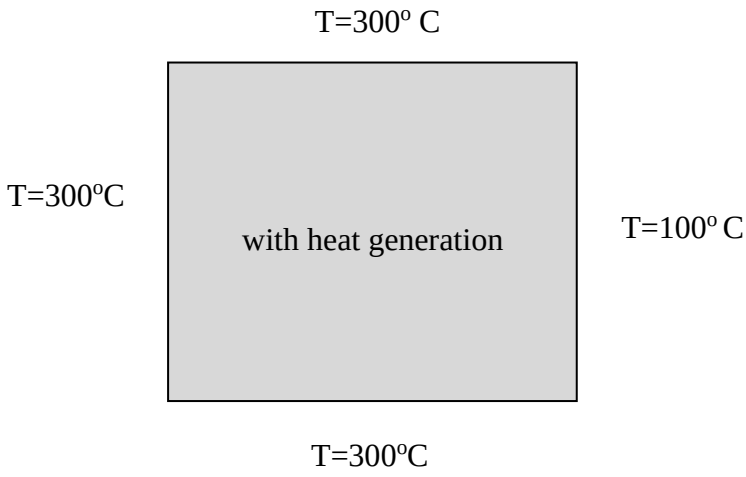
Where $K=\frac{k_y}{k_x}$

```
%1. 2D Diffusion model for non-isotropic material
%2. Setting number of grids in x and y-direction
%3. Setting the BCs
%4. Initialization of the Domain
%5. Iterative loop
Nx=100;
Ny=100;
%:::::::::::::::::::::::::::
X=0:1/(Nx-1):1;
Y=0:1/(Ny-1):1;
Theta=zeros(Nx,Ny);
% Setting up the BCs
Theta(:,1)=1;  % Left BCs
Theta(:,Nx)=0; % Right BCs
Theta(1,:)=1;  % Lower BCs
Theta(Ny,:)=1; % Upper BCs
%:::::::::::::::::::::::::::::::
mstep=1000; % Total Number of iterations
dX=1/(Nx-1);% The distance between two adjacent node in x-direction
dY=1/(Ny-1);% The distance between two adjacent nodes in y-direction
AR=Nx/Ny;   % Aspect ratio
K=10;        % Thermal conductivity ratio (dimensionless)
for kk=1:mstep

    for i=2:Nx-1
        for j=2:Ny-1
            Theta(j,i)=1/(2*(1/dX.^2+(AR.^2*K)/dY.^2))*(1/dX.^2*(Theta(j,i+1)+Theta(j,i-
1))...
                +(AR.^2*K)/dY.^2*(Theta(j+1,i)+Theta(j-1,i)));
        end
    end
end
figure(1)
contourf(Theta);
xlabel('Number of grids in x-direction');
ylabel('number of grid in y-direction');
figure(2)
plot(Y,Theta(:,50),'ko')
hold on
plot(Y,Theta(:,60),'--')
plot(Y,Theta(:,80))
hold off
xlabel('Y');ylabel('Normalized Temperature')
legend('X=0.5','X=0.6','X=0.8')
```

Example (58): Find the temperature profile for the following square block in case

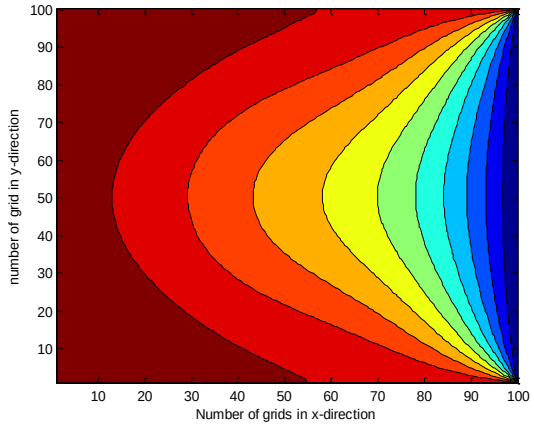
a. Isotropic material

b. Non-isotropic material

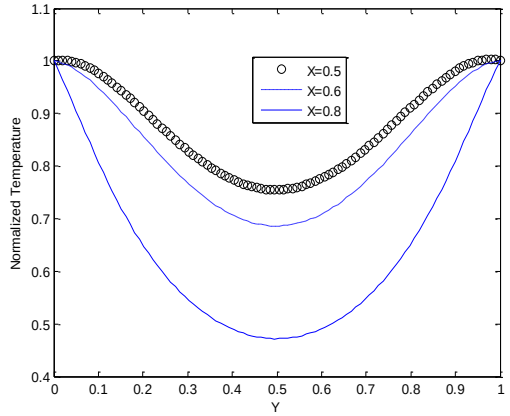


Isotropic governing after discretizing will be

$$\theta(i,j)=\frac{1}{2(\frac{1}{\Delta X^2}+\frac{AR^2}{\Delta Y^2})}\{\frac{1}{\Delta X^2}(\theta(j,i+1)+\theta(j,i-1))+\frac{AR^2}{\Delta Y^2}(\theta(j+1,i)+\theta(j-1,i))+Q\}$$



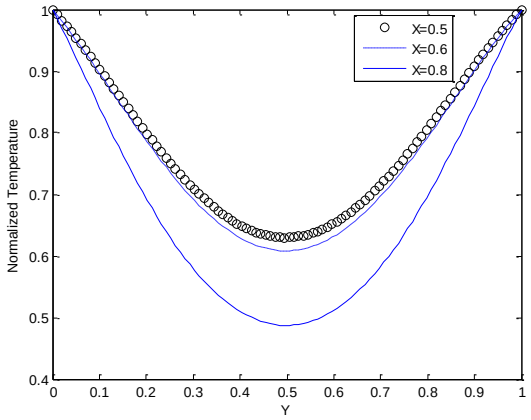
Figure(7.9). The temperature distribution contour of 2D steady state for isotropic material with heat generation



Figure(7.10). The temperature distribution at various cross-sections of 2D steady state for isotropic material with heat generation.

b.

```
%1. 2D Diffusion model
%2. Setting numbre of grids in x and y-direction
%3. Setting the BCs
%4. Initialization of the Domain
%5. Iterative loop
Nx=100;
Ny=100;
%:::::::::::::::::::::::::::
X=0:1/(Nx-1):1;
Y=0:1/(Ny-1):1;
Theta=zeros(Nx,Ny);
% Setting up the BCs
Theta(:,1)=1;  % Left BCs
Theta(:,Nx)=0; % Right BCs
Theta(1,:)=1;  % Lower BCs
Theta(Ny,:)=1; % Upper BCs
%:::::::::::::::::::::::::::::::
mstep=1000; % Total Number of iterations
dX=1/(Nx-1);% The distance between two adjacent node in x-direction
dY=1/(Ny-1);% The distance between two adjacent nodes in y-direction
AR=Nx/Ny;    % Aspect ratio
K=3;         % Thermal conductivity ratio (dimensionless)
Q=10;        % Heat Generation Dimensionless
for kk=1:mstep

    for i=2:Nx-1
        for j=2:Ny-1
            Theta(j,i)=1/(2*(1/dX.^2+(AR.^2*K)/dY.^2))*(1/dX.^2*(Theta(j,i+1)+Theta(j,i-
1))...
                +(AR.^2*K)/dY.^2*(Theta(j+1,i)+Theta(j-1,i))+Q);
        end
    end
end
figure(1)
contourf(Theta);
xlabel('Number of grids in x-direction');
ylabel('number of grid in y-direction');
figure(2)
plot(Y,Theta(:,50),'ko')
hold on
plot(Y,Theta(:,60),'--')
plot(Y,Theta(:,80))
hold off
xlabel('Y');ylabel('Normalized Temperature')
legend('X=0.5','X=0.6','X=0.8')
```



Figure(7.11). The temperature distribution at various cross-sections of 2D steady state for non-isotropic

material with heat generation.

7.2.2. Two dimensional transient heat transfer diffusion

The two dimensional transient diffusion heat transfer is governed by

$$k_x \frac{\partial^2 T}{\partial x^2} + k_y \frac{\partial^2 T}{\partial y^2} = \rho c \frac{\partial T}{\partial t} \tag{7.10}$$

For isotropic material $k_y = k_y$

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = \frac{1}{\alpha} \frac{\partial T}{\partial t}$$

For non-isotropic material $k_y \neq k_y$

$$\frac{\partial^2 T}{\partial x^2} + \frac{k_y}{k_x} \frac{\partial^2 T}{\partial y^2} = \frac{1}{\alpha_x} \frac{\partial T}{\partial t} \tag{7.11}$$

$\kappa = \frac{k_y}{k_x}$ is thermal conductivity ratio (dimensionless)

$$\frac{\partial^2 T}{\partial x^2} + \kappa \frac{\partial^2 T}{\partial y^2} = \frac{1}{\alpha_x} \frac{\partial T}{\partial t}$$

If the material is isotropic, the thermal conductivity ratio is 1 and the pervious equation becomes after scaling is

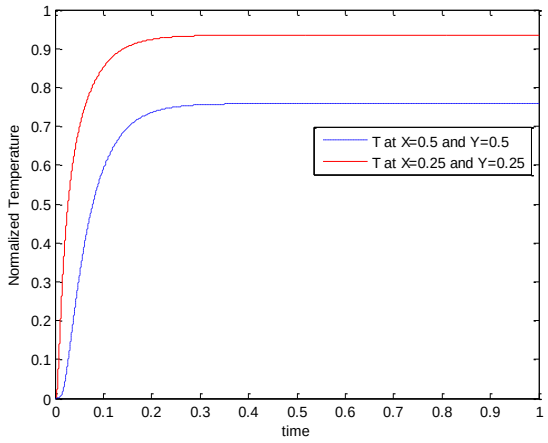
$$\frac{\partial^2 \theta}{\partial X^2} + AR^2 \frac{\partial^2 \theta}{\partial Y^2} = \frac{1}{\alpha} \frac{\partial \theta}{\partial t}$$

Using explicit method as

$$\frac{\theta^{\tau}(j,i+1) - 2\theta^{\tau}(j,i) + \theta^{\tau}(j,i-1)}{\Delta X^2} + AR^2 \frac{\theta^{\tau}(j,i+1) - 2\theta^{\tau}(j,i) + \theta^{\tau}(j,i-1)}{\Delta Y^2} = \frac{\theta^{\tau+1}(j,i) - \theta^{\tau}(j,i)}{\Delta \tau}$$

$$\theta^{\tau+1}(j,i) = \Delta\tau \times [\frac{\theta^{\tau}(j,i+1) - 2\theta^{\tau}(j,i) + \theta^{\tau}(j,i-1)}{\Delta X^2} + AR^2 \frac{\theta^{\tau}(j,i+1) - 2\theta^{\tau}(j,i) + \theta^{\tau}(j,i-1)}{\Delta Y^2}] + \theta^{\tau}(j,i)$$

Same as the example [7.5] , but using transient explicit method with and w/o heat generation.



Figure(7.12). The temperature distribution at various cross-sections of 2D transient for isotropic material.

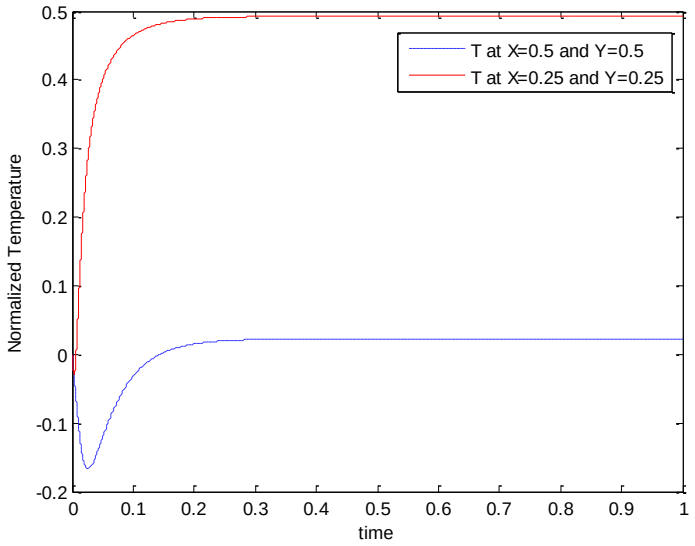
```
%1. 2D Diffusion model
%2. Setting number of grids in x and y-direction
%3. Setting the BCs
%4. Initialization of the Domain
%5. Iterative loop
clear all
clc
Nx=50;
Ny=50;
%:::::::::::::::::::::::::::
X=0:1/(Nx-1):1;
Y=0:1/(Ny-1):1;
Thetao=zeros(Nx,Ny);
Theta=zeros(Nx,Ny);
% Setting up the BCs at current time
Thetao(:,1)=1;  % Left BCs
Thetao(:,Nx)=0; % Right BCs
Thetao(1,:)=1;  % Lower BCs
Thetao(Ny,:)=1; % Upper BCs
%Setting up the BCs at Future time
Theta(:,1)=1;  % Left BCs
Theta(:,Nx)=0; % Right BCs
Theta(1,:)=1;  % Lower BCs
Theta(Ny,:)=1; % Upper BCs
mstep=10000; % Total Number of iterations
dX=1/(Nx-1);% The distance between two adjacent node in x-direction
dY=1/(Ny-1);% The distance between two adjacent nodes in y-direction
AR=Nx/Ny;    % Aspect ratio
K=3;         % Thermal conductivity ratio (dimensionless)
Q=10;        % Heat Generation Dimensionless
dt=0.0001;
for kk=1:mstep
    time(kk)=kk*dt;
    for i=2:Nx-1
        for j=2:Ny-1
            Theta(j,i)= dt*((Thetao(j,i+1)-2*Thetao(j,i)+Thetao(j,i-1))/dX.^2+...
            AR.^2*(Thetao(j+1,i)-2*Thetao(j,i)+Thetao(j-1,i))/dY.^2)+Thetao(j,i);
        end
    end
    Thetao=Theta;
    Temp1(kk)=Theta(Ny/2,Nx/2);
    Temp2(kk)=Theta(round(Ny/4),round(Nx/4));
end
figure(1)
contourf(Theta);
xlabel('Number of grids in x-direction');
ylabel('number of grid in y-direction');
figure(2)
plot(time,Temp1,'--')
hold on
plot(time,Temp2,'r')
hold off
legend ('T at X=0.5 and Y=0.5','T at X=0.25 and Y=0.25');
xlabel ('time');ylabel('Normalized Temperature');
```

With heat generation:

```
%1. 2D Diffusion model
%2. Setting numbre of grids in x and y-direction
%3. Setting the BCs
%4. Initialization of the Domain
%5. Iterative loop
clear all
clc
Nx=50;
Ny=50;
%:::::::::::::::::::::::::::
X=0:1/(Nx-1):1;
Y=0:1/(Ny-1):1;
Thetao=zeros(Nx,Ny);
Theta=zeros(Nx,Ny);
% Setting up the BCs at current time
Thetao(:,1)=1;  % Left BCs
Thetao(:,Nx)=0; % Right BCs
Thetao(1,:)=1;  % Lower BCs
Thetao(Ny,:)=1; % Upper BCs
%Setting up the BCs at Future time
Theta(:,1)=1;  % Left BCs
Theta(:,Nx)=0; % Right BCs
Theta(1,:)=1;  % Lower BCs
Theta(Ny,:)=1; % Upper BCs
mstep=10000; % Total Number of iterations
dX=1/(Nx-1);% The distance between two adjacent node in x-direction
dY=1/(Ny-1);% The distance between two adjacent nodes in y-direction
AR=Nx/Ny;    % Aspect ratio
Q=10;        % Heat Generation Dimensionless
dt=0.0001;
for kk=1:mstep
    time(kk)=kk*dt;
    for i=2:Nx-1
        for j=2:Ny-1
            Theta(j,i)= dt*((Thetao(j,i+1)-2*Thetao(j,i)+Thetao(j,i-1))/dX.^2+...
            AR.^2*(Thetao(j+1,i)-2*Thetao(j,i)+Thetao(j-1,i))/dY.^2)+Thetao(j,i)-dt*Q;
        end
    end
    Thetao=Theta;
    Temp1(kk)=Theta(Ny/2,Nx/2);
    Temp2(kk)=Theta(round(Ny/4),round(Nx/4));
end
figure(1)
contourf(Theta);
xlabel('Number of grids in x-direction');
ylabel('number of grid in y-direction');
figure(2)
plot(time,Temp1,'--')
hold on
plot(time,Temp2,'r')
hold off
legend ('T at X=0.5 and Y=0.5','T at X=0.25 and Y=0.25');
xlabel ('time');ylabel('Normalized Temperature');
```

In this code, we just added the heat generation in the main governing equation as

$$\theta^{\tau+1}(j,i)=\Delta\tau\times[\frac{\theta^{\tau}(j,i+1)-2\theta^{\tau}(j,i)+\theta^{\tau}(j,i-1)}{\Delta X^2}+AR^2\frac{\theta^{\tau}(j,i+1)-2\theta^{\tau}(j,i)+\theta^{\tau}(j,i-1)}{\Delta Y^2}]+\theta^{\tau}(j,i)-\Delta\tau\times Q$$



Figure(7.13). The temperature distribution at various cross-sections of 2D transient for isotropic material with heat generation.

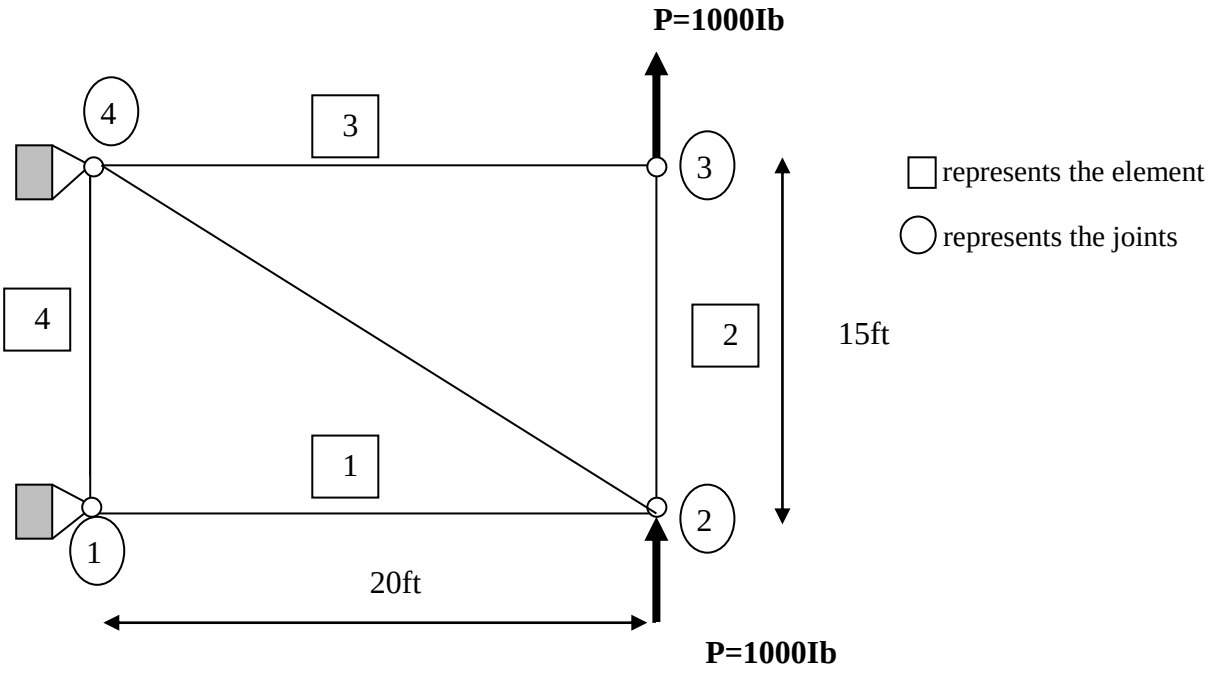
Table7.1 Second order Approximations of the Finite difference schemes

Second order Finite Difference Scheme	
Forward scheme	$(\frac{\partial\theta}{\partial x})_i = \frac{\theta(i+1)-\theta(i)}{\Delta x}$
Backward scheme	$(\frac{\partial\theta}{\partial x})_i = \frac{\theta(i)-\theta(i-1)}{\Delta x}$
Central finite difference 1D (x-direction	$(\frac{\partial\theta}{\partial x})_i = \frac{\theta(i+1)-\theta(i-1)}{2\Delta x}$
Central finite difference 2D(x-direction)	$(\frac{\partial^2\theta}{\partial x^2})_{i,j} = \frac{\theta(i+1,j)-2\theta(i,j)+\theta(i-1,j)}{(\Delta x)^2}$
Central finite difference 2D (y-direction)	$(\frac{\partial^2\theta}{\partial y^2})_{i,j} = \frac{\theta(i,j+1)-2\theta(i,j)+\theta(i,j-1)}{(\Delta y)^2}$

7.2.3. Truss using FEM using MATLAB

The finite element is used to solve different engineering applications. This section will cover one dimensional bar with various cross-sectional areas when the external concentrated load exerted on the bar. Furthermore, the finite element is employed here to evaluate reaction force, stresses and elongation.
The following two examples show how a programmer can use the MTALB to solve any engineering problems. FEM is a numerical technique and MATLAB is a tool to solve a problem.

Example (59): Determine the forces and stresses on each elements if E=30 x10^6 Psi and A=2 in. The dispalcment at 2 is 0.05 in




```
% Objective: This computer program computes the 2 truss
% clear memory
clear all
clc
% E; modulus of elasticity
% A: area of cross section
% L: length of bar
EA=1e7;
E=30e6;
% GENERATION OF COORDINATES AND CONDUCTIVITIES
nel=6;          % number of elements
nonodes=4;      % number of nodes
ielem=[1 2;1 3;4 2;3 2;4 3;4 1];  % IELEM Matrix
nodeCoor=[0 0;20*12  0;20*12  15*12;0  15*12];% Node Coordinates
xx=nodeCoor(:,1);   % X POSITION FOR EACH NODE
yy=nodeCoor(:,2);   % Y POSITION FOR EACH NODE
% FOR STRUCTURE
ndof=2*nonodes;
D=zeros(ndof,1);   %DISPLACEMENT
force=zeros(ndof,1);% FORCE VECTOR
%EXTERNAL APPLIED FORCE
force(4)=1000;
force(6)=1000;
stiffness=zeros(ndof);
%BOUNDARY CONDITIONS LOCATION
prescribeddof=[1 2 7 8]';

% finding the global stiffness matrix
for i=1:nel
    indice=ielem(i,:);
    elementDof=[indice(1)*2-1 indice(1)*2 indice(2)*2-1 indice(2)*2];
    xa=xx(indice(2))-xx(indice(1));
    ya=yy(indice(2))-yy(indice(1));
    length_element=sqrt(xa^2+ya^2);
    C=xa/length_element;
    S=ya/length_element;
    k1=EA/length_element*[C*C C*S -C*C -C*S; C*S S*S -C*S -S*S;
                          -C*C -C*S C*C C*S;-C*S -S*S C*S S*S];
   stiffness(elementDof,elementDof)=stiffness(elementDof,elementDof)+k1;
```

```
end
for i=1:2
    stiffnessl(1,i)=stiffness(1,i);
    stiffnessl(2,i)=stiffness(2,i);
end
   activeDof=setdiff([1:ndof]',[prescribeddof]);
   U=stiffness(activeDof,activeDof)\force(activeDof);
   displacements=zeros(ndof,1);
   displacements(activeDof)=U;
% stresses at elements
for i=1:nel
indice=ielem(i,:);
elementDof=[ indice(1)*2-1 indice(1)*2 indice(2)*2-1 indice(2)*2] ;
xa=xx(indice(2))-xx(indice(1));
ya=yy(indice(2))-yy(indice(1));
length_element=sqrt(xa*xa+ya*ya);
C=xa/length_element;
S=ya/length_element;
sigma(i)=E/length_element*[-C -S C S]*displacements(elementDof);
end
fprintf(' The Displacement is')
U
fprintf(' The force is')
reactionforce =stiffness*displacements
fprintf('The stress is')
sigma'
```

The Displacement is
U =
0.0320
0.1260
-0.0320
0.1260

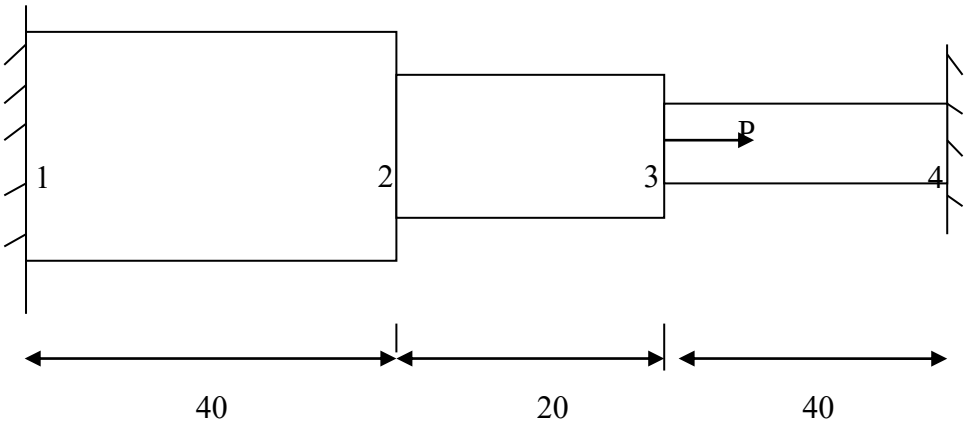
The force is
reactionforce =

*1.0e+03 **
-2.6667
-1.0000
0
1.0000
0
1.0000
2.6667
-1.0000
The stress is
ans =
*1.0e+03 **

4.0000
5.0000
-5.0000
0
-4.0000
0

7.2.4. Strain and Stress of 1D bar Using FEM

Example (60): Find the displacement , the force at the left and the right reactions , and the stress using FEM When the axial load is subjected P=1000N and E=207GPa.




```
clear all
clc
global K u X
format long
n=4; % Number of nodes
for i=1:n
   E(i)=207e9;
end
A(1)=5e-4; A(2)=3e-4;A(3)=1e-4;
L(1)=0.4;L(2)=0.3;L(3)=0.4;
L_t=L(1)+L(2)+L(3);
H=L_t/(n+1);
x(1)=0;
for i=2:n
   x(i)=x(i-1)+H;
end
K=zeros(n,n);
K(1,1)=E(1)*A(1)/L(1);
K(1,2)=-E(1)*A(1)/L(1);
u=zeros(n,1);  % Initialization of the displacement
% Displacement of the  Boundary conditions
u(1)=0;  % Fixed Geometry at LHS
u(n)=0;  % Fixed Geometry at RHS
%---- Build up the  Global Stiffness Matrix -------
for i=2:n-1
K(i,i-1)=-(A(i-1)*E(i-1))/L(i-1);
K(i,i)=(A(i-1)*E(i-1))/L(i-1)+(A(i)*E(i))/L(i);
K(i,i+1)=-(A(i)*E(i))/L(i);  % Internal Forces
end
 K(n,n)=-A(n-1)*E(n-1)/L(n-1);
 K(n,n-1)=A(n-1)*E(n-1)/L(n-1);
 K1=K(2:n-1,2:n-1);
 F=[0  2000]';
 Z= K1\F;
% THE REACTION FORCE AT WALL A AND WALL B
 FA=A(1)*E(1)/L(1)*Z(1);   % in Newton
 FB=A(n-1)*E(n-1)/L(n-1)*Z(2);  % in Newton
% FOR CHEACKING
F_load=FA+FB;
fprintf(' The Displacement is')
fprintf('\n \n  Z,      ')
```

z;
The reaction forces are
F_1= 1379.31N (the left reaction force)
F_4 = 620.6 N (the right reaction force)
The displacements are

$u_2=5.33066799933367\times10^{-6}$m
$u_3=1.19940029985008\times10^{-5}$ m

Appendix

MATLAB Commands and their Meaning

Table: Important MATLAB Commands

Commands Section	MATLAB Commands	Meaning
Matrices Commands	blkdiag ( )	Construct a block diagonal matrix.
	diag ( )	The diagonal of a matrix command.
	trace ( )	To compute the sum of the elements in the main diagonal.
	trill ( )	Return to the lower triangle part of a matrix.
	triu ( )	Return to the upper triangle part of a matrix.
	det ( )	To compute the magnitude of a matrix.
	inv ( ), rank ( )	To compute the inverse of a matrix, and rank of a matrix.
Plotting Commands	plot( )	To plot single plotting or data series plotting
	plotyy( )	To plot a graph with y-axes on both left and right side.
	plotmatrix( )	To plot a matrix.
	fplot ( )	To plot mathematical functions.
	plotv(M,T)	To plot vectors as lines from the origin.
	plotsvec (x,c,M)	To plot vectors w/ different colors.
	grid on	To create a graphical sheet.
	hold on	To hold some data or figures.
	set (gca,…. ,……)	To set color of line, shape, and names.
	bar( )	To plot a bar chart.
	bar3( )	To plot a 3D bar chart.
	bar3h ( )	To plot a 3D bar chart in horizontal.
	barh ( )	To plot a single bar chart in horizontal.
	candle ( )	To plot a candle chart.

	chartft ( )	To plot an interactive display.
	highlow ( )	For high-low plotting.
Conditional and for loop commands	**if** logical expression statements **end**	For single statements (true or false condition)
	if expression statements1 **else** statements2 **end**	For double statements.
	if expression1 statements1 **elseif** expression2 statements2 **end**	Double expressions with two conditions.
	for variable = expression statements **end**	Do loop statement.
Digital and Signal Analysis	svd( )	To show a singular value decomposition.
	eigshow( )	To show the value of a vector from the original.
	ftt( )	To compute discrete Fourier transform of a signal.
	ifft( )	To compute the inverse Fourier transform of a signal.
	fft2 ( )	2D discrete Fourier transform of a signal.
	fftn ( )	nD discrete Fourier transform of a signal.
	ifft2( )	To compute 2D the inverse Fourier transform of a signal.
	ifftn ( )	To compute nD the inverse Fourier transform of a signal.
	abs ( )	Magnitude.
	angle ( )	To compute the phase angle.
	unwrap( )	To put a phase angle in radians.
Interpolations, Integrations, and differential Functions commands	interp1 ( )	To compute one-dimensional interpolation.
	interp2 ( )	To compute two-dimensional interpolation.
	interp3( )	To compute three-dimensional interpolation.
	interpft ( )	To compute one-dimensional interpolation

		using FFT method.
	interpn ( )	To compute n-dimensional interpolation (table lookup).
	rand ( )	To compute a random element.
	EraseMode	To show an animated gragh.
	int( )	To compute limited or unlimited integrations.
	dde23 ( )	To compute initial value problems for delay differential equations with constant delays.
	deval ( )	To evaluate the numerical solution using the output of dde23.
	ddeset ( )	To create/alter the DDE options structure.
	ddeget ( )	To extract properties from options structure created with ddeset.
	dsolve( )	To compute single or several differential functions.

References

[1]. William J. Palm III, Introduction for MATLAB for Engineers, MacGraw Hill 2005.

[2]. Moler,Cleve," Floating points", MATLAB news and notes, Fall,1996.

[3]. Moler et at. Numerical computing with MATLAB, S.I.A.M,1996.

[4]. Karam et al. Complex Chebyshev Approximation for FIR filter design, IEEE trans. On circuits and systems II. March 1995,pages 207-216

[5]. Brain et al. A guide to MATLAB for Beginners and Experienced users, Cambridge, 2nd edition

[7]. Vinay K. et al. Digital signal processing using MATLAB ,McGraw Hill 2009.