DRONACHARYA
Group of Institutions

CONTROL SYSTEM

LABORATORY MANUAL

B.Tech. V Semester

Subject Code: KEE-552

Session: 2023-24, Odd Semester

DRONACHARYA GROUP OF INSTITUTIONS
DEPARTMENT OF EEE
#27 KNOWLEDGE PARK 3

GREATER NOIDA

AFFILATED TO Dr. ABDUL KALAM TECHNICAL UNIVERSITY,
LUCKNOW

Lab Experiments

1.To determine speed-torque characteristics of an AC servomotor.

2. To study

i) Synchro Transmitter characteristics.

ii) Obtain Synchro Transmitter – Receiver output vs input characteristics.

3. To determine response of first order and second order systems for step input for various values of constant 'K' using linear simulator unit and compare theoretical and practical results.

4. To study characteristics of positional error detector by angular displacement of two servo potentiometers. 5. To simulate and compare the response of 2nd order system with and without lead, lag, Lead- Lag compensator / simulate PID controller for transportation lag.

6. To study P, PI and PID temperature controller for an oven and compare their characteristics.

7. To study performance of servo voltage stabilizer at various loads using load bank.

8. To study behavior of separately excited dc motor in open loop and closed loop conditions at various loads.

9. To determine time domain response of a second order system for step input and obtain performance parameters.

10. To convert transfer function of a system into state space form and vice-versa.

11. To plot root locus diagram of an open loop transfer function and determine range of gain 'k' for stability.

12. To plot a Bode diagram of an open loop transfer function. 13. To draw a Nyquist plotof an open loop transfers functions and examine the stability of the closed loop system

SOFTWARE BASED LAB EXPERIMENTS

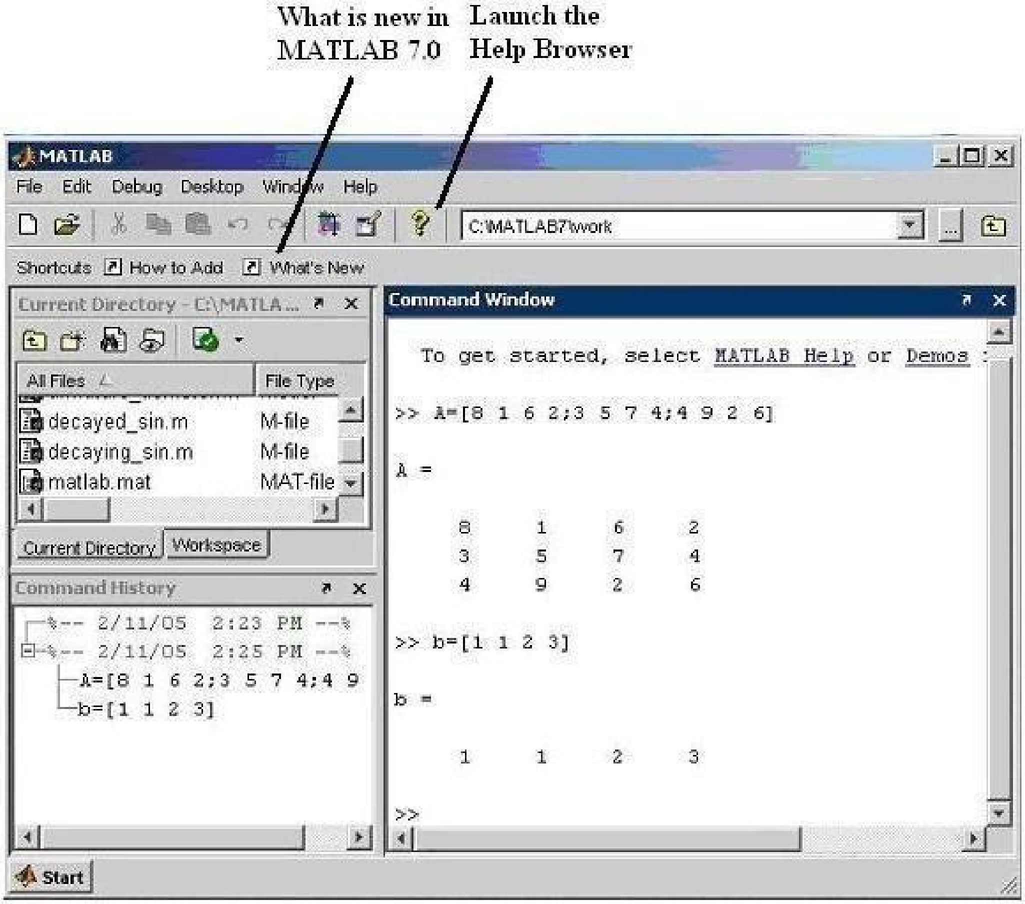
EXPERIMENT 1

Introduction to MATLAB

On the Windows desktop, the installer usually creates a shortcut icon for starting MATLAB; double-clicking on this icon opens MATLAB desktop.
The MATLAB desktop is an integrated development environment for working with MATLAB suite of toolboxes, directories, and programs. We see in Fig. 1 that there are four panels, which represent:

1. Command Window
2. Current Directory
3. Workspace
4. Command History

A particular window can be activated by clicking anywhere inside its borders.

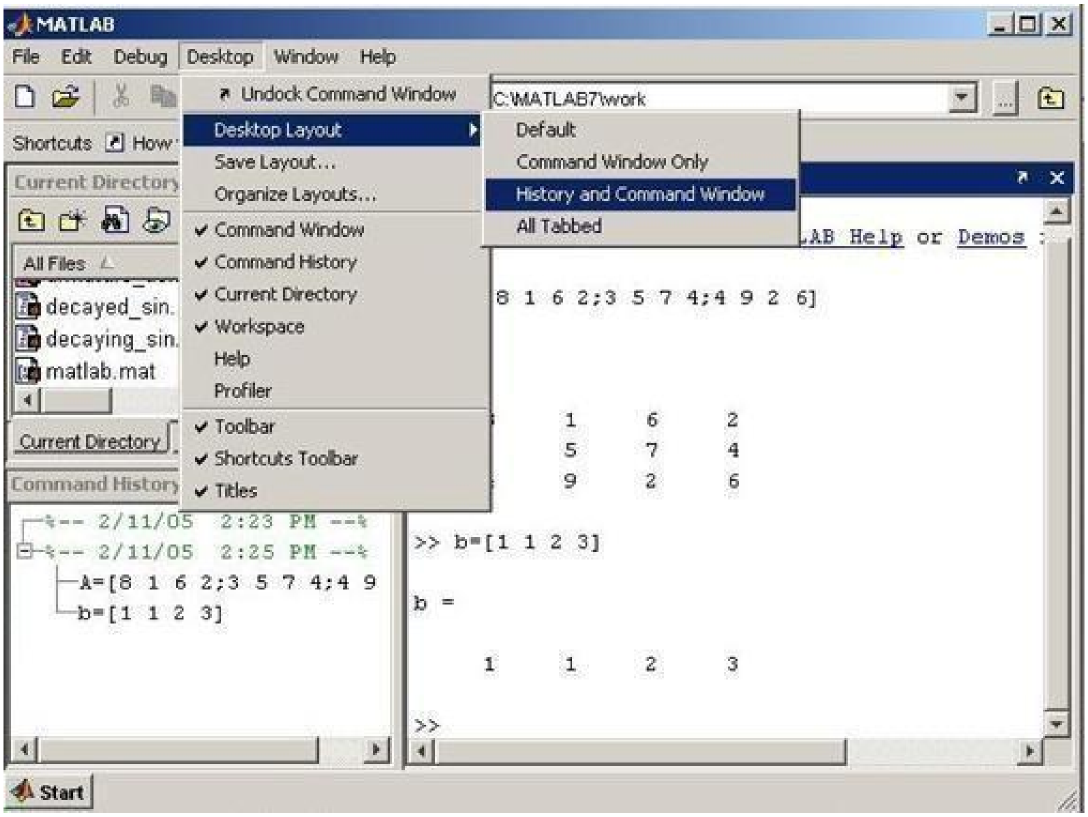


```
%-- 2/11/05  2:23 PM --%
%-- 2/11/05  2:25 PM --%
  A=[8 1 6 2;3 5 7 4;4 9
  b=[1 1 2 3]
```

```
  To get started, select MATLAB Help or Demos :

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

A =

     8     1     6     2
     3     5     7     4
     4     9     2     6

>> b=[1 1 2 3]

b =

     1     1     2     3

>>
```

Fig. 1 *MATLAB Desktop (version 7.0, release 14)*

Desktop layout can be changed by following **Desktop --> Desktop Layout** from the main menu asshown inFig. 2 (Default option gives Fig. 1).



```
%-- 2/11/05  2:23 PM --%
%-- 2/11/05  2:25 PM --%
   A=[8 1 6 2;3 5 7 4;4 9
   b=[1 1 2 3]
```

```
8 1 6 2;3 5 7 4;4 9 2 6]

      1     6     2
      5     7     4
      9     2     6

>> b=[1 1 2 3]

b =

     1     1     2     3

>>
```

Fig. 2 *Changing Desktop Layout to History and Command Window option*

Command Window

We type all our commands in this window at the prompt (>>) and press return (↵) to see the results of our operations. Type the command **ver**on the command prompt to get information about MATLAB version, license number, operating system on which MATLAB is running, JAVA support version, and all installed toolboxes. If MATLAB don't regard to your speed of reading and flush the entire output at once, just type **more on** before supplying command to see one screen of output at atime. Clicking the **What's New** button located on the desktop shortcuts toolbar, opens the release notes for release 14 of MATLAB in Help window. These general release notes give you a quick overview of what products have been updated for Release 14.

Working with *Command Window* allows the user to use MATLAB as a versatile scientific calculator for doing online quick computing. Input information to be processed by the MATLAB commands can be entered in the form of numbers and arrays.

As an example of a simple interactive calculation, suppose that you want to calculate the torque (T)

acting on 0.1 kg mass (m) at 30^0 swing $(\theta)$of the pendulum of length (l) 0.2 m. For small values of swing, T is given by the formula $mgl\theta$. This can be done in the MATLAB command window by typing:

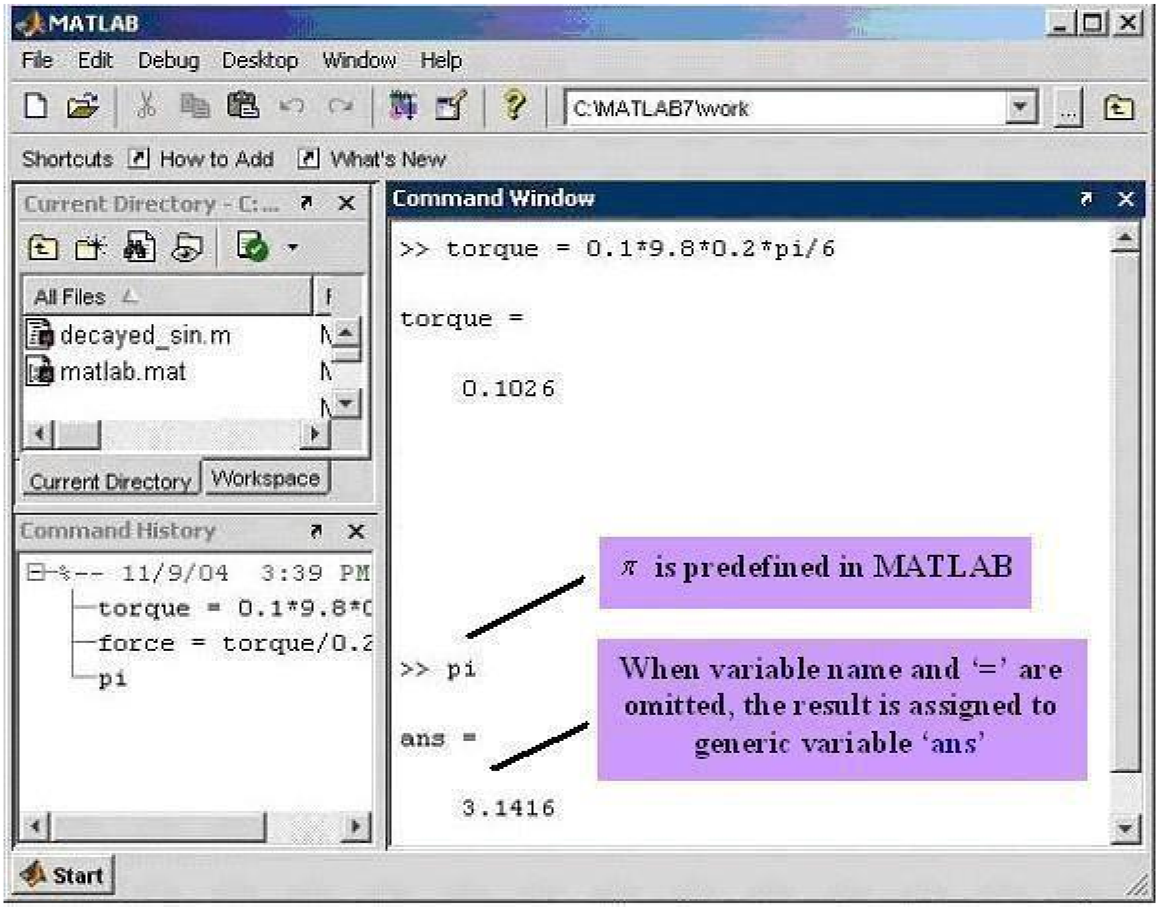
```
>> torque = 0.1*9.8*0.2*pi/6   ↵
```

MATLAB responds to this command by:

torque =

0.1026

MATLAB calculates and stores the answer in a variable **torque** (in fact, a 1×1 array) as soon as the Enter key is pressed. The variable **torque** can be used in further calculations. π is predefined in MATLAB; so we can just use **pi** without declaring it to be 3.14….Command window indicating these operations is shown in Fig. 3.


MATLAB
File Edit Debug Desktop Window Help
C:\MATLAB7\work
Shortcuts How to Add What's New
Current Directory - C:...
All Files
decayed_sin.m
matlab.mat
Current Directory Workspace
Command History
Command Window
Start

```
%-- 11/9/04  3:39 PM
torque = 0.1*9.8*0
force = torque/0.2
pi
```

```
>> torque = 0.1*9.8*0.2*pi/6

torque =

    0.1026



>> pi

ans =

    3.1416
```

π is predefined in MATLAB

When variable name and '=' are omitted, the result is assigned to generic variable 'ans'

Fig. 3 *Command Window for quick scientific calculations (text in colored boxes corresponds toexplanatory notes*).

If any statement is followed by a semicolon,

```
>> m = 0.1; ↵

>> l = 0.2; ↵

>> g = 9.8; ↵
```

the display of the result is suppressed. The assignment of the variable has been carried out even though the display is suppressed by the semicolon. To view the assignment of a variable, simply type the variable name and hit Enter. For example:

```
>>torque=m*g*l*pi/6;   ↵

>>torque ↵

torque  =

0.1026
```

It is often the case that your MATLAB sessions will include intermediate calculations whose display is of little interest. Output display management has the added benefit of increasing the execution speed of the calculations, since displaying screen output takes time.

Variable names begin with a letter and are followed by any number of letters or numbers (including underscore). Keep the name length to 31 characters, since MATLAB remembers only the first 31 characters. Generally we do not use extremely long variable names even though they may be legal MATLAB names. Since MATLAB is case sensitive, the variables **A**and **a**are different.

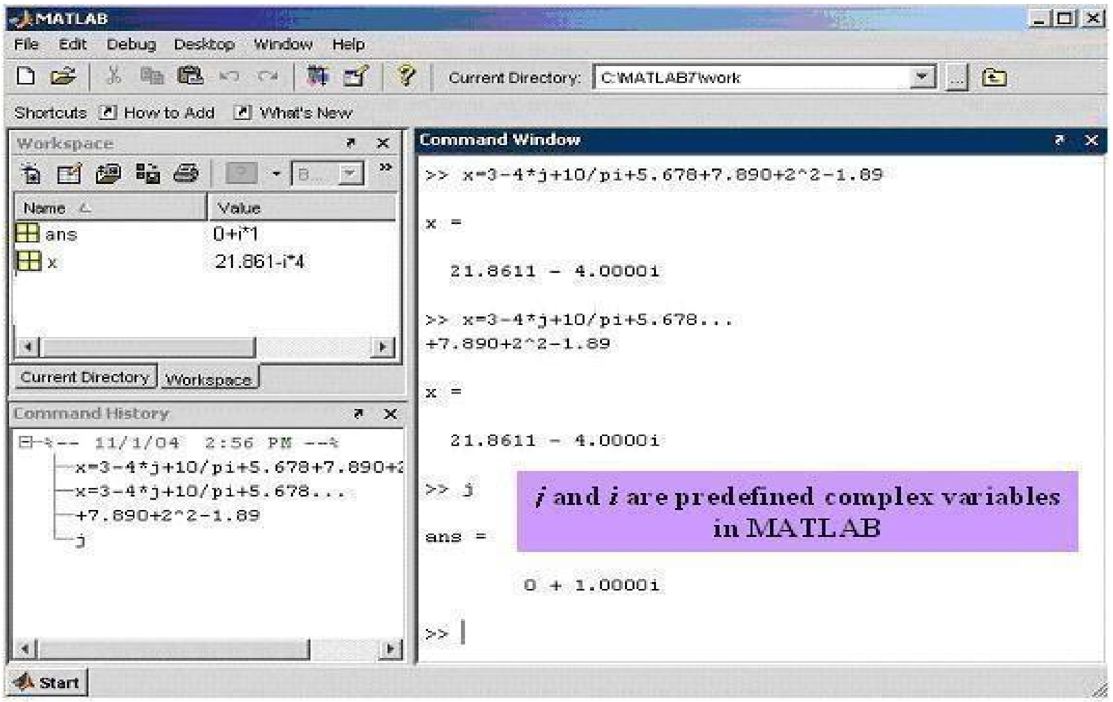
When a statement being entered is too long for one line, use three periods, **...** , followed by to indicate↵that the statement continues on the next line. For example, the following statements are identical (see Fig. 4).

```
>> x=3-4*j+10/pi+5.678+7.890+2^2-1.89

>> x=3-4*j+10/pi+5.678...

+7.890+2^2-1.89
```

+ addition, − subtraction, * multiplication, / division, and ^ power are usual arithmetic operators. The

basic MATLAB trigonometric commands are **sin, cos, tan, cot, sec** and **csc**. The inverses $\sin^{-1}(x), \cos^{-1}(x)$, etc., are calculated by **asin, acos**, etc. The same is true for hyperbolic functions. Some of the trigonometric operations are shown in Fig 5.

Variables **j** = $\sqrt{-1}$ and **i** = $\sqrt{-1}$ are predefined in MATLAB and are used to represent complex numbers.


MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\MATLAB7\work
Shortcuts How to Add What's New
Workspace
Command Window
Current Directory
Workspace
Command History
Start

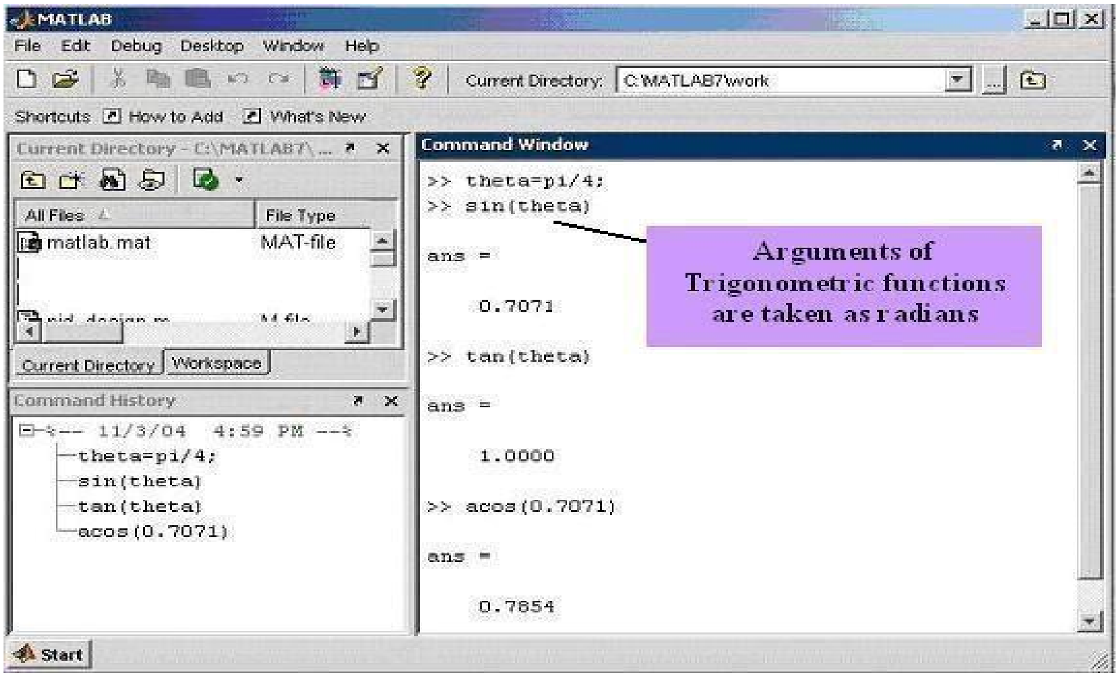
Name	Value
ans	0+i*1
x	21.861-i*4

```
⊟-%-- 11/1/04   2:56 PM --%
   x=3-4*j+10/pi+5.678+7.890+2
   x=3-4*j+10/pi+5.678...
   +7.890+2^2-1.89
   j
```

```
>> x=3-4*j+10/pi+5.678+7.890+2^2-1.89

x =

  21.8611 - 4.0000i

>> x=3-4*j+10/pi+5.678...
+7.890+2^2-1.89

x =

  21.8611 - 4.0000i

>> j

ans =

        0 + 1.0000i

>> |
```

j and *i* are predefined complex variables in MATLAB

Fig. 4 *Command Window with example operations*


MATLAB
File Edit Debug Desktop Window Help
Current Directory: C:\MATLAB7\work
Shortcuts How to Add What's New
Current Directory - C:\MATLAB7\...
Command Window
Current Directory
Workspace
Command History
Start

All Files	File Type
matlab.mat	MAT-file
pid_design.m	M-file

```
⊟-%-- 11/3/04   4:59 PM --%
   theta=pi/4;
   sin(theta)
   tan(theta)
   acos(0.7071)
```

```
>> theta=pi/4;
>> sin(theta)

ans =

    0.7071

>> tan(theta)

ans =

    1.0000

>> acos(0.7071)

ans =

    0.7854
```

Arguments of Trigonometric functions are taken as radians

Fig. 5 *Example trigonometric calculations*

MATLAB representation of complex number $z = \sigma + j\varpi$:

$z = \sigma + j * \omega$ or $z = \sigma + \omega j$

The later case is always interpreted as a complex number, whereas, the former case is a complexnumber in MATLAB only if **j** has not been assigned any prior local value.

MATLAB representation of complex number $z = re^{j\theta}$:
$z = r*exp(j*\theta)$ or $z = r*exp(\theta j)$
$z = r*(cos\theta + j*sin\theta)$ or $z = r*(cos\theta + sin\theta*j)$

In Cartesian form, arithmetic additions on complex numbers are as simple as with real numbers.
Consider two complex numbers $z_1 = \sigma_1 + j\omega_1$ and $z_2 = \sigma_2 + j\omega_2$. Their sum $z = z_1 + z_2$ is given by

$$z = (\sigma_1 + \sigma_2) + j(\omega_1 + \omega_2)$$

For example, two complex numbers $3 + 4j$ and $1.8 + 2j$ can be added in MATLAB as:

```
>> z1=3+4j;  ↵
>> z2=1.8+2j;  ↵
>> z=z1+z2  ↵
z =
4.8000 + 6.0000i
```

Multiplication of two or more complex numbers is easier in polar/complex exponential form. Two complex numbers with radial lengths $r_1 = 2$ and $r_2 = 2.5$ are given with angles $\theta_1 = 35^0$ and $\theta_2 = \frac{\pi}{4}$ rad. We change $\theta_1$to radians to give $\theta_1 = \left(\frac{35}{180}\right)\times\pi$ rad= $0.19\times\pi$ rad. The complex exponential form of their product $z = z_1 z_2$ is given by

$$z = r_1 r_2 e^{j(\theta_1+\theta_2)} = 5e^{j(0.25\pi+0.19\pi)} = 5e^{j0.44\pi}$$

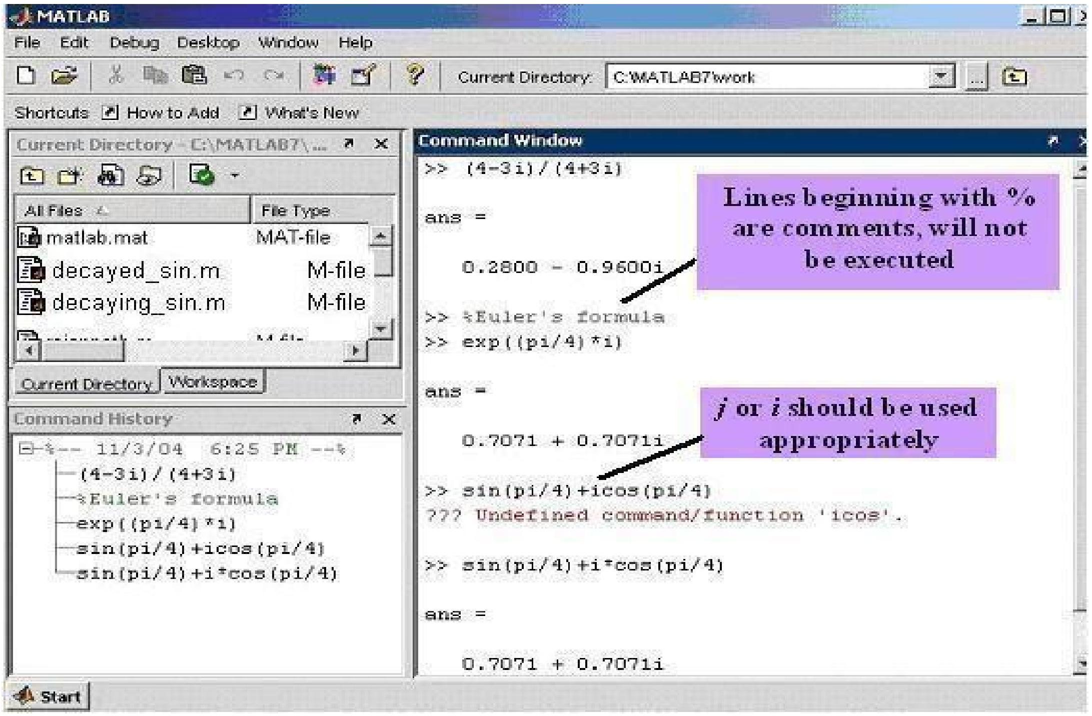
This can be done in MATLAB by:

```
>> theta1=(35/180)*pi;  ↵
>> z1=2*exp(theta1*j);  ↵
>> z2=2.5*exp(0.25*pi*j);  ↵
>> z=z1*z2  ↵
z = 0.8682 - 4.9240j
```

Magnitude and phase of a complex number can be calculated in MATLAB by commands **abs** and **angle.** The following MATLAB session shows the magnitude and phase calculation of complex numbers $z = 5e^{j0.19\pi}$ and $z = \frac{1}{2+\sqrt{3}j}$.

```
>>abs(5*exp(0.19*pi*j))  ↵
ans =
5
>> angle(5*exp(0.19*pi*j))  ↵
ans =
0.5969
>> abs(1/(2+sqrt(3)*j))  ↵
ans =
0.3780
>> angle(1/(2+sqrt(3)*j))  ↵
ans =
-0.7137
```

Some complex numbered calculations are shown in Fig. 6.



```
%-- 11/3/04   6:25 PM --%
(4-3i)/(4+3i)
%Euler's formula
exp((pi/4)*i)
sin(pi/4)+icos(pi/4)
sin(pi/4)+i*cos(pi/4)
```

```
>> (4-3i)/(4+3i)

ans =

   0.2800 - 0.9600i

>> %Euler's formula
>> exp((pi/4)*i)

ans =

   0.7071 + 0.7071i

>> sin(pi/4)+icos(pi/4)
??? Undefined command/function 'icos'.

>> sin(pi/4)+i*cos(pi/4)

ans =

   0.7071 + 0.7071i
```

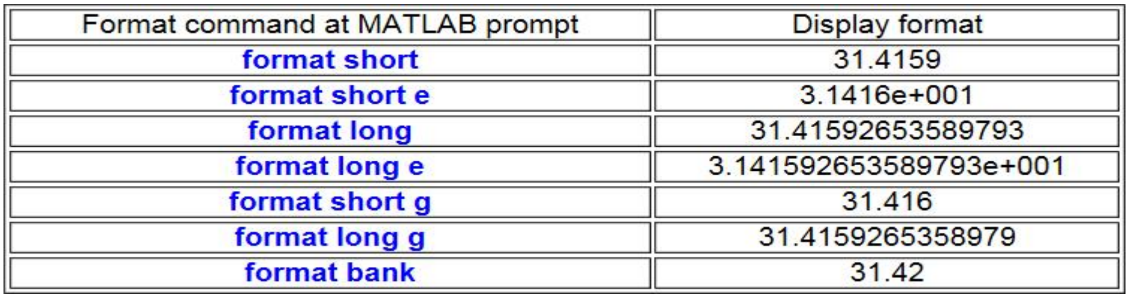
Lines beginning with % are comments, will not be executed

j or *i* should be used appropriately

Fig. 6 *Example complex numbered calculations*

The mathematical quantities $e^x, \log(x)$, and $\ln(x)$ are calculated with **exp(x)**, **log10(x)**,and **log(x)**, respectively.

All computations in MATLAB are performed in *double precision* . The screen output can be displayed in several formats. The default output format contains four digits past the decimal point for nonintegers. This can be changed by using the **format** command. Remember that the **format** command affects only how numbers are displayed, not how MATLAB computes or saves them. See how MATLAB prints 10π in different formats.

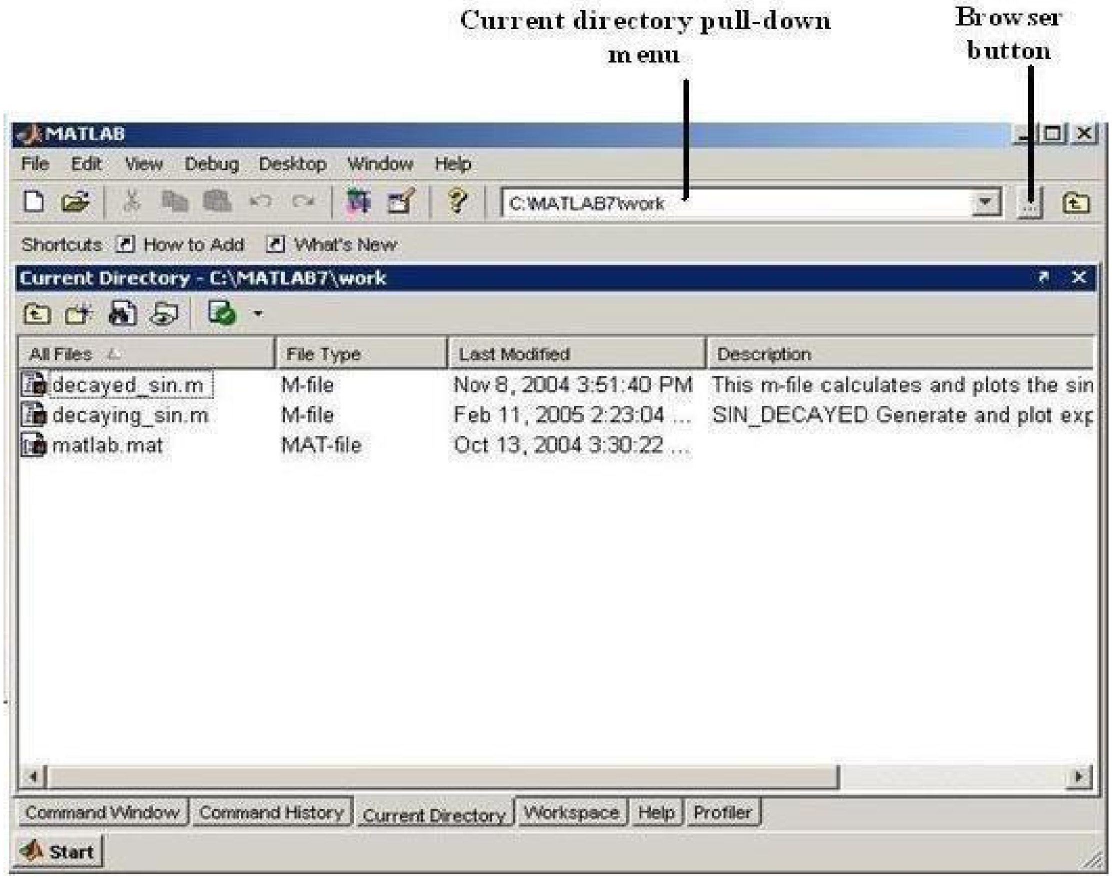


Format command at MATLAB prompt	Display format
format short	31.4159
format short e	3.1416e+001
format long	31.41592653589793
format long e	3.141592653589793e+001
format short g	31.416
format long g	31.4159265358979
format bank	31.42

The following exercise will enable the readers to quickly write various mathematical formulas, interpreting error messages, and syntax related issues.

Current Directory Window

This window (Fig.7) shows the directory, and files within the directory which is in use currently in MATLAB session to run or save our program or data. The default directory is ‘C:\MATLAB7\work'. We can change this directory to the desired one by clicking on the square browser button near the pull-down window.



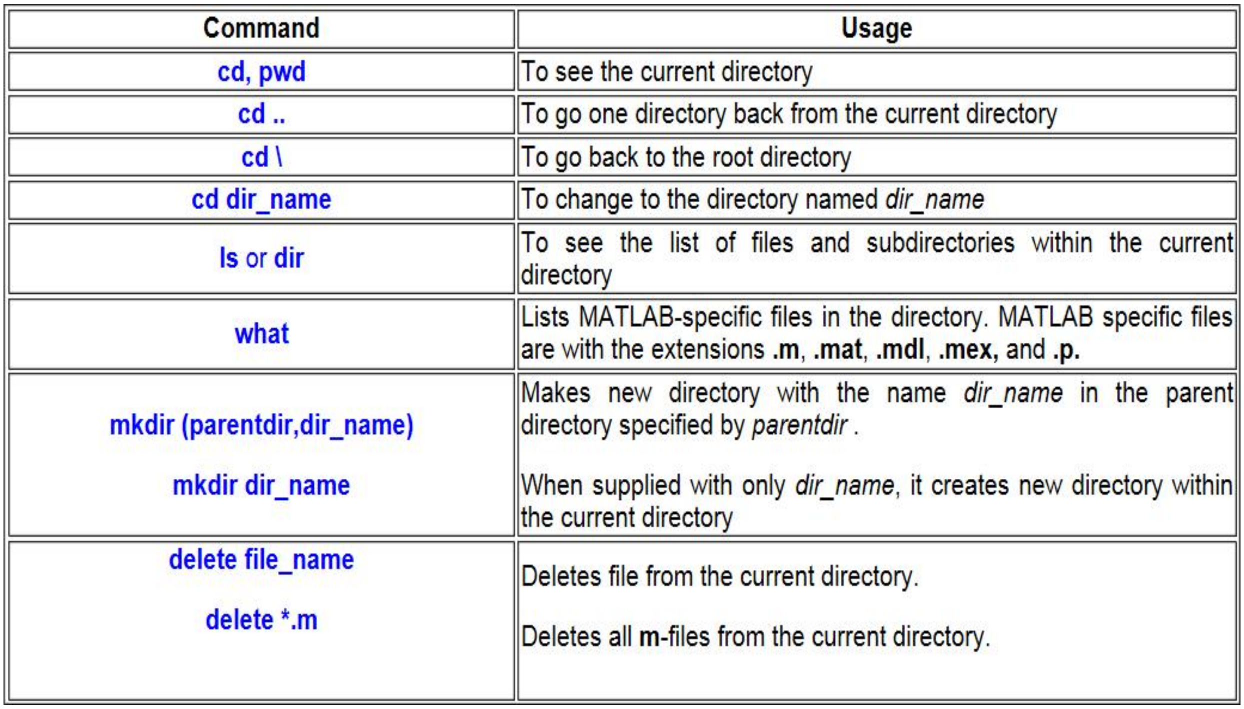
Current directory pull-down menu
Browser button
MATLAB
File Edit View Debug Desktop Window Help
C:\MATLAB7\work
Shortcuts How to Add What's New
Current Directory - C:\MATLAB7\work
Command Window
Command History
Current Directory
Workspace
Help
Profiler
Start

All Files	File Type	Last Modified	Description
decayed_sin.m	M-file	Nov 8, 2004 3:51:40 PM	This m-file calculates and plots the sin
decaying_sin.m	M-file	Feb 11, 2005 2:23:04 ...	SIN_DECAYED Generate and plot exp
matlab.mat	MAT-file	Oct 13, 2004 3:30:22 ...	

Fig. 7 *Current directory window*

One can also use command line options to deal with directory and file related issues. Some useful commands are shown in Table 1.

Table 1

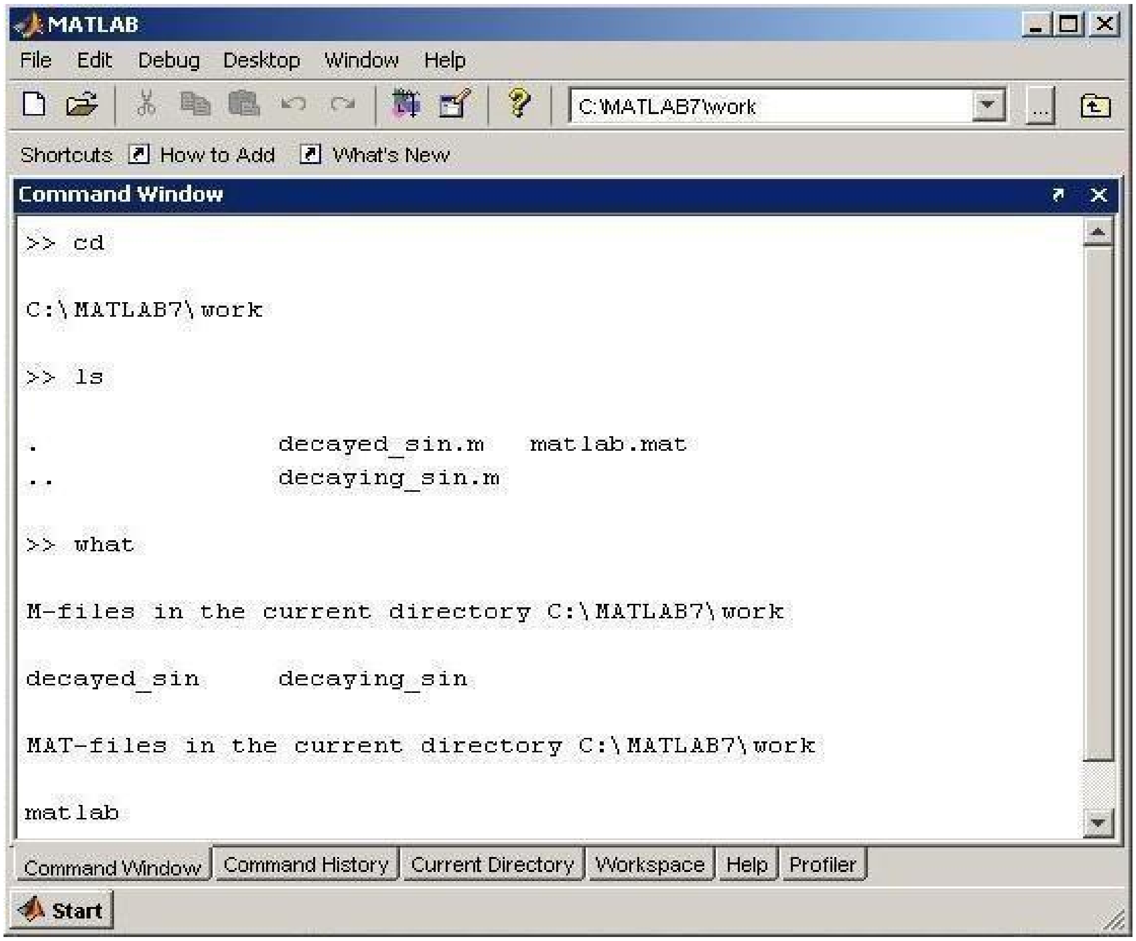


Command	Usage
cd, pwd	To see the current directory
cd ..	To go one directory back from the current directory
**cd **	To go back to the root directory
cd dir_name	To change to the directory named *dir_name*
ls or **dir**	To see the list of files and subdirectories within the current directory
what	Lists MATLAB-specific files in the directory. MATLAB specific files are with the extensions **.m**, **.mat**, **.mdl**, **.mex,** and **.p.**
mkdir (parentdir,dir_name) **mkdir dir_name**	Makes new directory with the name *dir_name* in the parent directory specified by *parentdir* . When supplied with only *dir_name*, it creates new directory within the current directory
delete file_name **delete *.m**	Deletes file from the current directory. Deletes all **m**-files from the current directory.

MATLAB desktop snapshot showing selected commands from Table 1 are shown in Fig. 8.

Workspace

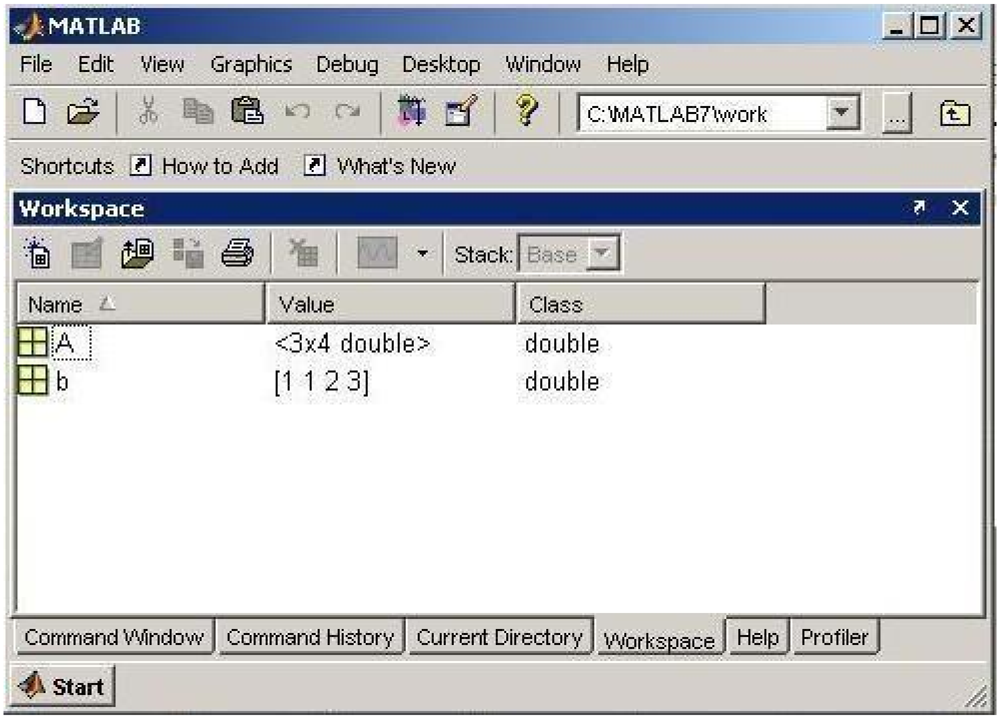
Workspace window shows the name, size, bytes occupied, and class of any variable defined in the MATLAB environment. For example in Fig.9, ‘b' is 1 X 4 size array of data type **double** and thus occupies 32 bytes of memory. Double-clicking on the name of the variable opens the array editor (Fig. 10). We can change the format of the data (e.g., from integer to floating point), size of the array (for example, for variable **A**, from 3 X 4 array to 4 X 4 array) and can also modify the contents of the array.



```
>> cd

C:\MATLAB7\work

>> ls

.                decayed_sin.m   matlab.mat
..               decaying_sin.m

>> what

M-files in the current directory C:\MATLAB7\work

decayed_sin     decaying_sin

MAT-files in the current directory C:\MATLAB7\work

matlab
```

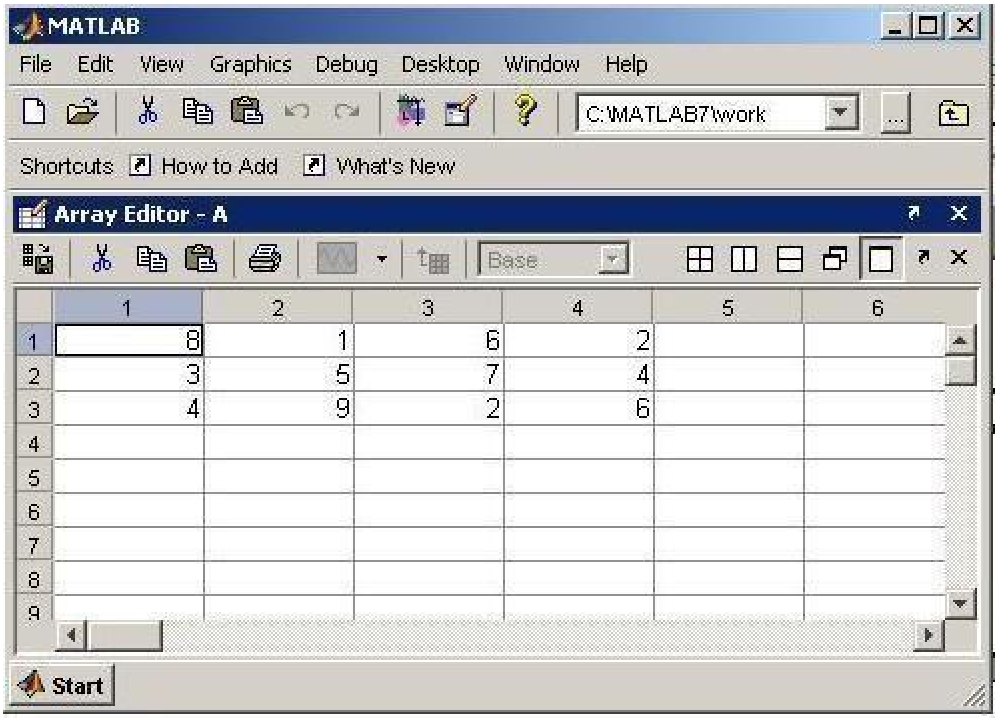
Fig. 8 **Example directory related commands**

If we right-click on the name of a variable, a menu pops up, which shows various operations for the selected variable, such as: open the array editor, save selected variable for future usage, copy, duplicate, and delete the variable, rename the variable, editing the variable, and various plotting options for the selected variable.


MATLAB
File Edit View Graphics Debug Desktop Window Help
C:\MATLAB7\work
Shortcuts How to Add What's New
Workspace
Stack: Base
Command Window Command History Current Directory Workspace Help Profiler
Start

Name	Value	Class
A	<3x4 double>	double
b	[1 1 2 3]	double

Fig. 9 *Entries in the Workspace*

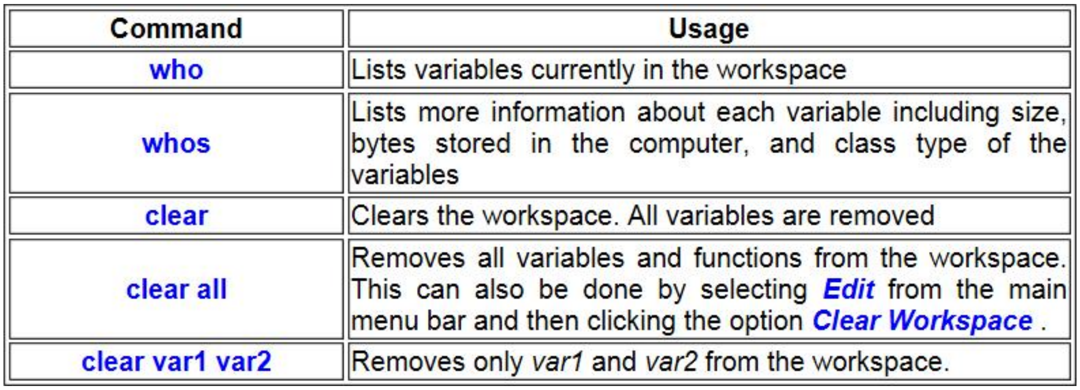

MATLAB
File Edit View Graphics Debug Desktop Window Help
C:\MATLAB7\work
Shortcuts How to Add What's New
Array Editor - A
Base
Start

	1	2	3	4	5	6
1	8	1	6	2		
2	3	5	7	4		
3	4	9	2	6		
4						
5						
6						
7						
8						
9						

Fig. 10 *Array editor window*

Workspace related commands are listed in Table 2.

Table 2



Command	Usage
who	Lists variables currently in the workspace
whos	Lists more information about each variable including size, bytes stored in the computer, and class type of the variables
clear	Clears the workspace. All variables are removed
clear all	Removes all variables and functions from the workspace. This can also be done by selecting ***Edit*** from the main menu bar and then clicking the option ***Clear Workspace*** .
clear var1 var2	Removes only *var1* and *var2* from the workspace.

For example, see the following MATLAB session for the use of **who** and **whos** commands.

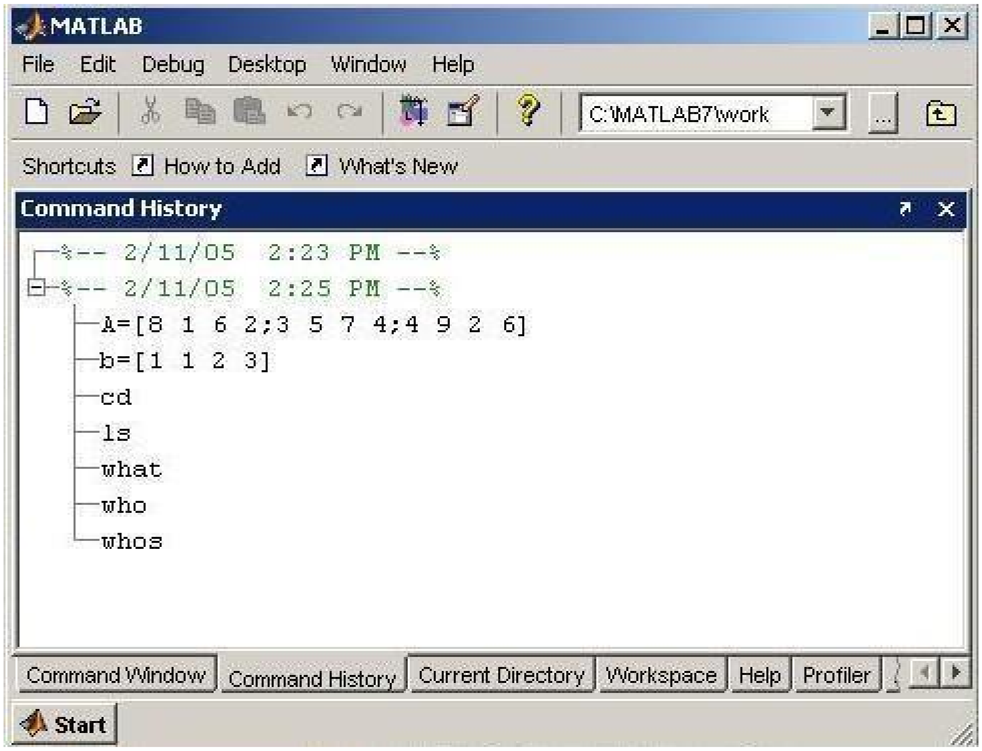
```
>>who
Your variables are:
A b
>>whos
```

Name	Size	Bytes	Class
A	3x4	96	double array
b	1x4	32	double array

```
Grand total is 16 elements using 128 bytes
```

Command History Window

This window (Fig. 11) contains a record of all the commands that we type in the command window. By double-clicking on any command, we can execute it again. It stores commands from one MATLAB session to another, hierarchically arranged in date and time. Commands remain in the list until they are deleted.



```
%-- 2/11/05  2:23 PM --%
%-- 2/11/05  2:25 PM --%
  A=[8 1 6 2;3 5 7 4;4 9 2 6]
  b=[1 1 2 3]
  cd
  ls
  what
  who
  whos
```

Fig. 11 *Command history window*

Commands can also be recalled with the **up-arrow** (↑) key. This helps in editing previous commands.

Selecting one or more commands and right-clicking them, pops up a menu, allowing users to perform various operations such as copy, evaluate, or delete, on the selected set of commands. For example, two commands are being deleted in Fig. 12.

EXPERIMENT NO. 2

OBJECTIVE: To determine time domain response of a second order system for step input and obtain performance parameters

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

MATLAB can conveniently be used to obtain the rise time, peak time, maximum overshoot, and settling time. Consider the system defined by

$$\frac{25}{{}^{2} + 6 \quad + 25}$$

Program

```
%--------This is a MATLAB program to find the rise time, peak time,
% maximum overshoot, and settling time of the second-order system
% and higher-order system -------
% ------- In this example, we assume zeta = 0.6 and wn = 5 -------
num = [25];
den = [1 6 25];
t = 0:0.005:5;
[y,x,t] = step(num,den,t);
r = 1; while y(r) < 1.0001; r = r + 1; end;
rise_time = (r - 1)*0.005
rise_time =
0.5550
[ymax,tp] = max(y);
peak_time = (tp - 1)*0.005
peak_time =
0.7850
max_overshoot = ymax-1
max_overshoot =
0.0948
s = 1001; while y(s) > 0.98 & y(s) < 1.02; s = s - 1; end;
settling_time = (s - 1)*0.005
settling_time =
```

EXPERIMENT NO. 3

OBJECTIVE: Plot bode plot of a open loop transfer function.

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

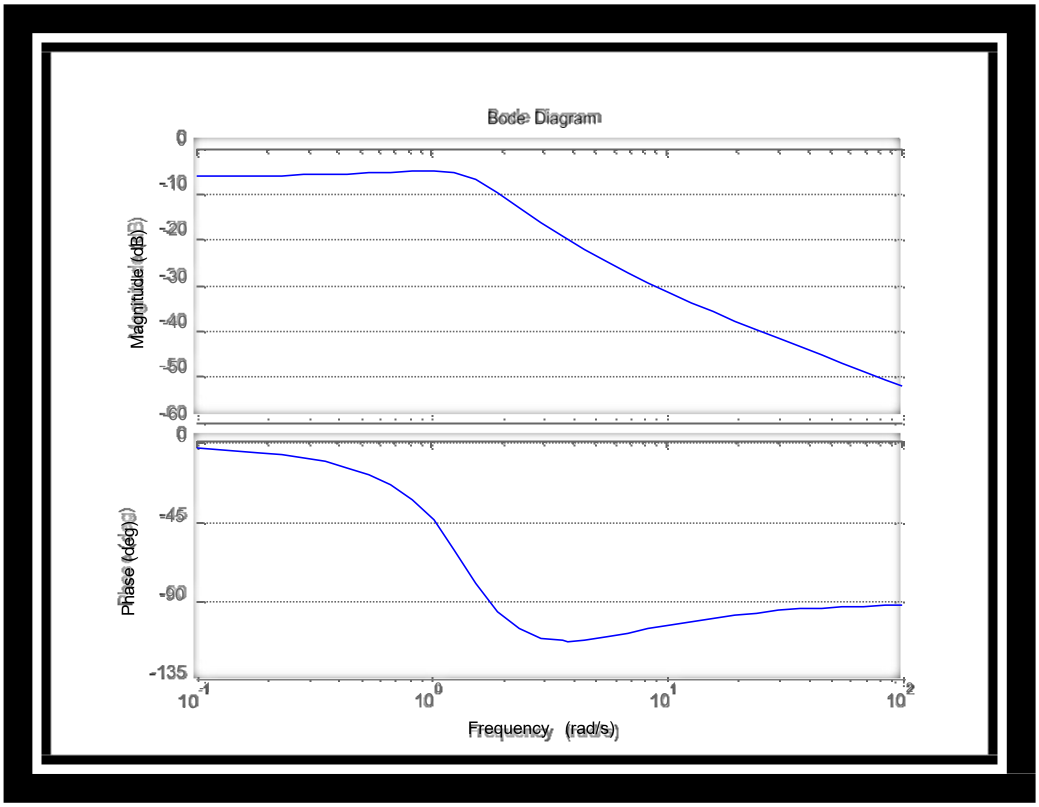
```
%Program to find bode plot %
clc
num= input('Enter the value numerator :')
den= input('Enter the value denominator :')
fun=tf(num,den)
bode(fun)
gridon
```

***Input*:-**

```
Enter the value numerator :[1 4]

num =

1     4

Enter the value denominator :[4 6 8]

den =

4     6     8
```

***Output*:-**

```
fun =

s + 4
----------
4 s^2 + 6 s + 8

Continuous-time transfer function.
```



Command details:-

1. **tf**= Create transfer function model, convert to transfer function model

Syntax:-tf

sys = tf(num,den) sys

= tf(num,den,Ts)sys

= tf(M)

sys = tf(num,den,ltisys)

tfsys = tf(sys)

tfsys = tf(sys)

tfsys = tf(sys, 'measured')

tfsys = tf(sys, 'augmented')

Description

Use tf to create real- or complex-valued transfer function models (TF objects) or to convert state-space or zero-pole-gain models to transfer function form. You can also use tf to createGeneralized state-space (genss) models.

2. **bode =** Bode plot of frequency response, magnitude and phase of frequency response

Syntax:- bode(sys)

bode(sys1,...,sysN)

bode(sys1,PlotStyle1,...,sysN,PlotStyleN)

bode(...,w)

[mag,phase] = bode(sys,w)

[mag,phase,wout] = bode(sys)

[mag,phase,wout,sdmag,sdphase] = bode(sys)

Description

bode(sys) creates a Bode plot of the frequency response of a dynamic system model sys. The plot displays the magnitude (in dB) and phase (in degrees) of the system response as a function of frequency.

When sys is a multi-input, multi-output (MIMO) model, bode produces an array of Bode plots, each plot showing the frequency response of one I/O pair.

bode automatically determines the plot frequency range based on system dynamics.

3. **grid** = Grid lines for 2-D and 3-D plots

Alternatives

To control the presence and appearance of grid lines on a graph, use the Property Editor, one of the plotting tools. For details, see The Property Editor in the MATLAB Graphics documentation.

Syntax

grid on

grid off

grid

grid(axes_handle,...)

grid minor

Description

The grid function turns the current axes' grid lines on and off. grid on adds major grid lines to the current axes. grid off removes major and minor grid lines from the current axes. grid toggles the major grid visibility state grid(axes_handle,...) uses the axes specified by axes_handle instead of the current axes.

EXPERIMENT NO. 4

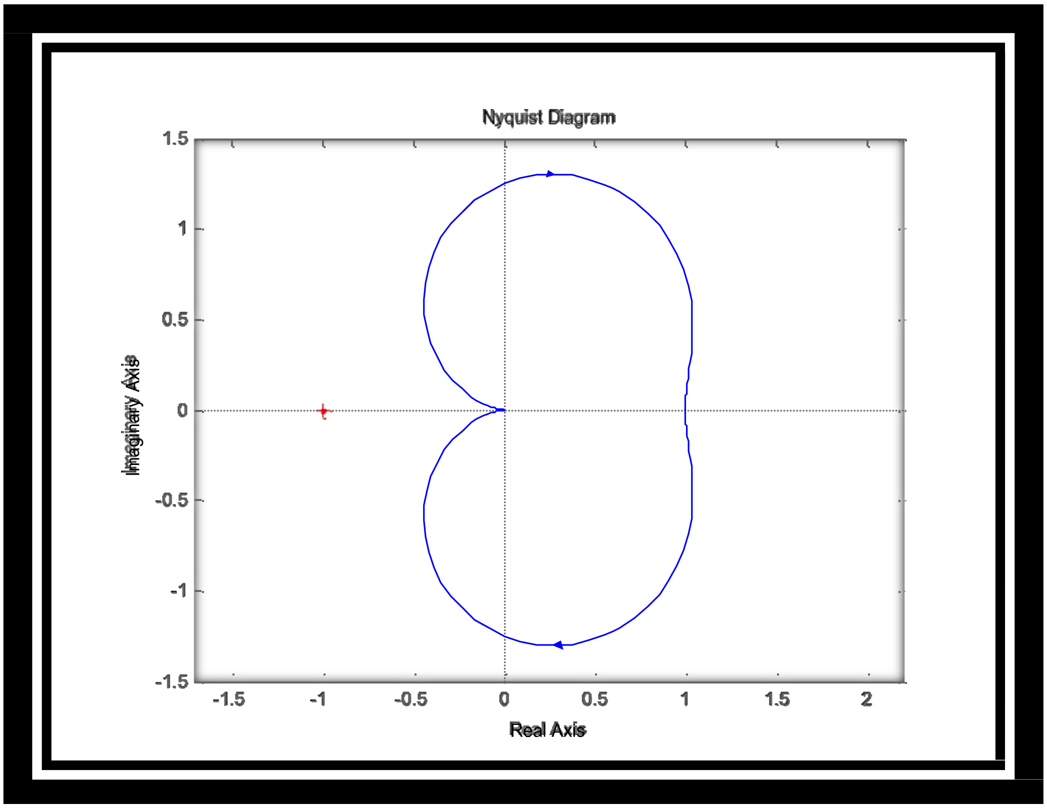
OBJECTIVE: Plot Nyquist plot for given transfer function and to compare their relative stability.

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

```
%Program to find gain and phase margin in Nyquist plot%s =

tf('s');

G = 1/(s^2+0.8*s+1);

nyquist(G);

axisequal;
```

Output:-



Command details:-

4. **nyquist**= Nyquist plot of frequency response

Syntax:-

nyquist(sys)

nyquist(sys,w)

nyquist(sys1,sys2,...,sysN)

nyquist(sys1,sys2,...,sysN,w)

[re,im,w,sdre,sdim] = nyquist(sys)

Description

nyquist creates a Nyquist plot of the frequency response of a dynamic system model. When invoked without left-hand arguments, nyquist produces a Nyquist plot on the screen. Nyquist plots are used to analyze system properties including gain margin, phase margin, and stability.

nyquist(sys) creates a Nyquist plot of a dynamic system sys. This model can be continuous or discrete, and SISO or MIMO. In the MIMO case, nyquist produces an array of Nyquist plots, each plot showing the response of one particular I/O channel. The frequency points are chosen automatically based on the system poles and zeros.

nyquist(sys,w) explicitly specifies the frequency range or frequency points to be used for the plot. To focus on a particular frequency interval, set w = {wmin,wmax}. To use particular frequency points, set w to the vector of desired frequencies. Use logspace to generate logarithmically spaced frequency vectors. Frequencies must be in rad/TimeUnit, where TimeUnit is the time units of the input dynamic system, specified in the TimeUnit property of sys.

nyquist(sys1,sys2,...,sysN) or nyquist(sys1,sys2,...,sysN,w) superimposes the Nyquist plots of several LTI models on a single figure. All systems must have the same number of inputs and outputs, but may otherwise be a mix of continuous- and discrete-time systems. You canalso specify a distinctive color, linestyle, and/or marker for each system plot with the syntax

nyquist(sys1,'PlotStyle1',...,sysN,'PlotStyleN')

See bode for an example.

When invoked with left-hand arguments

[re,im,w] = nyquist(sys)

[re,im] = nyquist(sys,w)

return the real and imaginary parts of the frequency response at the frequencies w (in rad/TimeUnit). re and im are 3-D arrays (see "Arguments" below for details).0

……………………………………………………………………………………..

[re,im,w,sdre,sdim] = nyquist(sys) also returns the standard deviations of re and im for theidentified system sys.

EXPERIMENT NO. 5

OBJECTIVE: PLOT UNIT STEP RESPONSEOF GIVEN TRANSFER FUNCTIONAND FIND PEAK OVERSHOOT, PEAK TIME.

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

```
%Program for unit step response to find rise time and delay time%clc
sys = tf([8 18 32],[1 6 14 24])
S = stepinfo(sys,'RiseTimeLimits',[0.05,0.95])
step(sys)
gridon
```

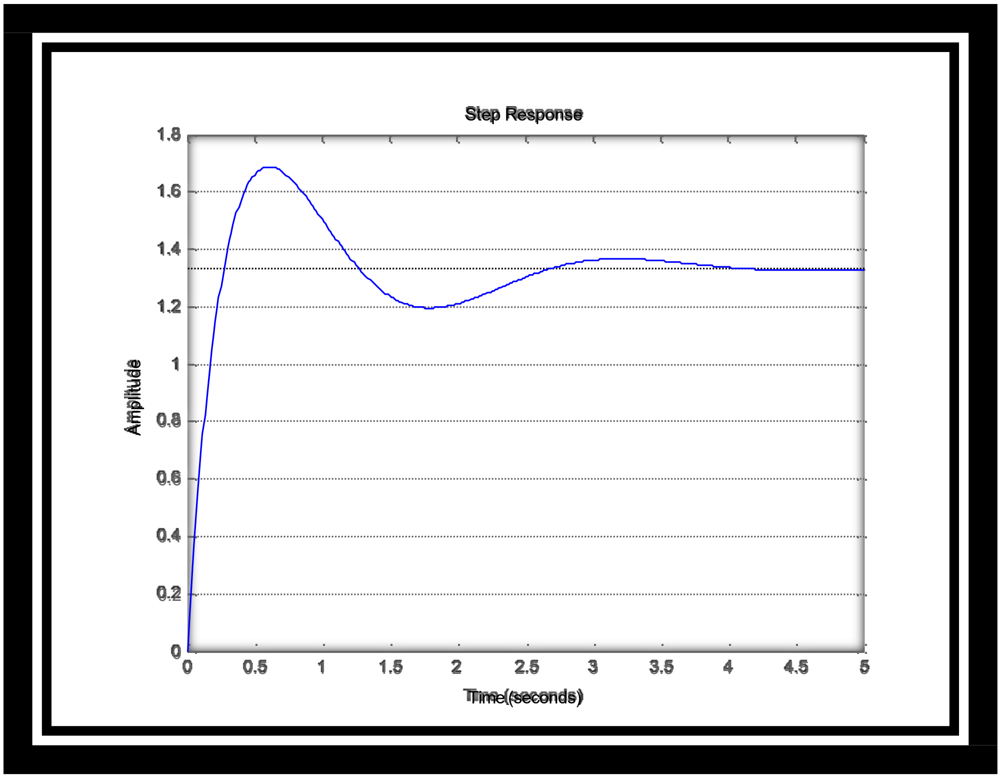
Output:-

```
sys =

8 s^2 + 18 s + 32

---------------

s^3 + 6 s^2 + 14 s + 24 Continuous-

time transfer function.S = RiseTime:

0.2393

SettlingTime: 3.4972

SettlingMin: 1.1956

SettlingMax: 1.6872

Overshoot: 26.5401
```

Undershoot: 0

Peak: 1.6872

PeakTime: 0.6033



Command details:-

5. **stepinfo =** Rise time, settling time, and other step response characteristics

Syntax:-

S = stepinfo(y,t,yfinal)

S = stepinfo(y,t)

S = stepinfo(y)

S = stepinfo(sys)

S = stepinfo(...,'SettlingTimeThreshold',ST)

S = stepinfo(...,'RiseTimeLimits',RT)

Description

S = stepinfo(y,t,yfinal) takes step response data (t,y) and a steady-state value yfinal and returns a structure S containing the following performance indicators:

RiseTime — Rise time

SettlingTime — Settling time

SettlingMin — Minimum value of y once the response has risen

SettlingMax — Maximum value of y once the response has risen

Overshoot — Percentage overshoot (relative to yfinal) Undershoot —

Percentage undershoot

Peak — Peak absolute value of y

PeakTime — Time at which this peak is reached

For SISO responses, t and y are vectors with the same length NS. For systems with NU inputs and NY outputs, you can specify y as an NS-by-NY-by-NU array (see step) and yfinal as an NY-by-NU array. stepinfo then returns a NY-by-NU structure array S of performance metrics for each I/O pair.

S = stepinfo(y,t) uses the last sample value of y as steady-state value yfinal. S = stepinfo(y) assumes t = 1:ns.

S = stepinfo(sys)computes the step response characteristics for an LTI model sys (see tf, zpk, or ss for details).

S = stepinfo(...,'SettlingTimeThreshold',ST) lets you specify the threshold ST used in the settling time calculation. The response has settled when the error |y(t) - yfinal| becomes smaller than a fraction ST of its peak value. The default value is ST=0.02 (2%).

S = stepinfo(...,'RiseTimeLimits',RT) lets you specify the lower and upper thresholds used inthe rise time calculation. By default, the rise time is the time the response takes to rise from10 to 90% of the steady-state value (RT=[0.1 0.9]). Note that RT(2) is also used to calculate SettlingMin and SettlingMax.

EXPERIMENT NO. 6

OBJECTIVE: PLOT THE NYQUIST PLOT FOR GIVEN TRANSFER FUNCTIONAND TO DISCUSS STABILITY, GAIN AND PHSE MARGIN

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

THEORY:

The Nyquist plot of a transfer function, usually the loop transfer function ***GH(z)***, is a mapping of Nyquist contour in z-plane onto ***GH(z)***plane which is in polar coordinates. Thus it is sometimes known as polar plot. Absolute and relative stabilities can be determined from the Nyquist plot using Nyquist stability criterion. Given the loop transfer function ***GH(z)***of a digital control system, the polar plot of ***GH(z)***is obtained by setting $z= e^{j\omega T}$ and varying ω from 0 to ∞.

Nyquist stability criterion: The closed loop transfer function of single input single output digital control system is described by ***M(z) = GH(z)/ 1+ GH(z)***

Characteristic equation ***1+ GH(z)=0***

The stability of the system depends on the roots of the characteristic equation or poles of the system. All the roots of the characteristic equation must lie inside the unit circle for the system to be stable.

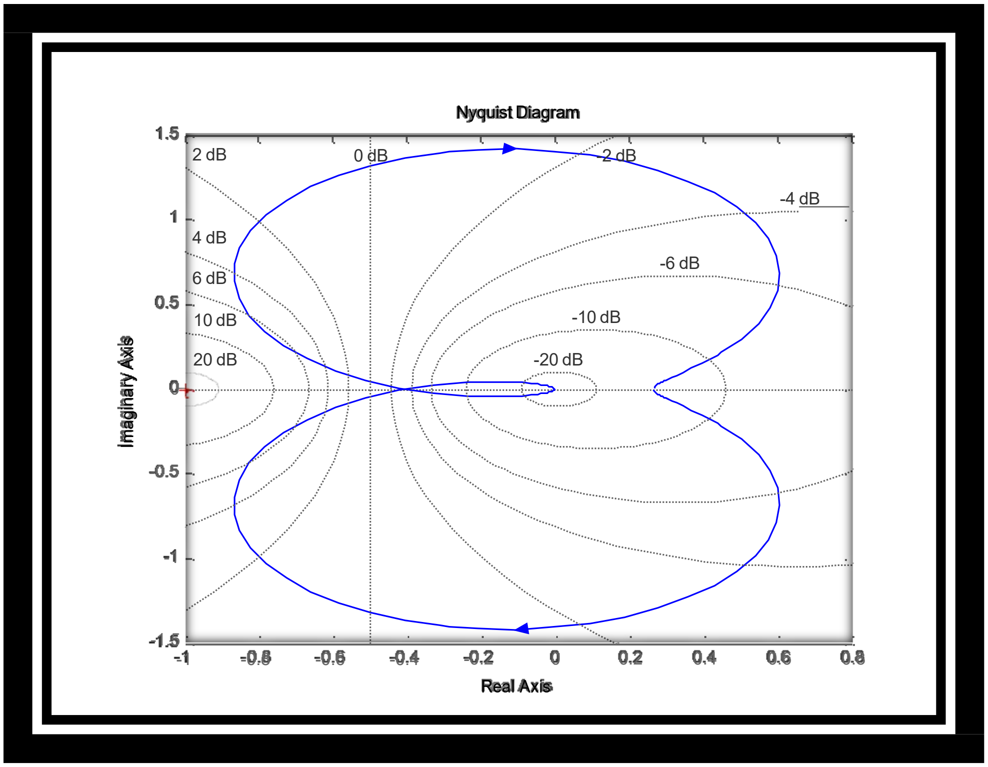
```
%Program to plot the Nyquist plot for given transfer function and to discuss closed loopstability, gain and phase margin. %

clc
num= input('Enter the value numerator :')
den= input('Enter the value denominator :')
time=0.1;
h=tf([num],[den],time) nyquist(h)
[Gm,Pm,Wgm,Wpm]=margin(h)
gridon
```

Input:-

```
Enter the value numerator :[0.04 0.04]
num =

0.0400    0.0400


Enter the value denominator :[1 -1.6 0.9]

den =

1.0000   -1.6000     0.9000
```

Output:-

```
h =

0.04 z + 0.04

- - - - - - - - - - -

z^2 - 1.6 z + 0.9

Sample time: 0.1 seconds

Discrete-time transfer function.

Gm = 2.5001

Pm = 31.6627

Wgm =

7.2274

Wpm =

6.1642
```


Nyquist Diagram
Imaginary Axis
Real Axis
1.5
1
0.5
0
-0.5
-1
-1.5
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
2 dB
0 dB
-2 dB
-4 dB
4 dB
6 dB
-6 dB
10 dB
-10 dB
20 dB
-20 dB

EXPERIMENT NO. 7

OBJECTIVE: To plot root locus diagram of an open loop transfer function and determine range of gain ‘k’ for stability.

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

Plot root loci with a square aspect ratio so that a line with slope 1 is a true 45° line. Choose the region of root-locus plot to be where x and y are the real-axis coordinate and imaginary-axis coordinate, respectively. To set the given plot region on the screen to be square, enter the command v = [-6 6 -6 6]; axis (v); axis('square') With this command, the region of the plot is as specified and a line with slope 1 is at a true 45°, not skewed by the irregular shape of the screen.

For this problem, the denominator is given as a product of first- and second-order terms. So we must multiply these terms to get a polynomial in s. The multiplication of these terms can be done easily by use of the convolution command, as shown next.

Define

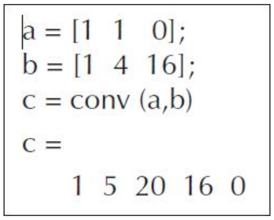
a = s (s + 1): a = [1 1 0]

b = s2 + 4s + 16: b = [1 4 16]

Then we use the following command:

c = conv(a, b)

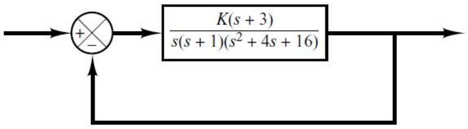
Note that conv(a, b) gives the product of two polynomials a and b. See the following computer output



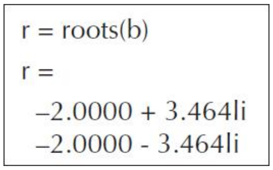
```
a = [1  1   0];
b = [1  4  16];
c = conv (a,b)
c =
      1  5  20  16  0
```

The denominator polynomial is thus found to be

den = [1 5 20 16 0]

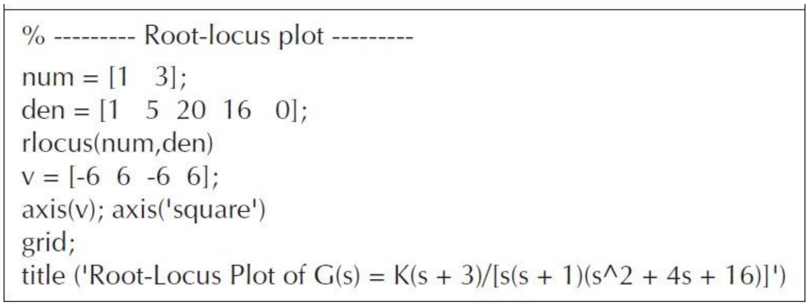


To find the complex-conjugate open-loop poles (the roots of s^2+4s+16=0), we may enter the roots command as follows:



```
r = roots(b)
r =
  –2.0000 + 3.464li
  –2.0000 - 3.464li
```

Thus, the system has the following open-loop zero and open-loop poles:
Open-loop zero: s=–3
Open-loop poles: s=0, s=–1, s=–2 ; j3.4641
MATLAB Program 6–1 will plot the root-locus diagram for this system. The plot is shown in Figure 7.1.



```
% --------- Root-locus plot ---------
num = [1   3];
den = [1   5  20  16   0];
rlocus(num,den)
v = [-6  6  -6  6];
axis(v); axis('square')
grid;
title ('Root-Locus Plot of G(s) = K(s + 3)/[s(s + 1)(s^2 + 4s + 16)]')
```

Note that in MATLAB Program above, instead of den = [1 5 20 16 0] we may enter

den = conv ([1 1 0], [1 4 16]). The results are the same.

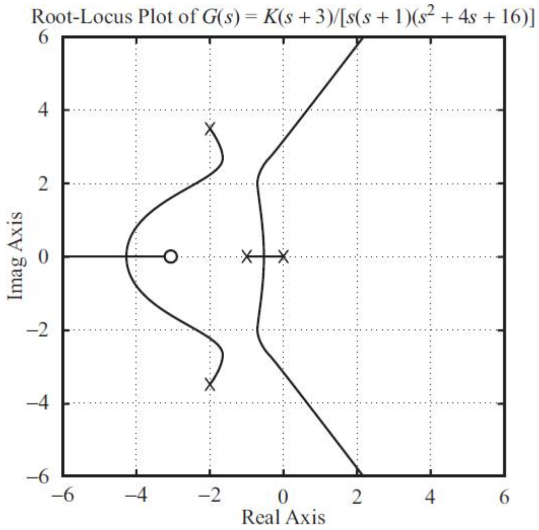


Figure 7.1

EXPERIMENT NO. 8

OBJECTIVE: TO CONVERT TRANSFER FUNCTION OF A SYSTEM INTO STATE SPACE FORM AND VICEVERSA

APPARATUS REQUIRED:

S. No.	Name of the software	Specification	Quantity
1.	MATLAB SOFTWARE	R2012a	

PROGRAM 1:

```
a= input ( “Enter the values of a matrix” );
b= input ( “Enter the values of b matrix” );
c= input ( “Enter the values of c matrix” );
d= input ( “Enter the values of d matrix” );
[num , den] = SS2 tf (a,b,c,d,1)
S1=tf (num(1, : ) , den );
S2=tf (num(2, : ) , den );
[num1 , den1 ] = SS2 tf (a,b,c,d,2);
S3=tf (num1 (1, : ) , den1 );
S4=tf (num1 (2, : ) , den1 );
DISP [S1,S2,S3,S4 ];
```

PROGRAM 2:

```
Num = input (“Enter numerator polynomial values in the form of matrix
array” );
den1 = input (“Enter denominator 1 values” );
den2 = input (“Enter denominator 2 values” );
den = conv (den1,den2);
H = tf (num,den);
P = SS(H);
[a,b,c,d] = SS data(P);
```

HARDWARE BASED LAB EXPERIMENTS

EXPERIMENT NO. 9

OBJECTIVE: TO DETERMINE SPEED TORQUE CHARACTERSTICS OF AN AC SERVOMOTOR

APPARATUS:

Sr. No.	Item	Type	Range	Quantity
1	A.C Servomotor Kit			1
2	Multimeter			1
3	Patch cords			

THEORY:

An A.C servomotor is basically a two phase induction motor except for certain special design features. A two phase induction motor consisting of two stator windings oriented 90 degrees electrically apart in space and excited by a.c voltages which differ in time phase by 90 degrees. Generally voltages of equal magnitude and 90 degrees phase difference are applied to the two stator phases thus making their respective fields 90 degrees apart in both time and space at synchronous speed.

The stator windings are excited by voltages of equal r.m.s magnitude & 900 phase difference these currents give rise to a rotating magnetic field of constant magnitude the direction of rotation depends of on the phase relationship of the two currents (or voltages) the exciting currents produce a clock wise rotating magnetic field & phase shift of 1800 in i1 will produce an anti clock wise rotating magnetic field. Due to the interaction of stator &rotor flux, a mechanical force (or torque) is developed on the rotor & so the rotor starts moving in same direction as that of rotating magnetic field.

PROCEDURE:

1. Ensure that all control pots are in minimum position and motor brake drum is free to rotate.
2. Switch on the power supply and adjust the control voltage to a certain value and fix it.
3. Note down the No-load speed 4. Now load the motor by tightening the wheels of spring balance. Note down speed, S1, & S2 values.
5. Repeat step 4 by increasing the load for 4 different set of loads.
6. Now remove the brake on the motor by loosening the wheels
7. Fix the control voltage at 4 different values and for each value perform steps 3,4,5.
8. Calculate torque and plot Torque VS speed characteristic for different control voltages.

OBSERVATIONS:

Sr. No.	SPEED(rpm)	E_b(VOLTS)

Sr. No.	Vc (V)	N(rpm)	Sb	Sz	Torque

PRECAUTION:

1. Before switching on the speed, adjust & had control pot should always at the most anticlockwise position & should be operated in gentle position.
2. Loose connections are to be avoided.
3. Circuit connections should not be made while power is on.
4. Readings of meters must be taken without parallax error.

RESULT: Speed torque characteristics of AC Servomotor are observed.

ASSESMENT QUESTIONS:

1. What is a servomotor?
2. What are the applications of servomotor?
3. How can we get the feed back characteristics of A.C Servomotor?
4. How do you load the A.C Servomotor?
5. Why a servomotor should not be switched on load?
6. How can a A.C servomotor be controlled?

EXPERIMENT NO. 10

OBJECTIVE: TO DETERMINE RESPONSE OF FIRST ORDER AND SECOND ORDER SYSTEMS FOR STEP INPUT FOR VARIOUS VALUES OF CONSTANT K USING LINEAR SIMULATOR AND COMPARE THEORITICAL AND PRACTICAL RESULTS

APPARATUS:

Sr. No.	Item	Type	Range	Quantity
1	Time Response Kit			1
2	Cathode Ray Oscilloscope			1
3	Patch cords			

THEORY:

When a system is given an excitation (INPUT), there is a response (OUTPUT).
This response varies with time, and is called the time response.
Time response is divided into two parts.
1) Transient Response - It is the part of response that goes to zero as the time increases.
2) Steady state Response - That part of the response that remains after the transients have died out.

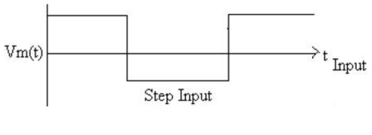
TIME RESPONSE SPECIFICATIONS:

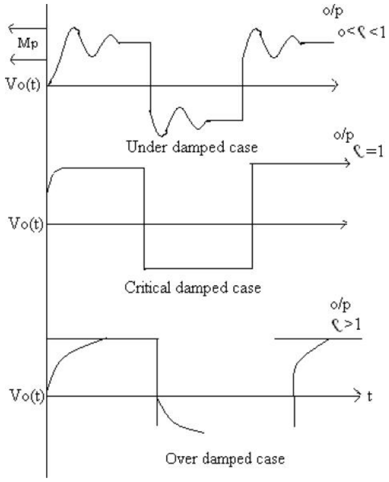
1. Delay Time (td): - It is the value of time required for the response to reach 50% of final value in first attempt.
2. Rise Rime (tr):- It is the time required for the response to rise from 10% to 90% of the final value for the over damped system and 0 to 100% value for under damped system.
3. Peak Time (tp) : - It is the time required for the response to reach the peak of the time response or the peak overshoot.
4. Peak Overshoot (MP): - It indicates the normalized difference between the time response peak & steady state output. It is defined as peak percent overshoot.

$$\% \text{M P} = \frac{C(tp) - C(00)}{C(00)} \times 100$$

Where C (tp) is output at t = tp & C (00) is steady state output.
5. Settling time (ts): - It is the time required for the response to reach and stay within a specific tolerance band (usually 2 to 5%) of its final value.
6.Steady State Error: - It indicates the error between the actual output & desired output as t tends to infinity.

$E_{ss} = lim_{t\to\infty}[r(t) - c(t)]$





OBSERVATIONS:

	R	L	C	$\zeta = R\sqrt{\frac{C}{L}}$	$\omega_n = \frac{1}{\sqrt{LC}}$	ω_d
SET1 **0<ζ<1**						
SET2 **ζ=1**						
SET3 **ζ>1**						

PROCEDURE :

1. Make connections as shown in circuit diagram.
2. Connect repeated step input to RLC circuit.
3. Make power on to the unit.
4. Connect C.R.O. at the output and adjust C.R.O. to get stable pattern on C.R.O.
5. Vary R by potentiometer and for a given set of values of L & C,
note down R for critically damped response
6. Vary R to obtain under damped response and measure R value, time response specifications.
7. Plot the same response on Graph paper.

NOTE: Actual value of R is R + resistance of Inductance

PRECAUTIONS :

1. Loose connections are to be avoided.
2. Circuit connections should not be made while power is on.
3. Readings of meters must be taken without parallax error.

RESULT: Time Response specifications are obtained

ASSESMENT QUESTIONS :

1. What is delay time?
2. What is rise time?
3. What is peak time?
4. What is peak overshoot?

EXPERIMENT NO. 11

OBJECTIVE: TO STUDY (i) SYNCHRO TRANSMITTER CHARACTERSTICS (ii) OBTAIN SYNCHRO TRANSMITTER – RECEIVER CHARACTERSTICS

APPARATUS:

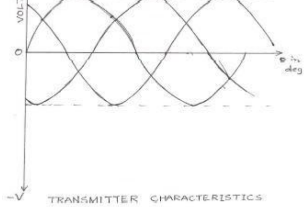
Sr. No.	Item	Type	Range	Quantity
1	Synchro's Kit			1
2	Multimeter			1
3	Patch cords			

THEORY :

The term synchro is a generic name for a family of inductive devices which works on the principle of a rotating transformer basically they are electro- mechanical devices or electromagnetic transducers which produces an o/p voltage depending upon angular position of the rotor a synchro system is formed by inter connection. The basic synchro is usually called asynchro transmitter. Its construction is similar to that of a three phase alternator. The stator (stationary member) is of laminated silicon steel and is slotted to accommodate a balanced three phase winding which is usually of concentric coil type f (three identical coils are placed in the stator with their axis 120 degree apart) and is Y connected. The rotor is a dumb bell shape type in construction and wound with a concentric coil. An a.c. voltage is applied to the rotor winding through slip rings. The system set up is consists of synchro transmitter and synchro receiver on a single rigid panel housed in MS cabinet plates, Rotor position of Tx and Rx is marked by graduated angular scale with pointer arrangement. AC input excitation supply for rotor of Tx andRx is provided internally and panel switches are provided to make it On and Off independently. Test points for Tx and Rx stator and Rotor points are provided on panel.

PROCEDURE :

1. Connect the mains supply to the unit.
2. Varying rotor position of Transmitter in steps of 300 note down VS1S2, VS2S3, VS3S1 readings.
3. Connect S1, S2, S3 of Transmitter to S1, S2, S3 of receiver respectively.
4. Move the rotor of synchro Transmitter in steps of 30 degrees and note down the change in Receiver rotor.
5. Enter the input angular position Transmitter rotor and output angular position of Receiver rotor in tabular form and plot a graph.
6. Plot graph settings: a) (Vs1s2, Vs2s3, Vs3s1) Vs θt and b) θr Vs θt

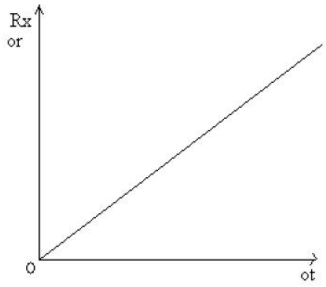


TRANSMITTER CHARACTERISTICS

OBSERVATIONS:

Sr. No.	TRANSMITTER	RECEIVER

GRAPHS:



Transmitter Rotor Position Vs Receiver Rotor Position

RESULT :

The characteristics of synchros are obtained from the graphs drawn.

PRECAUTIONS :

1 Loose connections are to be avoided.
2 Circuit connections should not be made while power is on.
3 Readings of meters must be taken without parallax error
4 Handle the pointers in gentle manner.
5 Do not attempt to pull out the pointers.
6 Do not short rotor or stator terminals.

ASSESMENT QUESTIONS :

1. What is a synchro?
2. What is the use of synchro?
3. What is the constructional difference between synchro transmitter & synchro receiver?
4. What is the relation between a synchro & a transformer?
5. Where do we get maximum e m f in a synchro?

EXPERIMENT NO. 12

OBJECTIVE: TO STUDY P, PI AND PID TEMPERATURE CONTROLLER FOR AN OVEN AND COMPARE THE CHARACTERSTICS

APPARATUS:

Sr. No.	Item	Type	Range	Quantity
1	P, PD,PI, PID Controller Kit			1
2	Cathode Ray Oscilloscope			1
3	Patch cords			

THEORY :

INTEGRAL ACTION (I ACTION):

This is proportional to the integral of error, thus is dependent on the history of the out put and not only the current value. thus, the value of integral action contribution is not as obvious from the apparent output plot as in case of proportional contribution. The effect of 1 action is to reduce average value of the steady state error. Note that if steady state error is not constant(for example steady state error of sinusoidal + some constant value) then PI controller reduces the dc part (the average value) to zero and now sinusoidal variation about zero dc value remains. The disadvantage of integral action is that it increases system's setting time. A typical output plot with PI action (proportional+ integral) is shown below. For this Kd is set to zero.

DERIVATIVE ACTION (D ACTION):

The third term of PID controller transfer equation contributes proportional to the time derivative of error (or rate of change of error). This part is introduced to compensate against the output variations with respect to time. If steady state error is not constant (for example steady state error of sinusoidal+ some constant value) then PD controller reduces the varying part to zero and the constant steady error of average value remains. The D action depends on few past and the current error value and not the complete history as in case of integral action. The contribution is high - speed changes in output that may occur because of various reasons. This also responds to the output variations that may rise due to noise in the sensing, conditioning and feedback network which is unnecessary and hence these parts must be precision type.

PROCEDURE :

1. Connect the circuit as per the block diagram shown
2. Switch ON the main power supply and select Input square wave signal of certain amplitude and frequency. Note the input waveform from C.R.O

P-Controller:

3. Switch ON Proportional stage keeping integral, derivative stage in OFF position.
4. Vary the proportional gain KP for 3 to 4 values and note down the Output C(S) waveforms for each value.

PD Controller:

5. Switch ON derivative stage keeping integeral stage OFF and fix KP at certain value.
6. Vary Derivative gain Dt for 3 to 4 values and Note down the Output C(S) waveforms for each value

PI Controller:

7. Switch OFF Dt stage, Switch ON Integral stage and Fix KP at certain value.
8. Vary Integral gain Int for 3 to 4 values and note down the output C(S) waveforms for each value.

PID Controller:

9. Switch ON all the three controllers fix derivative, Integral controller at certain values.
10. Vary proportional controller gain KP and note down output C(S)waveforms.
11. Compare the output waveforms for different controller gains and write remarks on the effects of various controller gains on 2nd order system time response specifications.

RESULT :

The effect of P, PI, PD, PID controllers are studied.

PRECAUTIONS :

1. Loose connections are to be avoided.
2. Circuit connections should not be made while power is on.
3. Readings of meters must be taken without parallax error.

ASSESMENT QUESTIONS:

1. What is the use of a controller in control system?
2. What is the use of proportionality controller?
3. Why is integral controller used?
4. Why is differential controller used?
5. How can you rectify an error using controller?

EXPERIMENT NO. 13

OBJECTIVE: TO SIMULATE AND COMPARE THE RESPONSE OF 2nd ORDER SYSTEMS WITH AND WITHOUT LEAD, LAG, LEAD-LAG COMPENSATORS/ SIMULATE PID CONTROLLER FOR TRANSPORTATION LAG

APPARATUS:

Sr. No.	Item	Type	Range	Quantity
1	LEAD-LAG Kit			**1**
2	Multimeter			**1**
3	Patch cords			

Theory :

Passive electric components –resistors, capacitors and inductors are used for implementation of a compensator. However the inductor is a very bulky component at low frequencies, passive networks are made up of only resistors and capacitors are used in practice. Condition for reliability of transfer function D(s) with passive resistor capacitor (RC) networks is that all finite poles of D(s) may lie anywhere in s-plane. By taking an operational amplifier to the output of passive RC network, it is possible to realize a specified gain of the compensator.

STUDY OF SIMPLE PHASE LEAD NETWORK:

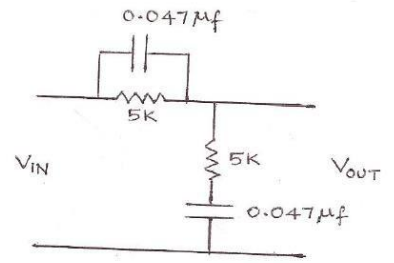
Lead compensator has a single pole and a single zero with the pole lying to the left of the zero on the negative real axis of the complex plane. The first order compensator is a lead compensator if $p>z$.

STUDY OF SIMPLE PHASE LAG NETWORK:

A lag compensator has a single pole and a single zero with the pole lying to the right of zero on negative real axis of the complex plane. The first order compensator is therefore a lag compensator if $p<z$.

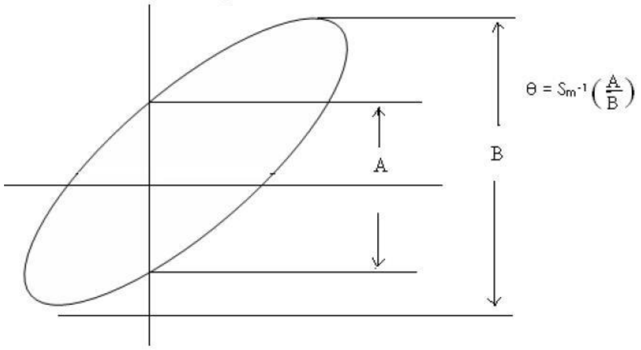
PROCEDURE:

1. Connect the circuit as per circuit diagram for phase lag network.
2. Switch ON the power supply to function generator and adjust input sine wave to say 5V.
3. C.R.O. is to be connected in X-Y made for phase displacement measurement connect X-input of CRO to input of lag network and Y- input of C.R.O to output of lag network.
4. Slowly vary the input frequency and note down the output magnitude to calculate gain $\left|T (jw)\right|$ and change in phase shift T(jw) using lassjous figures concept
5. Calculate the theoretical values of [T (jw)] and θ from the formula given above.
6. Plot the graphs of [T(jw)] and θ adjusting frequency. Find the common frequency.
7. Repeat the above procedure for lead network, lead compensator, lag compensator.
8. Draw bode plot for all the system.

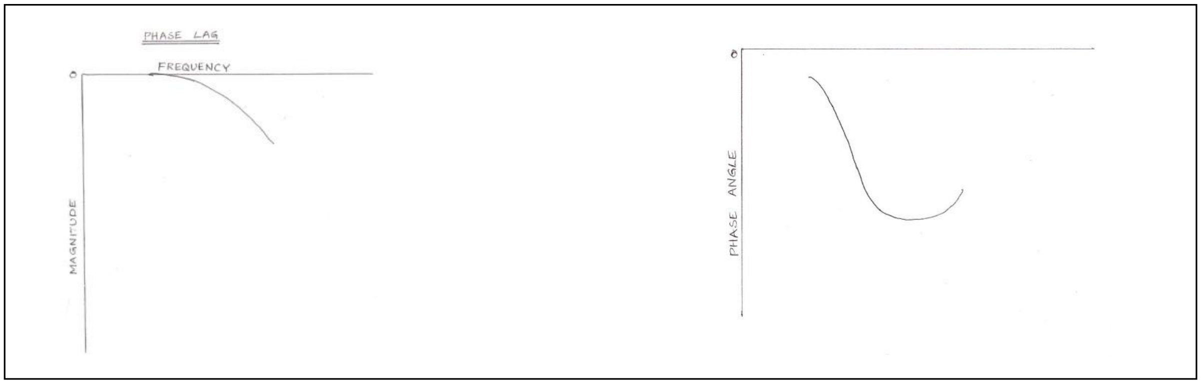


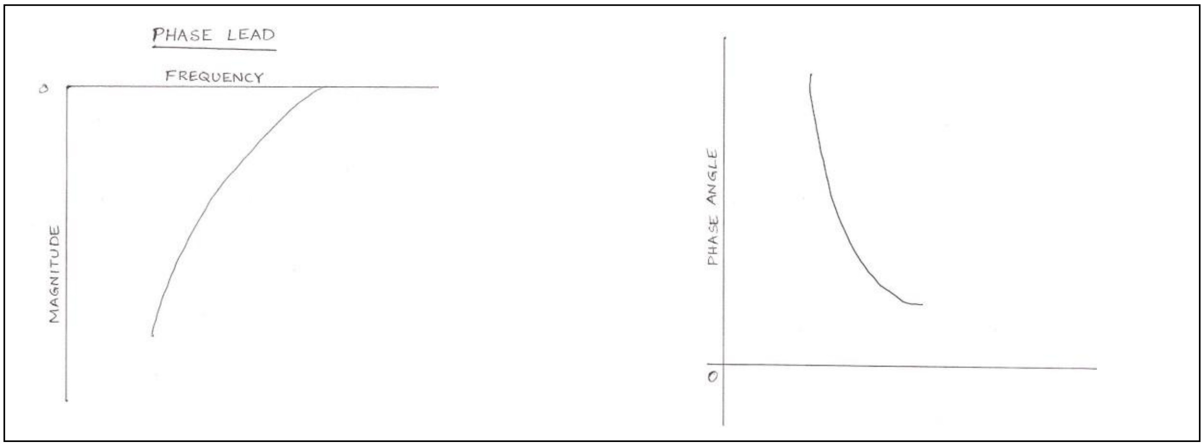
LAG LEAD COMPENSATOR

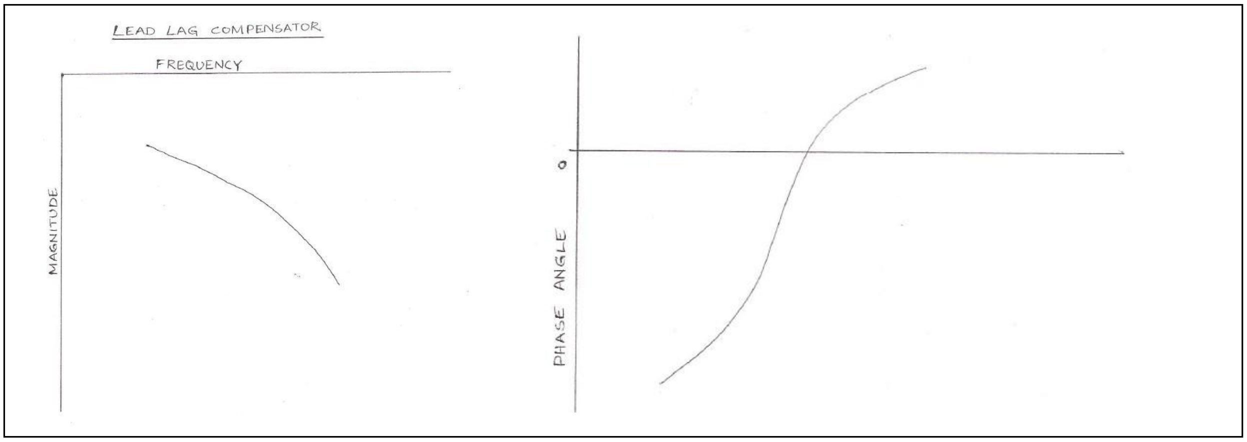
Angle measurment in C.R.O.



GRAPHS:







<table>
<tr><th colspan="8">LAG COMPENSATOR</th></tr>
<tr><th colspan="3">PRACTICAL</th><th colspan="5">THEORITICAL</th></tr>
<tr><th>Sr. No.</th><th>f(Hz)</th><th>Vo</th><th>Vi</th><th>IT(jw)|db =20 log |T(jw)|)</th><th>θ</th><th>IT(jw)|</th><th>θ</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><th colspan="8">LEAD COMPENSATOR</th></tr>
<tr><th colspan="3">PRACTICAL</th><th colspan="5">THEORITICAL</th></tr>
<tr><th>Sr. No.</th><th>f(Hz)</th><th>Vo</th><th>Vi</th><th>IT(jw)|db =20 log |T(jw)|)</th><th>θ</th><th>IT(jw)|</th><th>θ</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><th colspan="8">LEAD-LAG COMPENSATOR</th></tr>
<tr><th colspan="3">PRACTICAL</th><th colspan="5">THEORITICAL</th></tr>
<tr><th>Sr. No.</th><th>f(Hz)</th><th>Vo</th><th>Vi</th><th>IT(jw)|db =20 log |T(jw)|)</th><th>θ</th><th>IT(jw)|</th><th>θ</th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
</table>

RESULT:
Thus the design of lag lead compensator network has been verified.

PRECAUTIONS :

1. Loose connections are to be avoided.
2. Circuit connections should not be made while power is on.
3. Readings of meters must be taken without parallax error.

ASSESMENT QUESTIONS:

1. What is the formula for calculating phase angle?
2. What is the formula magnitude of phase lead circuit lag network & loss?
3. What is the difference between lag network & low pass filter?
4. What is meant by compensation?
5. How a lag network can be compensated?

This lab manual has been updated by

Dr. Teena Gakhar

Crosschecked By
HOD EEE

Verified By
Director, DGI Greater Noida

Please spare some time to provide your valuable feedback.