MATLAB MODULE 1

MATLAB Window Environment and the Base Program

Starting 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. M1.1 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.

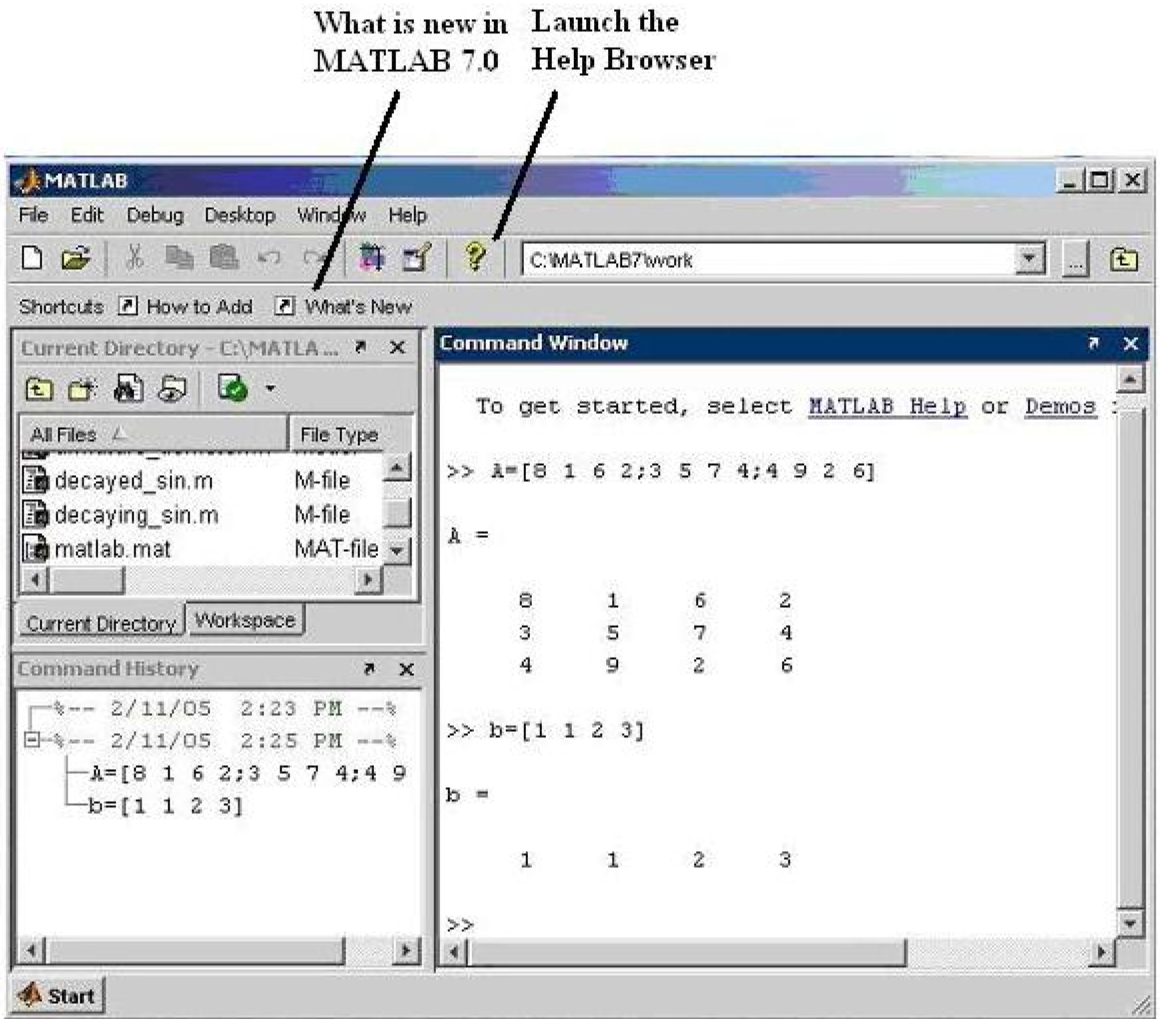

Fig. M1.1 *MATLAB Desktop (version 7.0, release 14)*

Desktop layout can be changed by following **Desktop --> Desktop Layout** from the main menu as shown in Fig. M1.2 (Default option gives Fig. M1.1).

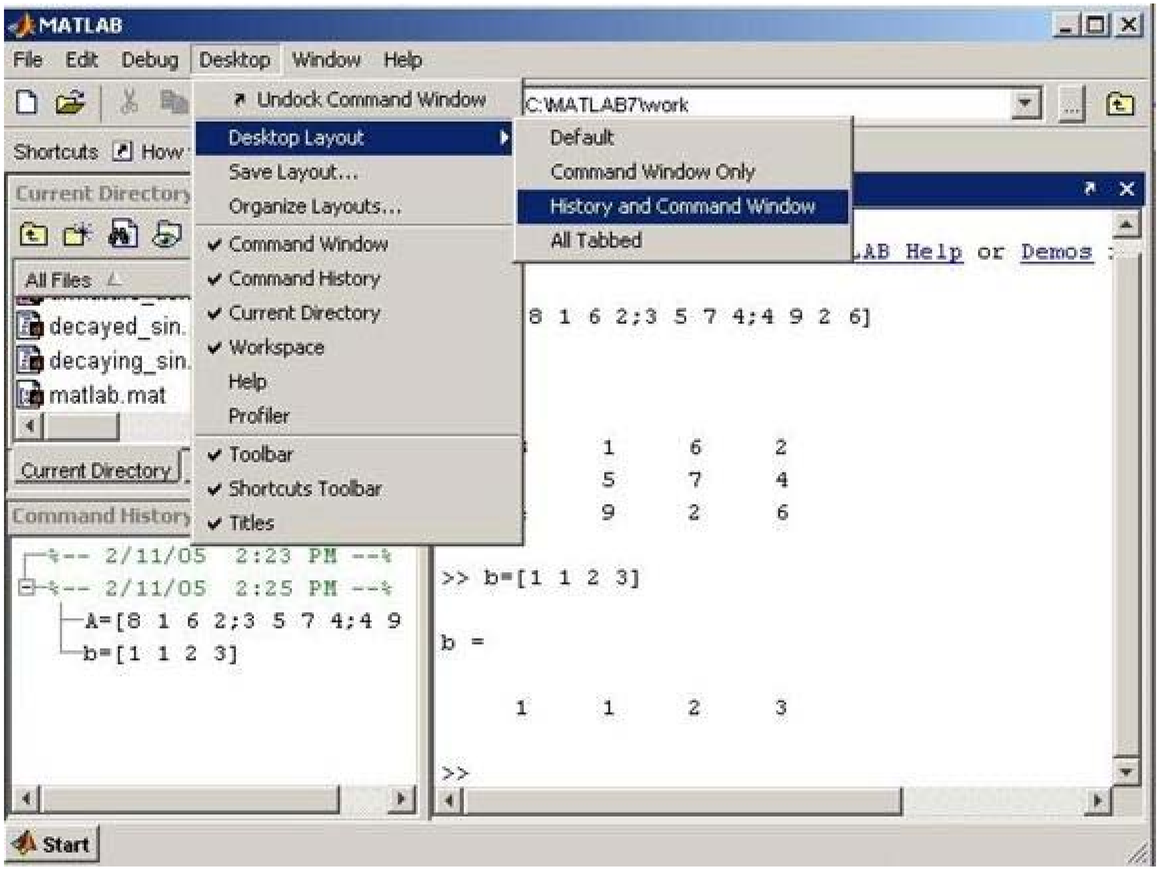

Fig. M1.2 *Changing Desktop Layout to History and Command Window option*

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 a time. 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 (θ) 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. M1.3.

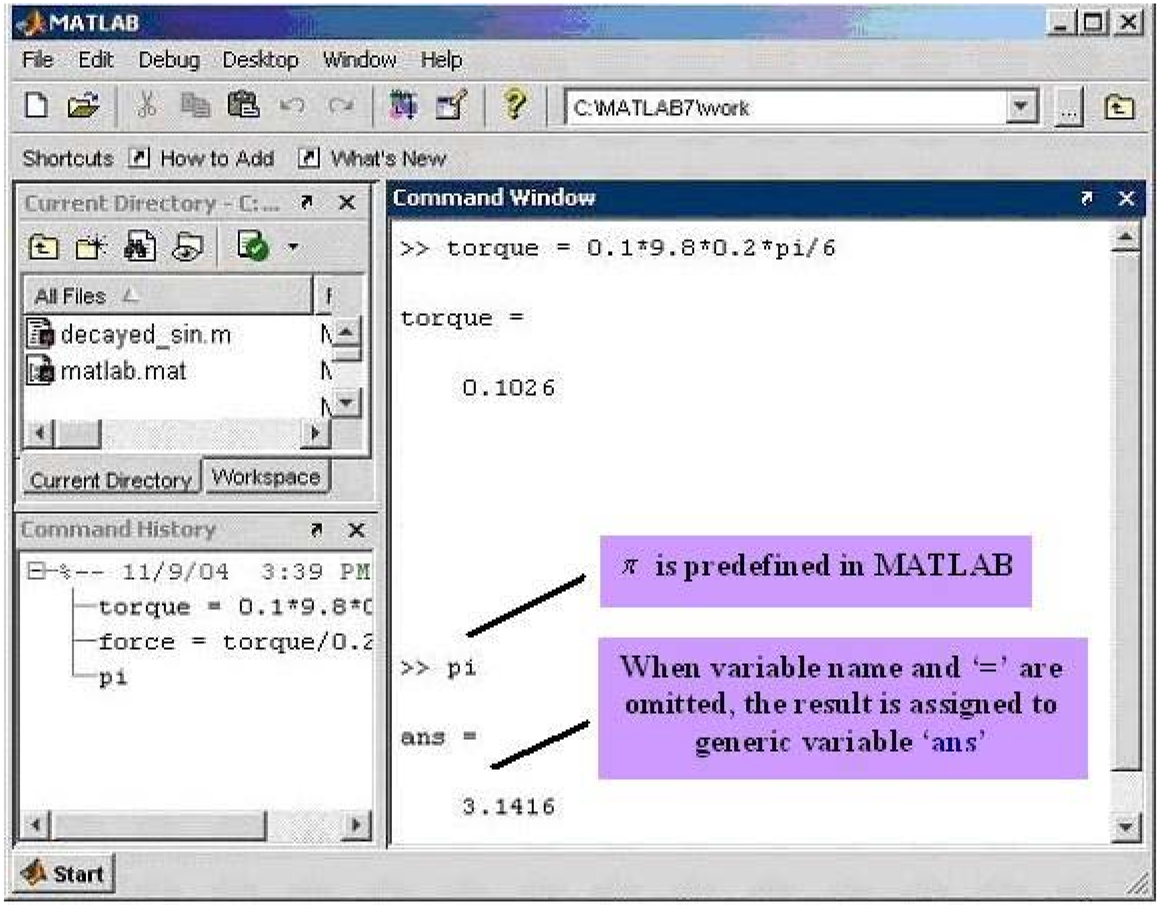

Fig. M1.3 *Command Window for quick scientific calculations* (*text in colored boxes corresponds to explanatory notes*).

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. M1.4).

>> 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 M1.5.

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

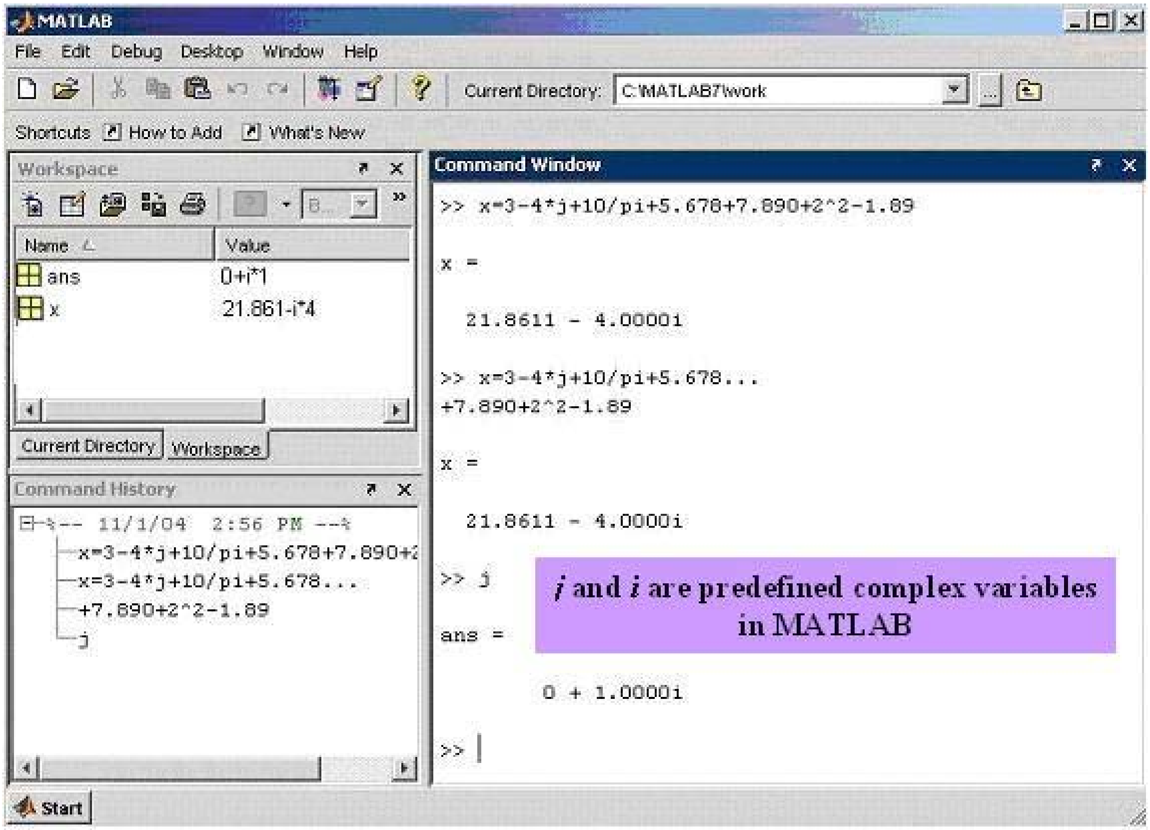

Fig. M1.4 *Command Window with example operations*

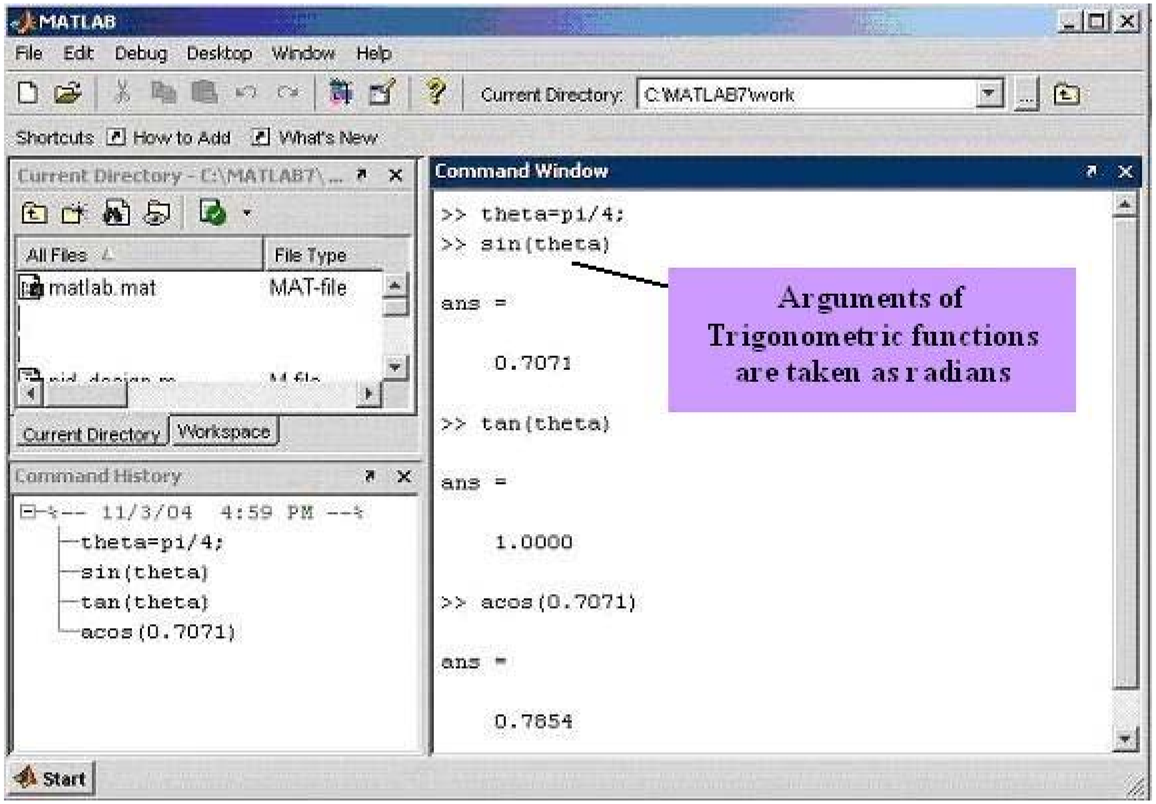

Fig. M1.5 *Example trigonometric calculations*

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

$$z = \sigma + j*\omega \quad \text{or} \quad z = \sigma + \omega j$$

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

MATLAB representation of complex number $z = re^{j\theta}$:

$$z = r * exp(j*\theta) \quad \text{or} \quad z = r * exp(\theta j)$$

$$z = r * (cos\theta + j * sin\theta) \quad \text{or} \quad z = r * (cos\theta + sin\theta * j)$$

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 θ_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. M1.6.

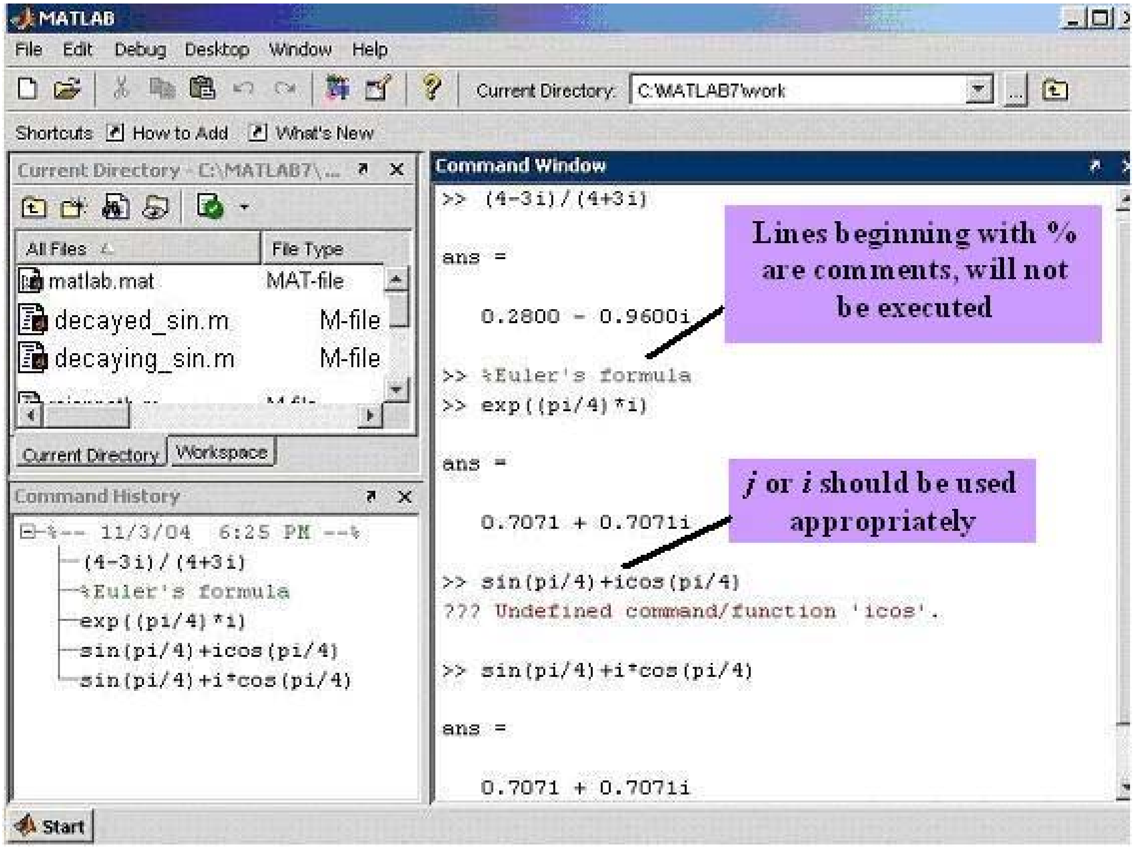

Fig. M1.6 *Example complex numbered calculations*

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.

Exercise M1.1

i. By using arbitrary values of θ, check that $\sin^2(\theta)+\cos^2(\theta)=1$.

ii. Verify with a few arbitrary values of θ that $\sinh(\theta)=\dfrac{\left(e^{j\theta}-e^{-j\theta}\right)}{2j}$.

iii. Verify with a few arbitrary values of θ that $\tan(\theta)=\sqrt{\dfrac{1-\cos(2\theta)}{1+\cos(2\theta)}}$.

iv. For t =0, 2, 5, 7, 12 and 25, find the value of the function $y(t)=e^{-0.2t}\left(\cos t+j\sin t\right)$.

Exercise M1.2

1. Try entering complex number $3+4i$ in MATLAB as **3+j4** and check the answer. Initialize $j=4$ and then enter **3+4j**, **3+j*4**, and **3+4*j** and check the various answers. Interpret messages given by MATLAB.

2. Calculate magnitude and phase of the following complex numbers for $\omega = 10\pi$ rad and $T = 0.1$ sec using MATLAB.

   a. $G(j\omega) = 1 + j\omega T$

   b. $G(j\omega) = \dfrac{1}{1+j\omega T}$.

3. Use MATLAB to calculate the magnitude and phase of $G(j\omega)$ for $\omega = 2\pi$ rad.

$$G(j\omega) = \frac{1-j\omega}{j\omega(1+j2\omega)}$$

Exercise M1.3

1. Calculate the quantity $\dfrac{1+e^{-x}}{1-e^{-x}}$ for $x = -1, 0,$ and 1.
2. Calculate $2e^{-t} - 3e^{-2t}$ for $t = -1, 0,$ and 1.

Note: **Inf**, and **NaN** are predefined in MATLAB. **NaN** stands for Not-a-Number and results from undefined operations like 0/0. **Inf** represents $+\infty$.

Current Directory Window

This window (Fig. M1.7) shows the directory, and files within the directory which are 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.

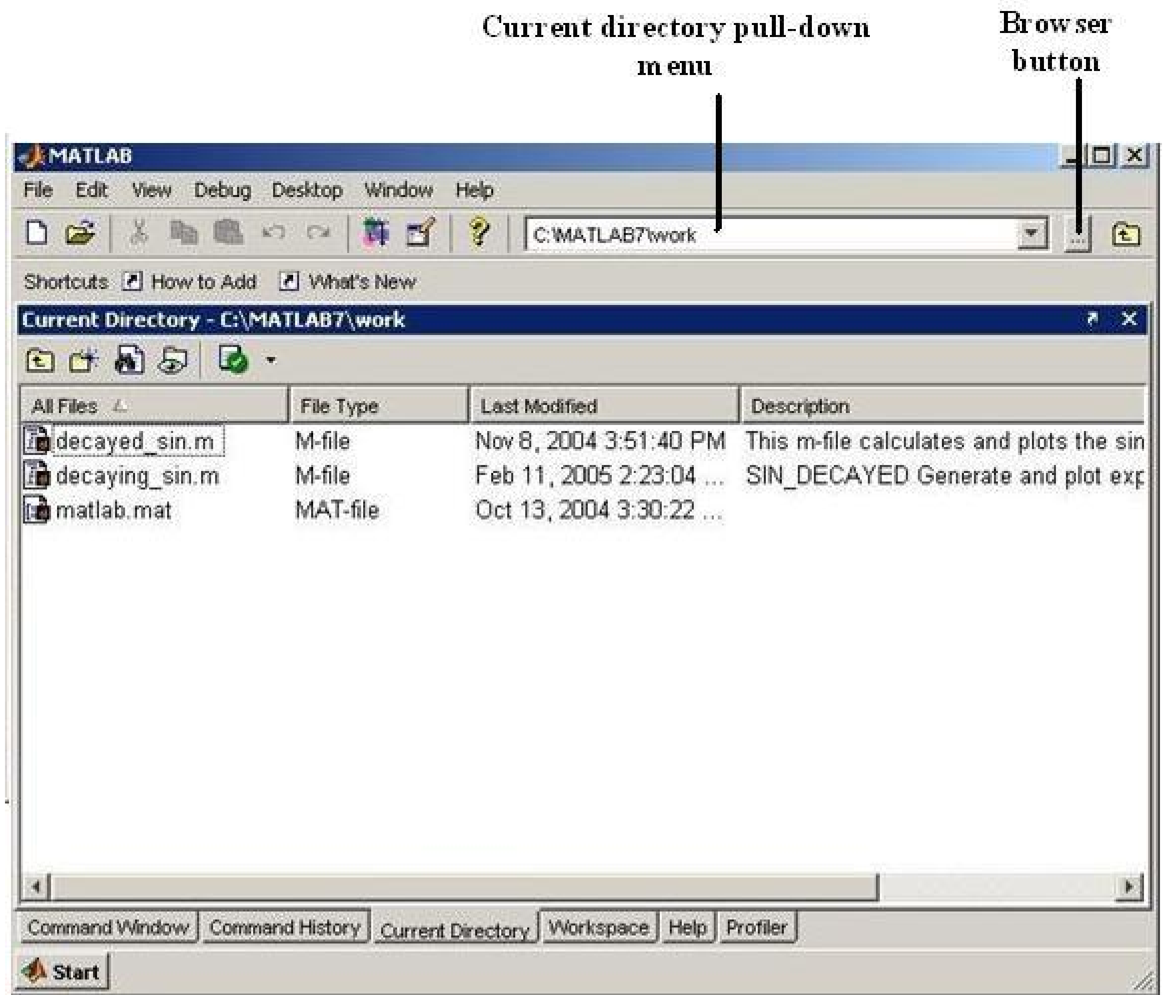

Fig. M1.7 *Current directory window*

One can also use command line options to deal with directory and file related issues. Some useful commands are shown in Table M1.1.

Table M1.1

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 M1.1 are shown in Fig. M1.8.

Workspace

Workspace window shows the name, size, bytes occupied, and class of any variable defined in the MATLAB environment. For example in Fig.M1.9, ‘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. M1.10). 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.

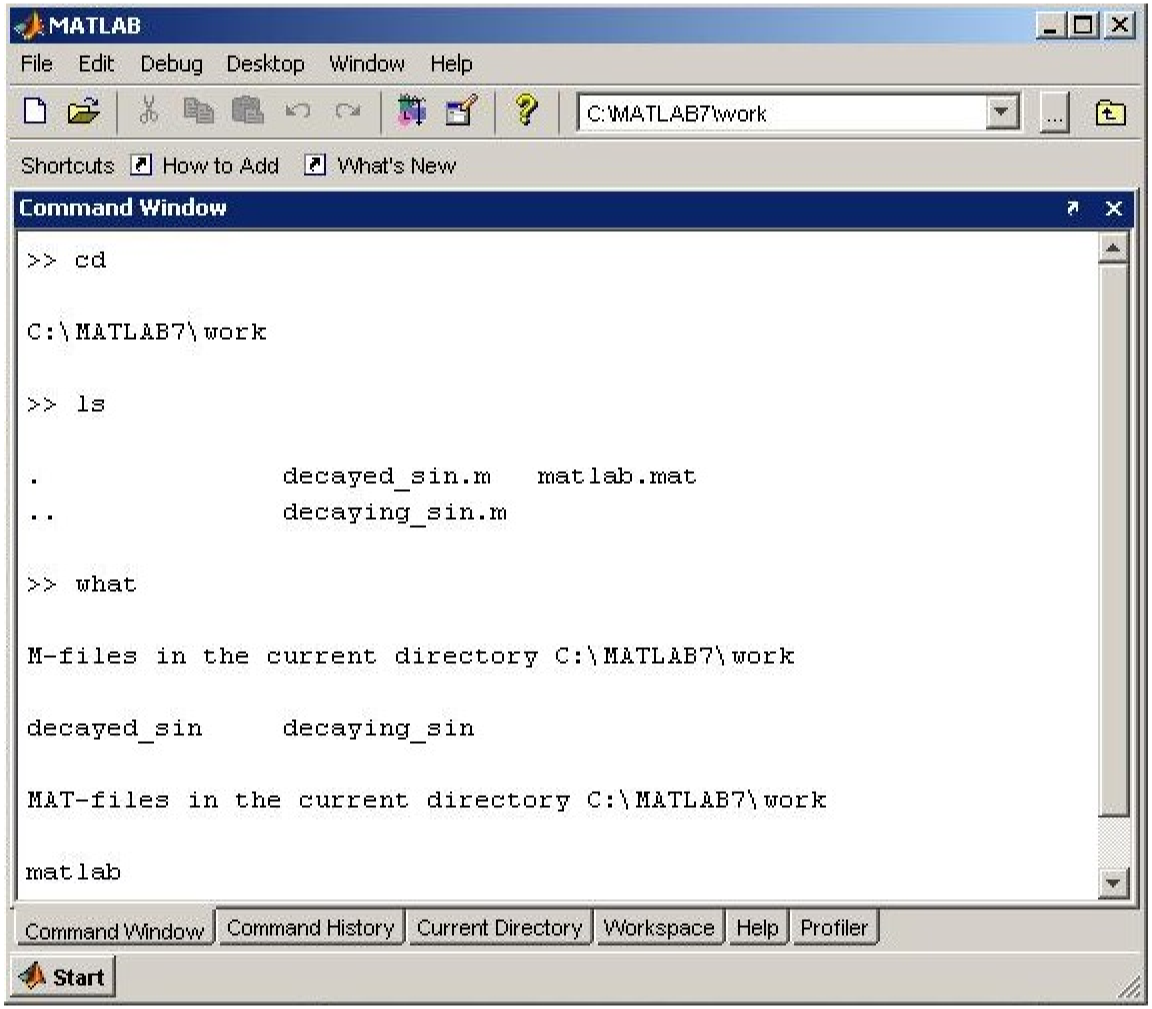

Fig. M1.8 *Example directory related commands*

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.

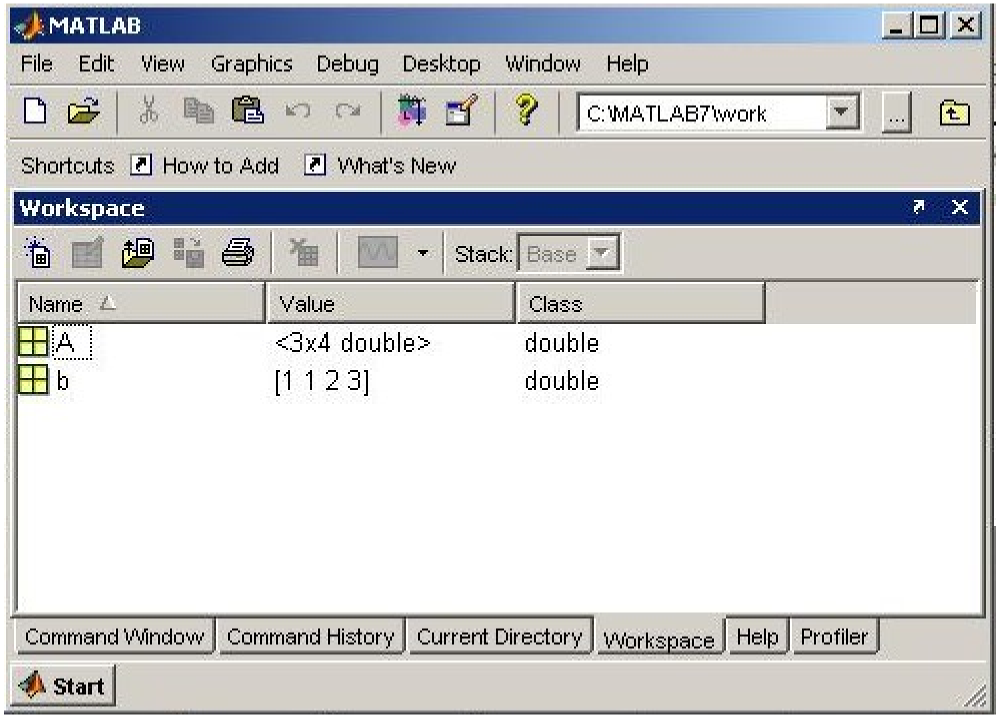

Fig. M1.9 *Entries in the Workspace*

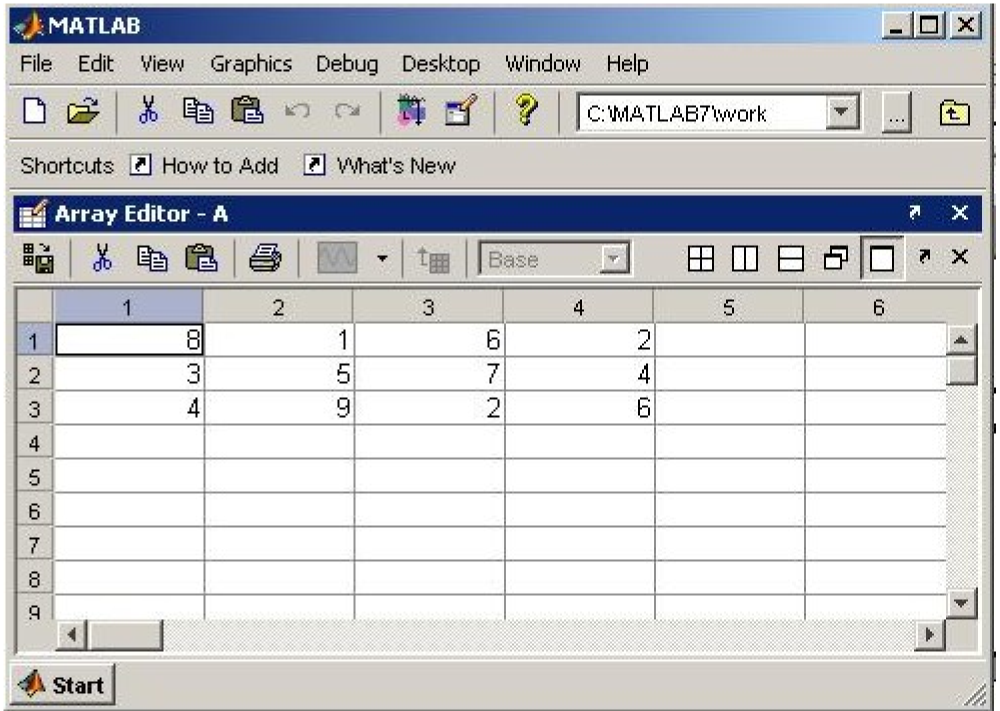

Fig. M1.10 *Array editor window*

Workspace related commands are listed in Table M1.2.

Table M1.2

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. M1.11) 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.

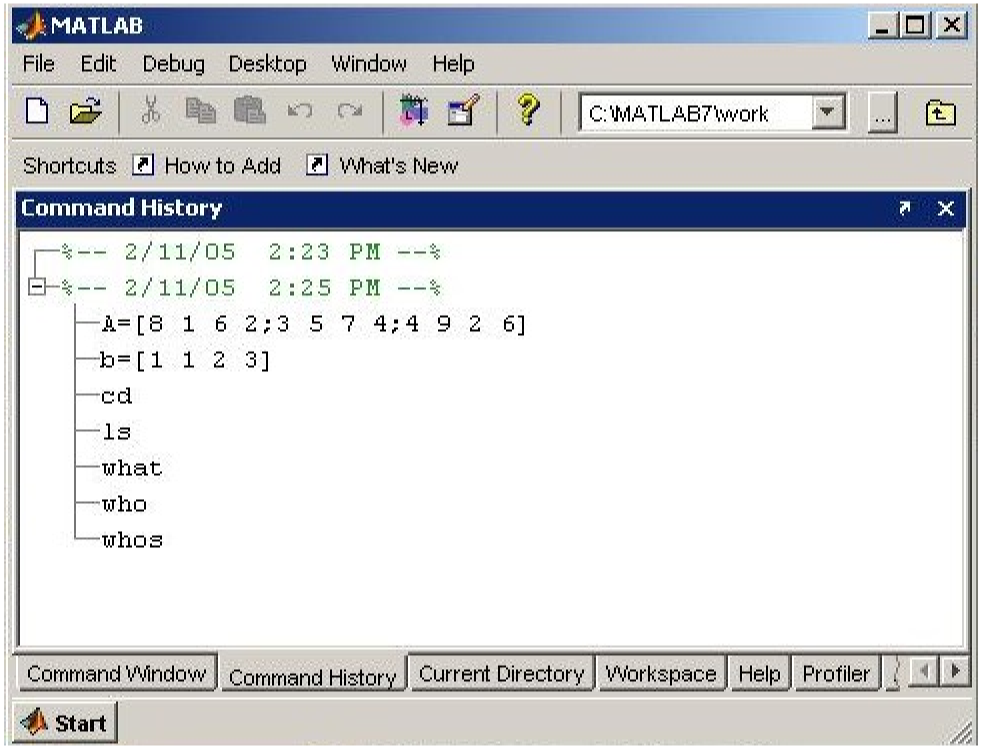

Fig. M1.11 *Command history window*

Commands can also be recalled with the **up-arrow** $(\uparrow)$ 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. M1.12.