Will MATLAB eliminate the need for FORTRAN or C ? The answer is no. FORTRAN and C are still important for high-performance computing that requires large memory or long computing time.

The speed of MATLAB computations is significantly lower than with FORTRAN or C because MATLAB is paying for the advanced features.

However, learning FORTRAN or C, is not a prerequisite for understanding MATLAB.

The results of the computation may differ slightly among different computers because errors can vary.

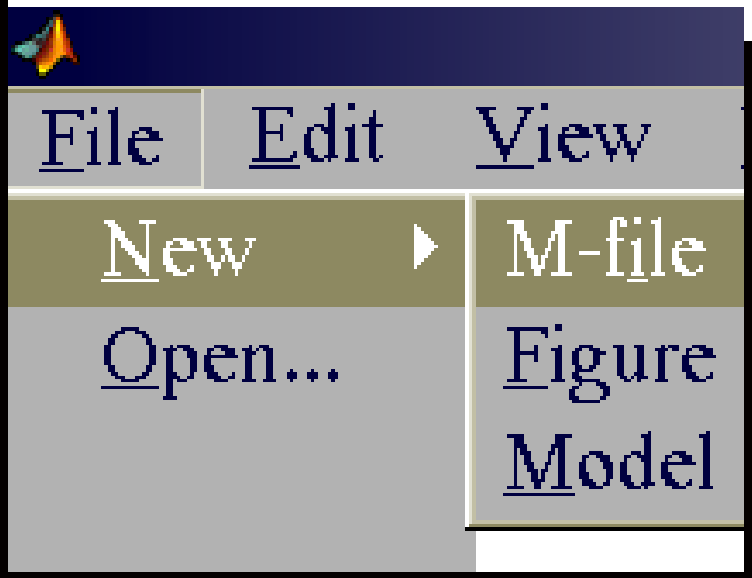
FILES

M-File: a script or function file (filename.m)

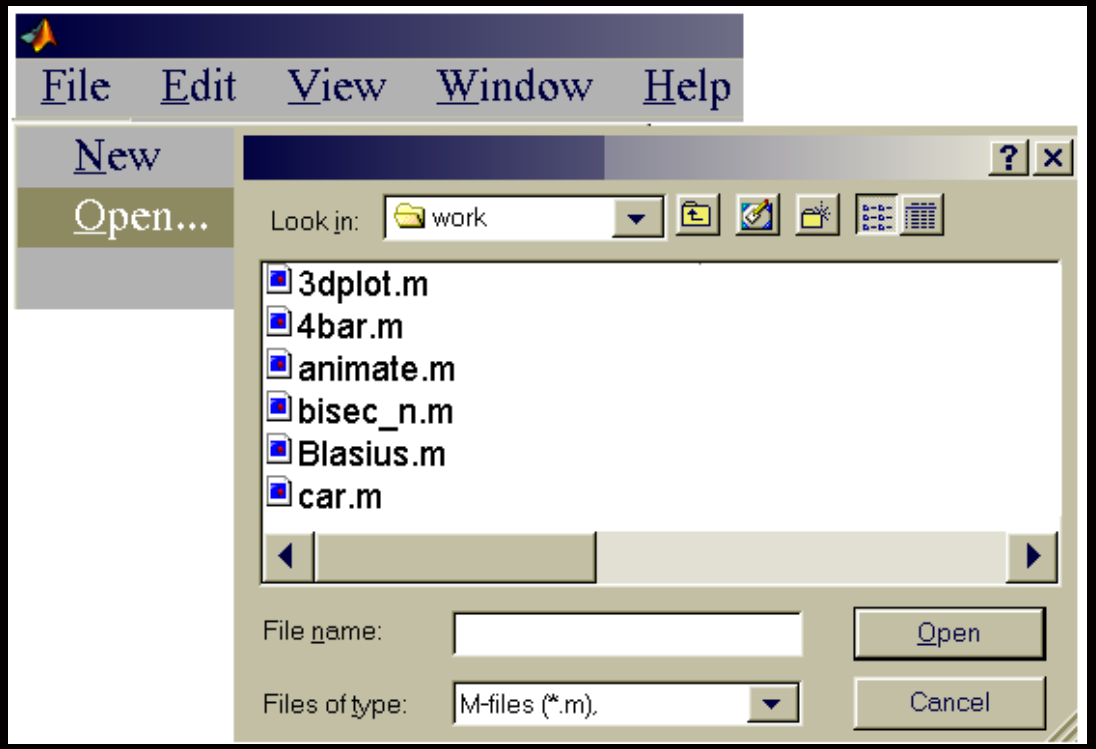
MAT-File: a file containing binary data (filename.mat)

MEX-File: MATLAB excutable file compiled from FORTRAN or C (filename.mex)

Command Window



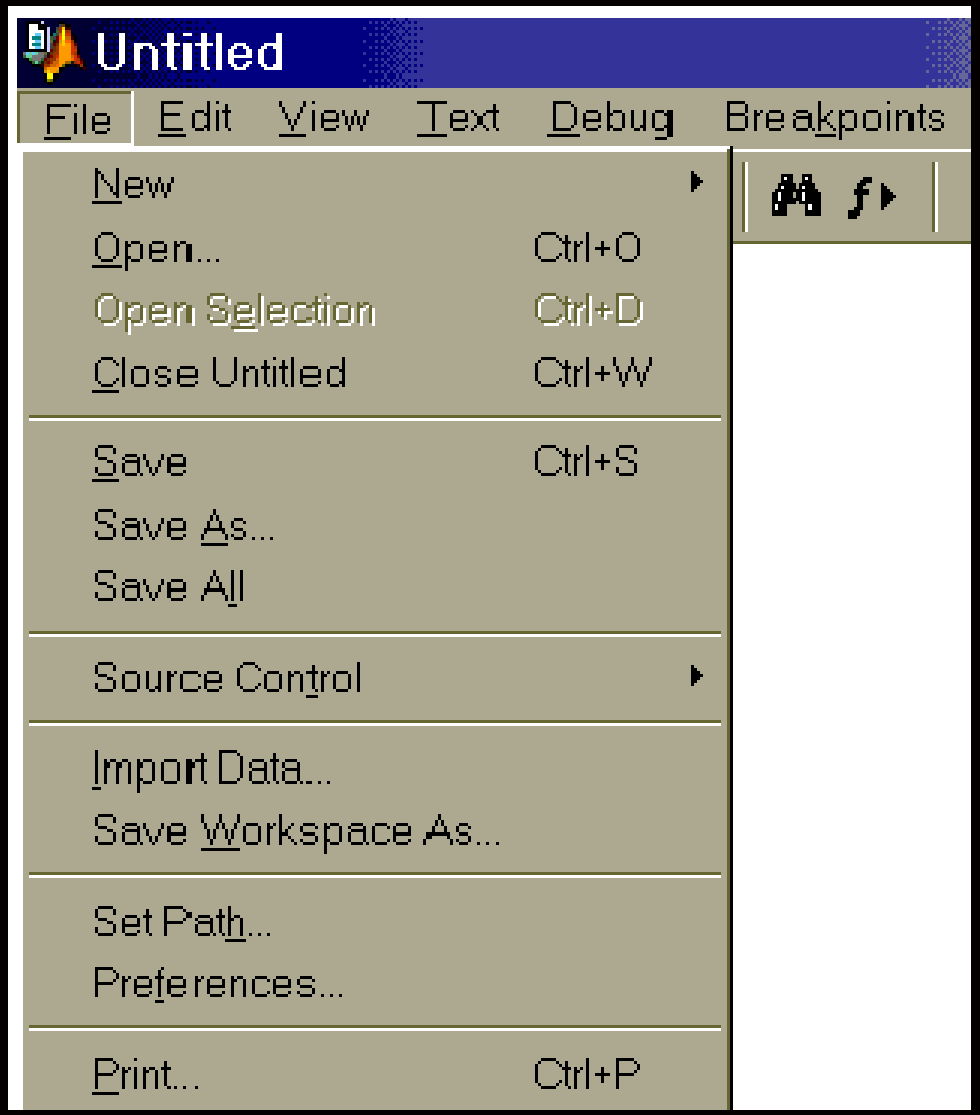
To open a new M-File from the command window



To open an existing M-File from the command window

M-File environment

To open, close, save and print an M-File



Working in the command window

When a command window is opened, a prompt sign » is seen at the upper left corner of the window.

The arithmetic operators

*	The multiplication operator	a * b
+	The addition operator	a + b
-	The subtraction operator	a - b

/ The division operator a / b

^ The exponent operator a ^ b

Constants (Numbers)

» pi (pi = π = 3.14159)

» inf (infinity)

Comments and titles

% Any comments can be written preceded by this sign.

Variables

Variable names and their types do not have to be declared. As MATLAB makes no distinction among integer, real, and complex variables.

- Any variable can take real, complex and integer values and any array size as well.
- Incompatible names should be avoided, for example, end as a name.
- Avoid conflict with sin, cos, tan etc.

→ Avoid using i, j as user defined variables as they are reserved for complex variables in MATLAB.

→ Lower and upper case variables are different in MATLAB unlike in FORTRAN.

→ Consider the following example:

>> X=1 ; ← A semicolon is added to avoid printing

```
>> x=2 ;
>> a=X+x
a=
    3
```

A semicolon is avoided for printing

```
>> a
a=
    3
```

Otherwise

format

Numbers are displayed in five-digit format by default

Consider the following example:

```
>> pi
ans=
      3.1416
>> format long
>> pi
ans=
      3.14159265358979
>> format short
>> pi
ans=
      3.1416
```

Displays variable in 15 digit format

Displays variables in 5 digit format

Example (In command window)

```
» r = 2 ;
» area = pi * r ^ 2 ;
» area
area =
   12.5664
```

multiple commands in a single line and no results displayed

```
» r = 2 ; area = pi * r ^ 2 ;
```

```
» r = 2 , area = pi * r ^ 2
r =
     2
area =
        12.5664
```

with values of r and area printed out

```
» r = 2 ; area = pi * r ^ 2
area =
   12.5664
```

only area displayed

To type long commands we may write:

```
» r = 2 ;
» area = pi ...
* r ^ 2 ;
» area
area =
   12.5664
```

Three dots mean continuation at the end of the line

if statement

An if statement is always closed with an end statement, for example,

```
» r = 2 ;
» if r > 0 , area = pi * r ^2 ;
end
»
```

The prompt does not appear until end is typed

```
» if  r = = 2 , area = pi * r ^ 2 ;
end
»
```

Double equal sign after if

The following symbols mean:

>	Greater than
<	Less than
> =	Equal or greater than
< =	Equal or less than
= =	Equal to

The logical statements:

and is denoted by &

or is denoted by |

For example,

```
» a = 4 ;
» if a >1   |   b < 0 , c = 20 ; end
» b = -2 ;
» if a >1   |   b < 0 , c = 20 ; end
```

```
» a = 4 ; b = -2;
» if a >1   |   b < 0 , c = 20 ; end
```

The & and | operators can be used in a clustered form, for example,

```
» if ((a = = 2 | b = = 3) & c < 5)  ,  g = 1 ; end

» if ((a = = 2 | b = = 3) & c < 5)   g = 1 ; end
```

Without colon is possible

else or elseif statements:

```
» r = 2 ;
» if r > 3          b = 1 ;
elseif  r = = 3     b = 2 ;
else                b = 0 ;
end
» b
b =
     0
```

Note

elseif and else sometimes become tricky, particularly when the variables after them involve array variables of different sizes.

If they do not work the simple if statements may be repeated as many times as needed.

Loops:

for along with end loop
while along with end loop

are available for MATLAB

Example

```
» for r = 1 : 5
area = pi * r ^ 2 ;
disp([r,area])
end
```

For printing the values of r and area in a raw each time area is computed.

```
    1.0000    3.1416
    2.0000   12.5664
    3.0000   28.2743
    4.0000   50.2655
    5.0000   78.5398
```

Note that no semicolon is necessary after each r = 1 : 5 and end

```
» r = 0 ;
» while r < 5
r = r + 1 ;
area = pi * r ^ 2 ;
disp([r,area])
end
```

Alternative statements

1.0000	3.1416
2.0000	12.5664
3.0000	28.2743
4.0000	50.2655
5.0000	78.5398

```
» r = 6 ;
» while r >1
r = r - 1 ;
area = pi * r ^ 2 ;
disp([r,area])
end
```

Alternative statements

```
 5.0000   78.5398
 4.0000   50.2655
 3.0000   28.2743
 2.0000   12.5664
 1.0000    3.1416
```

```
» for r = 5 : -1 : 1
area = pi * r ^ 2 ;
disp([r,area])
end
```

Alternative statements

```
    5.0000   78.5398
    4.0000   50.2655
    3.0000   28.2743
    2.0000   12.5664
    1.0000    3.1416
```

Second alternative statements may be considered as follows:

```
» for r = 1 : 1 : 5
area = pi * r ^ 2 ;
disp([r,area])
end
```

Alternative statements

```
    1.0000    3.1416
    2.0000   12.5664
    3.0000   28.2743
    4.0000   50.2655
    5.0000   78.5398
```

Double and triple loops can be written as,

```
» for    r = 1: 3
for s = 1: r
area = pi * (r ^ 2 - s ^ 2) ;
disp([r,area])
end
end
```

```
1     0
2.0000    9.4248
2     0
3.0000   25.1327
3.0000   15.7080
3     0
```

break

The break statement terminates the execution of a for or while loop.When used in nested loops, only the immediate loop where break is located is terminated.

Example

```
» for i=1 : 6
for  j =1 : 20
if  j > 2 * i , break , end
end , end
» disp([i , j])
```

6 13

The results

In this example, break terminates the inner loop as $j > 2 * i$ is satisfied once, but the loop continues for i until i = 6

In the programming language such as FORTRAN the command GOTO would be used to break a loop.

MATLAB on the other hand,
has no GOTO statement.

Clearing variables

As we excute commands, MATLAB memorizes the variables used.

Their values stay in memory until we quit MATLAB, or clear the variables.

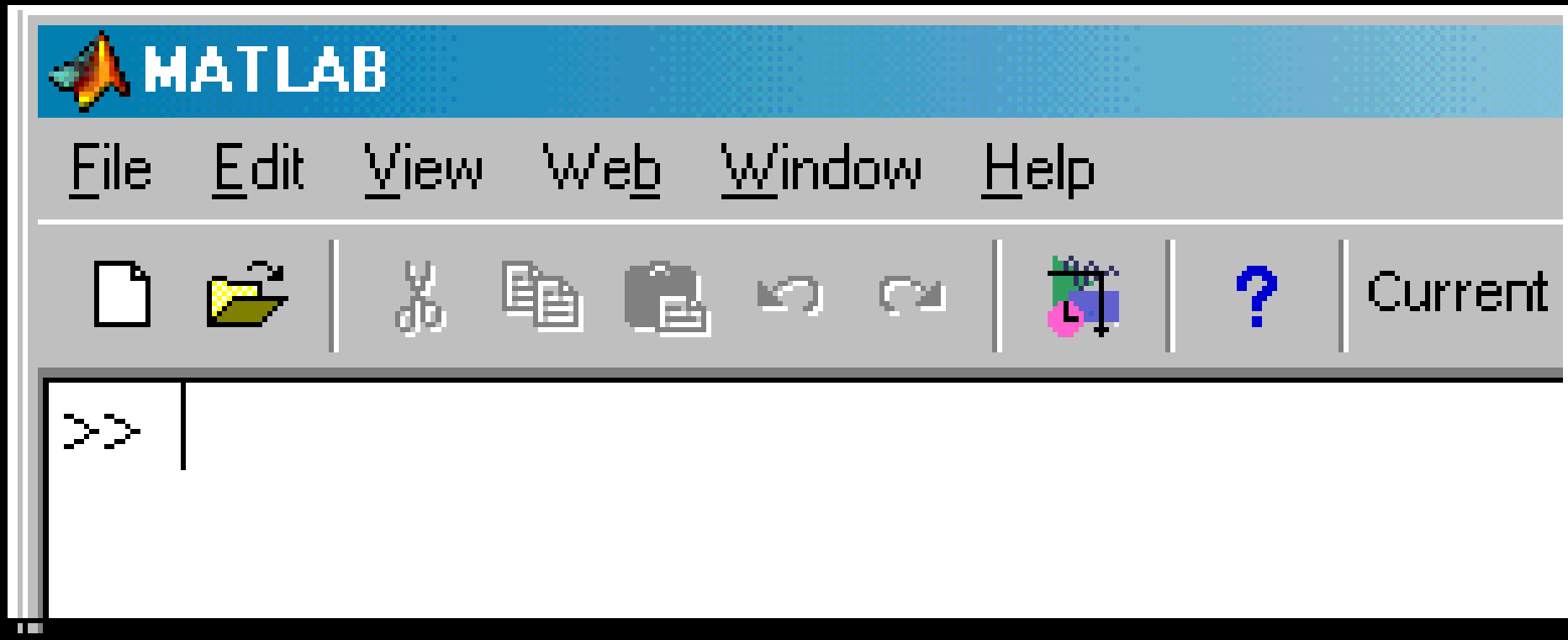
The following commands are available:

clear (To clear all variables)

clear x y z (To clear certain variables)

clc (To clear the window)

The command window may be cleared from the pull-down menu as,



Reading and writing

Passing data to and from MATLAB is possible in several ways:

1-Interactive operation by keyboard or mouse.

2-Reading from or writing to a data file.

3-Using save or load.

Reading input from keyboard

```
» x = input ( ' type length : ' )
type length: |
type length: 4
 x =
    4
```

A prompting message on screen

The input is saved in x once entered from keyboard

The input is typed once entered

```
» x = input ( ' Your name please : ' , ' s ' )
```

To indicate that the input from the keyboard is a string

```
Your name please: El-Gamal
x =
   El-Gamal
```

Type the name (string)to be entered

```
» x = input ( ' Your name please …
                    (in single quote signs) : ' )
```

Without 's' if the typed string
is enclosed by single quote signs

```
Your name…
        please (in single quote signs):'El-Gamal'
```

Type the name (string) to be
entered enclosed by single quotes

```
x =
   El-Gamal
```

Output format

The variable to be printed

New line operator

```
» area = 123.452 ;
» fprintf ( ' The area …
                of the circle = %12.5 f \ n ' , area )
```

A string to be printed out

format for printing similar to FORTRAN

The area of the circle = 123.45200

```
» fprintf ( ' The area of the …
                circle = %12.5 e \ n ' , area )
The area of the circle =1.23452e+002

» fprintf ( ' The area of  the …
                circle = %12.5 f sq.m \ n ' , area )
The area of the circle=   123.45200 sq.m

» fprintf ( ' The area …
                of the circle = %12.3 f \ n ' , area )
The area of the circle =     123.452
```

12 spaces and 3 digits
after decimal point

```
» fprintf ( ' The area of the …
                circle = %12.1f \ n ' , area )
The area of the circle=      123.5
```

12 spaces and 1 digit
after decimal point

```
» fprintf ( ' The area of the…
                circle = %12.0f \ n ' , area )
The area of the circle=      123
```

12 spaces and printed
integer value

Writing into a specific file

» area =123.452 ;

To open an existing file or creating a new file:

fid=fopen(‘notes.m’,’w’)

file identification

file name (may also be of extension .txt)

‘w’ If the file exists the output is discarded

‘a’ If the file exists the output is appended

If no such file exists, a new file is created

» fprintf (fid , ' The area of … the circle = %12.3f \ n ' , area)

» fclose(fid)

Below is the created m-file named notes.m