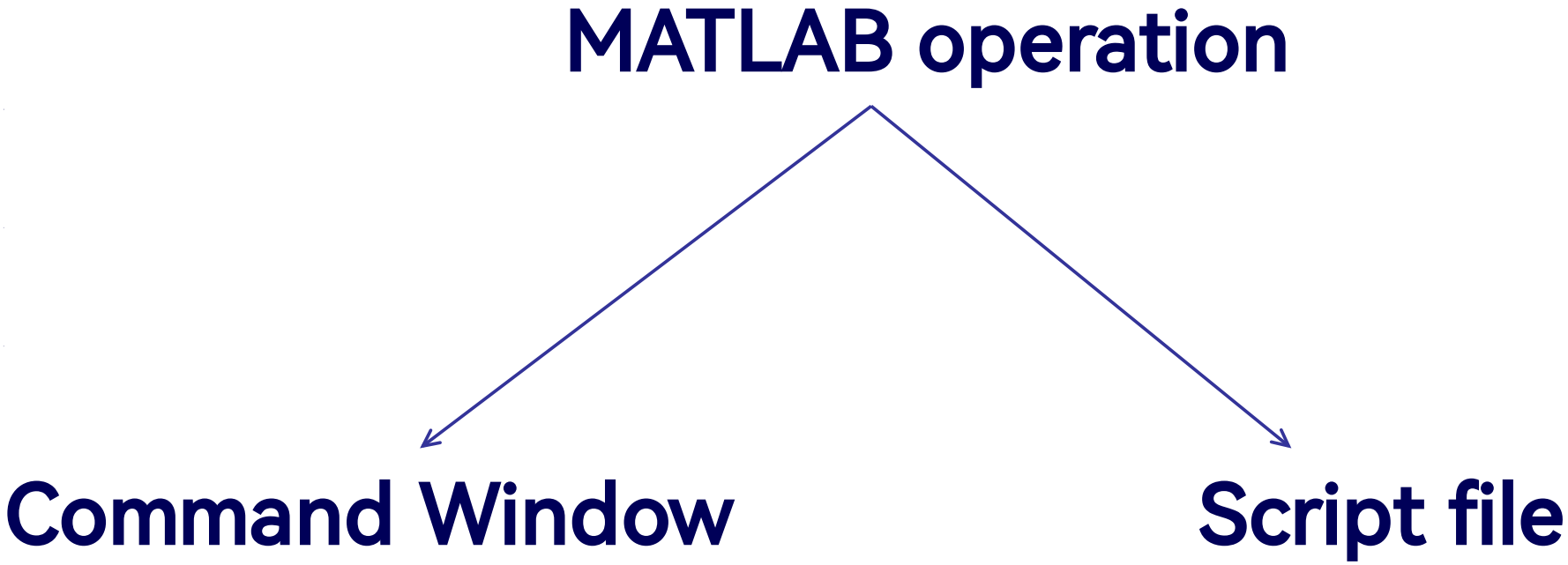
Computer Application II

Lectures (05-7)

Lecturer. Mohamad Hassan

Script files and the Editor/ Debugger



Script files and the Editor/ Debugger

Creating and Using a Script File

- The symbol % designates a comment, which is not executed by MATLAB.

```
% Program example1.m
% This program computes the sine of the square root,
% and dispalys the result.
x=sqrt([3:2:11]);
y=sin(x)
```

Script files and the Editor/ Debugger

Effective use of script Files

- Create script file to avoid the need to retype commonly used procedures.
- Here are some other things to keep in mind when using script files:

o The name of the script file must follow the MATLAB convention for naming variables; that is, the name must begin with letter, and may include digits and the underscore character, up to 31 characters.

Script files and the Editor/ Debugger

- Recall that typing a variable name at the command window prompt causes MATLAB to display the value of that variable. Thus do not give script file the same name as a variable it computes because MATLAB will not be able to execute that script file more than once.
- Do not give a script file the same name as a MATLAB command or function

Script files and the Editor/ Debugger

- As in interactive mode, all variables created by script file are global variables, which mean that their values are available in the basic workspace.
- Variable created by a function file are local to that function, which means that their values are not available outside the function.

Script files and the Editor/ Debugger

Using script Files to Store Data

- You might have applications which require you to access the same set of data frequently. If so, you can store the data in an array within a script file.
- An example is a set of daily temperature measurements at a particular location, which are needed from time to time for calculations.

Script files and the Editor/ Debugger

- As a short example, consider the following four-line script file, whose name is mydata.m.

```
% File mydata.m. Stores temperature data.
% Store the array temp_F,
% Which contains temperatures in degree Fahrenheit.
temp_F = [72, 68, 75, 77, 83, 79]
```

Script files and the Editor/ Debugger

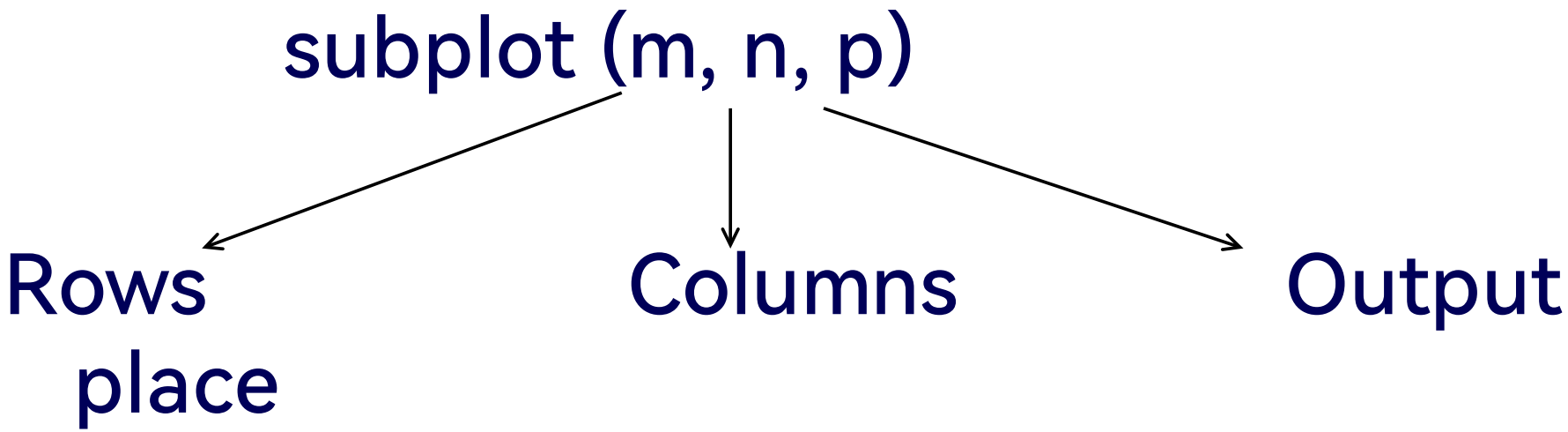
- The array temp_F contains temperatures in degrees Fahrenheit.
- A session to access this data from the Command window, and convert the temperature to degree Celsius, is

```
>> mydata
temp_F =
72 68 75 77 83 79
>> temp_C=5*(temp_F-32)/9
temp_C =
22.2222 20.0000 23.8889 25.0000 28.3333 26.1111
```

- Thus 68° Fahrenheit corresponds to 20° Celsius.

Subplots

- MATLAB can create a figures that contain an array of plots, called subplots.
- These are useful when we want to compare the same data plotted with different axis types, for example. The MATLAB subplot command creates such figures.


subplot (m, n, p)
Rows
place
Columns
Output

- For example, subplot (3, 2, 5) creates an array of six panes, three panes deep and two panes across, and direct the next plot to appear in the fifth pane (in the bottom-left corner).

	Column #1	Column #2
Raw #1	Pane #1	Pane #2
Raw #2	Pane #3	Pane #4
Raw #3	Pane #5 Output Place	Pane #6

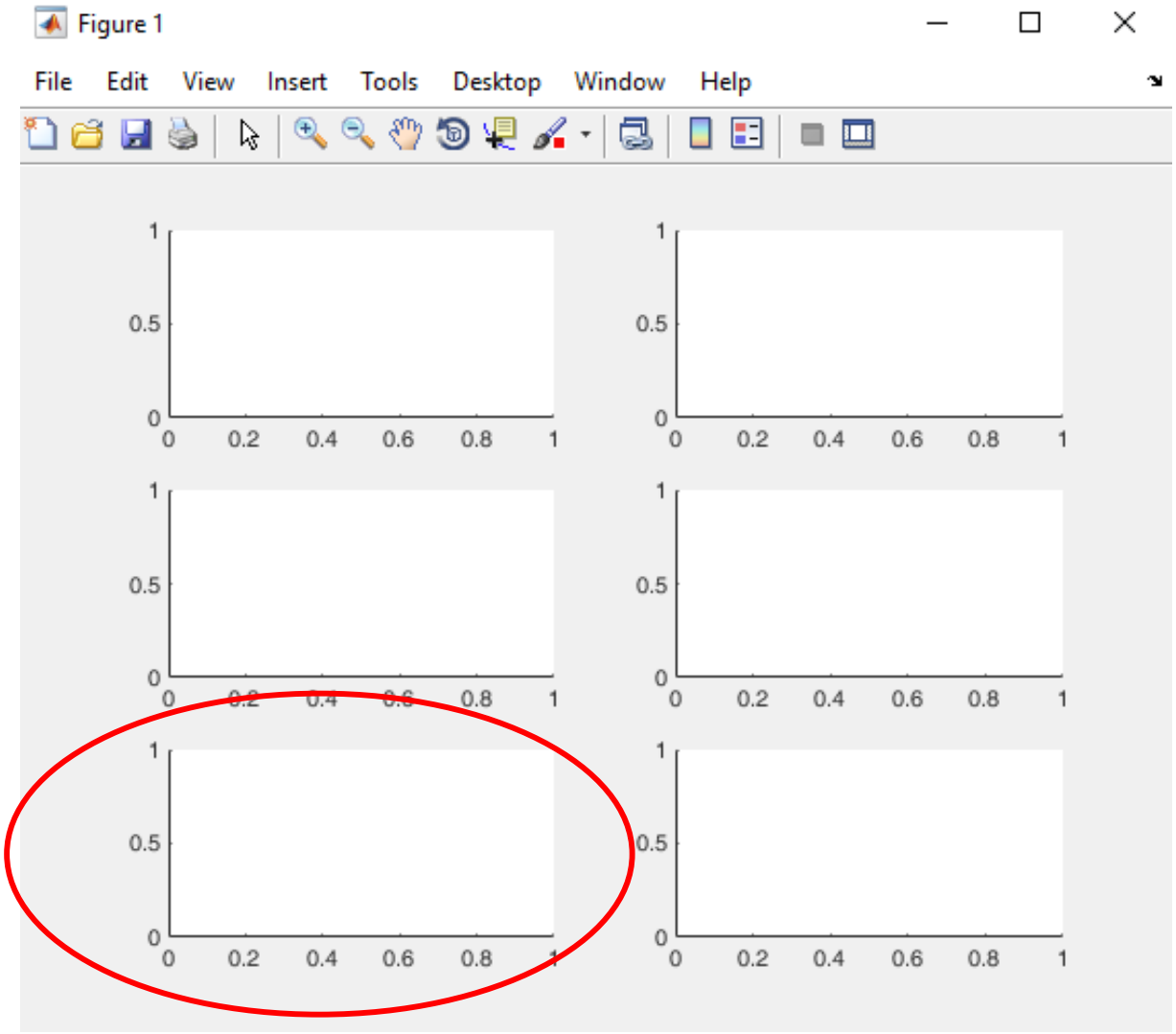


Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.5
0
0 0.2 0.4 0.6 0.8 1

Labeling Curves and Data

- When more than one curve or data set is plotted on a graph, we must distinguish between them. If we use different data symbols or different line types, then we must either provide a legend or place a label next to each curve.

- To create a legend, use the legend command.
- The basic form of this command is legend('string1','string2'), where string1 and string2 are text strings of your choice.
- The legend command automatically obtains from the plot the line type use for each data set and displays a sample

Conditional Statements

- The MATLAB *conditional statements* allow us to write programs that make decisions.
- Conditional statement contain one or more of the if, else, and elseif statements.
- The end statement denotes the end of a conditional statement

Conditional Statements

- The if statement in the following script file accomplishes this in MATLAB, assuming that the variable x already has a scalar value.

```
if x>=0
y=sqrt(x)
end
```

- If x is negative, the program takes no action.

Conditional Statements

- When more than one action must occur as a result of a decision, we can use the else and elseif statements.
- The statements after the else are executed if all the preceding if and elseif expressions are false. The general form of the if statement is:

Conditional Statements

If expression
commands
elseif expression
commands
else
commands
end

- The else and elseif statements may be omitted

Conditional Statements

Suppose that we want to compute y from $y = \sqrt{x}$ for $x \geq 0$, and $y = -\sqrt{-x}$ for $x < 0$. The following statements will calculate y, assuming that the variable x already has scalar value.

```
if x>=0
y=sqrt(x)
else
y=-sqrt(-x)
end
```

Conditional Statements

Loops

- A loop is a structure for repeating a calculation a number of times. Each repetition of the loop is a pass.
- There are two types of explicit loops in MATLAB: the *for loop*, used when the number of passes is known a head of time, and the *while loop*, used when the looping process must terminate when a specified condition is satisfied, and thus the number

Conditional Statements

- A simple example of a for loop is:

```
m=0;
x(1)=10;
for k=2:3:11
m=m+1;
x(m+1)=x(m)+k^2;
end
```

Conditional Statements

- The loop variable k is initially assigned the value 2. During each successive pas through the loop k is incremented by 3, and x is calculated until k exceeds 11.
- Thus, k takes on the values 2, 5, 8, 11.

Conditional Statements

- The while loop is used in cases where the looping process must terminate when a specified condition is satisfied, and thus the number of passes is not known in advance.

Conditional Statements

- A simple example of a while loop is as follows:

```
x=5;
k=0;
while x<25
k=k+1;
y(k)=3*x;
x=2*x-1;
```

Conditional Statements

- The loop variable x is initially assigned the value 5, and it keeps this value until the statement x=2*x-1 is encountered the first time. Its value then changes to 9. before each passes through the loop, x is checked to see if its value is less than 25. If so, the pass is made. If not, the loop is skipped and the program continues to execute any statements following the end statement. The variable x takes on the values 9, 17, and 33 within the loop. The resulting array y

User-Defined Functions

- Another type of M-file is a function file. Unlike script file, all the variable in a function are local, which means their values are available only within the function.
- Function files are useful when you need to repeat a set of a commands several times.
- Function files are like functions in C,

User-Defined Functions

- They are the building blocks of larger programs. The first line in a function file must begin with a function definition line that has a list of inputs and outputs.
- This lines distinguishes a function M-file from a script M-file. Its syntax is as follows:

```
function [output variables]=function_name(input variables);
```

User-Defined Functions

- The output variables are enclosed in square brackets, while the input variables must be enclosed with parentheses.
- The function_name must be the same as the file name, in which is saved (with the m.extension)

Some MATLAB programming statement

Command	Description
else	Delineates an alternate block of commands.
elseif	Conditionally executes an alternate block of commands.
end	Terminates *for*, *while*, and *if* statements.
find(x)	Computes an array containing the indices of the nonzero elements of the array x.
for	Repeats commands a specified number of times.
if	Executes commands conditionally.
while	Repeats commands an indefinite number of times.

Local and Global Variables

- The names of the input variables given in the function definition line are local to that function.
- This means that other variable names can be used when you call the function.
- This is what is meant by the function's variables being local to the function.

Local and Global Variables

- The global command declares certain variables global, therefore their values are available to the basic workspace and to other functions that declare these variable global.
- The syntax to declare the variables a, x, and q global is:

```
global a x q
```

Local and Global Variables

- Any assignment to those variables, in any function or in the base workspace, is available to all the other functions declaring them global. If the global variable doesn't exist the first time you issue the global statement, it will be initialized to the empty matrix.