MATLAB Workshop 4 - Managing the MATLAB Workspace

Objectives: Learn about managing the MATLAB workspace.

MATLAB Features:

workspace commands

Command	Action
`who`	lists all variables in workspace
`whos`	lists all variables in workspace with variable properties
`clear`	removes all variables from workspace
`clear var_name`	removes only specified variable from workspace
`clear all`	removes all variables, functions, etc., from workspace
`clc`	clears the Command Window and resets the cursor to the top without affecting the workspace
`save filename`	saves workspace as **`filename.mat`** in current directory
`load filename`	loads the workspace **`filename.mat`** from current directory **warning**: the **`load`** command will overwrite variables of the same name in the current workspace and **`filename`** with the values in **`filename`**
`help`	provides a listing of all help files and help file groups
`help name`	provides help information on **`name`**
`diary name`	creates and opens the specified diary • provide full directory path, filename, and extension ex: **`c:\temp\engr_hmwk`** • defaults current directory: **`engr_hmwk`** • if diary already exists opens for *appending* new session all lines following this command are placed in diary
`diary off`	closes the current diary
`diary on`	opens the most recently used diary

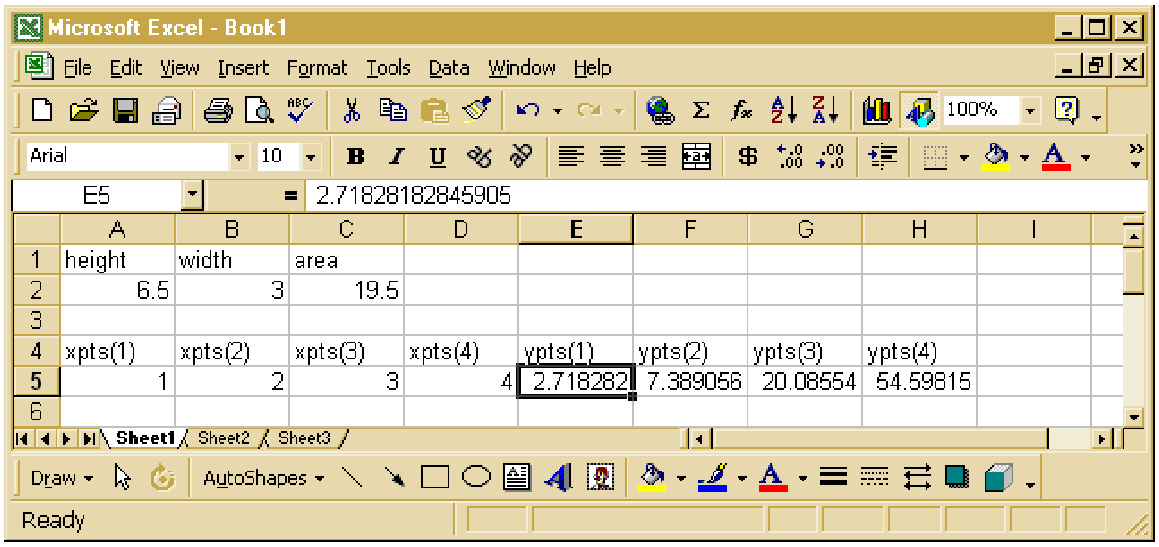
- **The MATLAB workspace**

MATLAB creates a workspace which it uses to keep track of all of the variable names and values associated with the variables from commands that are issued in the Command Window. Ordinarily, we do not need to keep track of the workspace. However, there are a few features of the workspace with which we should be familiar.

Enter the following commands in the Command Window.

```
» height = 6.5;
» width = 3.0;
» area = height*width;
» xpts = 1:4;
» ypts = exp(xpts);
```

We can think of the workspace as being similar to a spreadsheet as depicted here. Each of the



E5 = 2.71828182845905

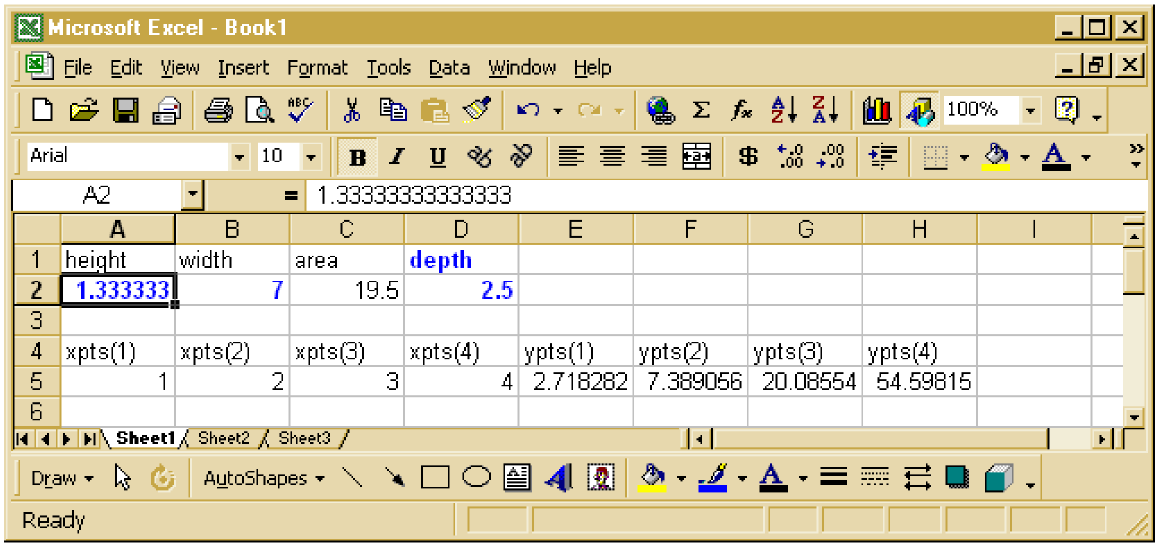
	A	B	C	D	E	F	G	H	I
1	height	width	area						
2	6.5	3	19.5						
3									
4	xpts(1)	xpts(2)	xpts(3)	xpts(4)	ypts(1)	ypts(2)	ypts(3)	ypts(4)	
5	1	2	3	4	2.718282	7.389056	20.08554	54.59815	
6									

above commands creates a new variable by naming it on the left hand side of the assignment operator. MATLAB will look for available locations in the workspace to store the value(s) assigned to the variables. As depicted here, it takes two cells to hold a value: a cell for the variable name and a cell for the variable value just below the variable name. MATLAB stores the value to ~16 significant figures, as noted for the variable **`ypts(1)`**. When a variable name is used on the right hand side of the assignment operator, MATLAB gets the values associated with the variable name and uses them as specified. Thus the value for **`area`** was computed by getting the values for **`height`** and **`width`** and multiplying them together before storing the result in **`area`**. Likewise, the values for **`ypts`** were created by using the exponentiation function on the corresponding values of **`xpts`**.

We can change variable values in the workspace and add new variables to the workspace as needed for our problems. For example, enter the commands

```
» height = 4.0/3.0;
» width = 7.0;
» depth = 2.5;
» tri_area = base*height/2.0;
??? Undefined function or variable 'base'.
```

MATLAB made the changes to the workspace depicted in (bold) blue here. In addition, MATLAB issued the error message because an attempt was made to use a variable (**`base`**) that had not previously been defined. As will be seen shortly, MATLAB also *did not* create a variable named **`tri_area`** because no assignment was made.



A2 = 1.33333333333333

	A	B	C	D	E	F	G	H	I
1	height	width	area	**depth**					
2	**1.333333**	**7**	19.5	**2.5**					
3									
4	xpts(1)	xpts(2)	xpts(3)	xpts(4)	ypts(1)	ypts(2)	ypts(3)	ypts(4)	
5	1	2	3	4	2.718282	7.389056	20.08554	54.59815	
6									

Remember, we cannot see the workspace and the spreadsheet shown here is only an analogy for the actual workspace. Although we cannot see the workspace, MATLAB provides a few commands that allow us to manage the workspace.

(1) Variables in the workspace.

At this point, you should have entered the preceding commands. To see what variables are active in the workspace, enter

```
» who
Your variables are:
area    depth   height  width   xpts    ypts
```

A listing of variables that are currently in the workspace is provided. Note that **`tri_area`** is not listed even though it was on the left side of an assignment statement.

If you have forgotten what “type” of variable any of these are, you can ask by entering **`whos`** followed by a list of variable names. For example,

```
» whos area xpts
  Name       Size             Bytes  Class
  area       1x1                  8  double array
  xpts       1x4                 32  double array
Grand total is 5 elements using 40 bytes
```

A listing of the properties of the specified variables is provided. **Size** is the dimensions of the variable matrix. A 1x1 variable is a scalar. A 1xN variable is a row vector with N elements. A Nx1 variable is a column vector with N elements. A NxM variable is a matrix with N rows and M columns (We will learn more about matrices later). **Bytes** tells how much memory is used to store the entire variable (vector, matrix). **Class** identifies the type of information. For our purposes, the two classes of information are **`double array`** and **`char array`**. The final part of the response is the total number of values (elements) associated with all of the variables listed. Entering **`whos`** without a variable name will provide the information for every variable in the workspace. Try it!

(2) Saving your work.

A valuable pair of commands are **save** and **load**. **save filename** will save the entire contents of the workspace as **filename.mat** in the current directory. This is convenient if you have been working on a problem and want to set it aside for awhile. To see how the command works, enter

```
» save  wkshp4_ac1
» clear
» who
»
```

The **clear** command will be discussed shortly. For now, note that it has removed the variables from the workspace.

Now enter

```
» load  wkshp4_ac1
» who
Your variables are:
area    depth   height  width   xpts    ypts
» area
area =
   19.5000
```

The variables and their values have been restored.

(3) Managing/maintaining your workspace.

The **most dangerous command** in MATLAB is the **clear** command. The **clear** command is used to remove variables, functions, and other information from the workspace - **permanently**. MATLAB will not warn you that you are removing information - it assumes that, since you are issuing the command, that you know what you are doing. You worked hard to put information in the workspace. Don't remove it by mistake.

Designated variables can be removed by using the **clear** command with the variable name, such as

```
» clear area
» who
Your variables are:
depth   height  width   xpts    ypts
```

The variable **area** and its value have been removed from the workspace.

All of the variables can be removed by typing

```
» clear
» who
»
```

clear alone removes only variables. **clear all** will remove all variables, functions, and other information from the workspace.

(4) Help files.

A **very useful command** in MATLAB is the **help** command. The command can be used to locate useful MATLAB functions for learning how to properly use a particular MATLAB function. For a brief overview, enter

```
» help
HELP topics:
matlab\general        -  General purpose commands.
matlab\ops            -  Operators and special characters.
matlab\lang           -  Programming language constructs.
matlab\elmat          -  Elementary matrices and ...
matlab\elfun          -  Elementary math functions.
...
For more help on directory/topic, type "help topic".
For command syntax information, type "help syntax".
```

MATLAB responds with a long list of help topics.

To find more help on a particular topic, **help name** brings up more information. For example, to find information on the elementary mathematical functions available in MATLAB, enter

```
» help elfun
  Elementary math functions.
  Trigonometric.
    sin           - Sine.
    sinh          - Hyperbolic sine.
    asin          - Inverse sine.
    asinh         - Inverse hyperbolic sine.
    cos           - Cosine.
...
```

MATLAB responds with a long list of help topics about elementary functions.

To find more about how to use a specific function, ask. For example, enter

```
» help sin
   SIN    Sine.
   SIN(X) is the sine of the elements of X.
```

MATLAB responds with information on the function.

Information about all MATLAB-supplied functions can be obtained by using **help function_name**. Usually, more information is supplied than you really nead. For example, try

```
» help max
```

(5) **Diary** command.

You have already learned about the **diary** command. The workspace includes only variables and their values. It does not include the commands that have been used to produce the values assigned to variables. If you want to keep track of all of the commands you have issued, you need to use the **diary** command.

Note: The Command Window contains only a limited number of lines. When the number of lines used in a session exceeds that number, old commands will “scroll off” the top of the window and be lost. Again, if you want to keep track of all of the commands you have issued, you need to use the **diary** command.

Recap: You should have learned

- Basic commands that control the workspace environment.
- That **`clear`** is the most dangerous workspace command.
- That **`help`** is one of the most useful workspace commands.
- How to save your workspace.
- How to recall a saved workspace.