Computer Programming: Script Files

Asst. Prof. Dr. Yalçın İşler
Izmir Katip Celebi University

Outline

- Workspace
- Input to a Script File
- Output Commands
- Save and Load
- Importing and Exporting Data

Script File

- A script file is a list of MATLAB commands, called a program, that is saved in a file. When the script file is executed (run), MATLAB executes the commands.

Workspace

- The MATLAB workspace contains of the set of variables (named arrays) that are defined and stored during a MATLAB session.
- It includes variables that have been defined in the Command Window and variables defined when script files are executed.
- This means that the Command Window and script files share the same memory zone within the computer, which implies that once a variable is in the workspace, it is recognized and can be used, and can be reassigned new values, in both the Command Window and script files.

```
>> 'Variables in memory'
ans =
Variables in memory
>> a = 7;
>> E = 3;
>> d = [5,  a+E,  4,  E^2]
d =
     5    10     4     9
>> g = [a, a^2,  13;  a*E,  1,  a^E]
g =
      7    49    13
     21     1   343
>> who
Your variables are:
E    a    ans  d    g
>> whos
  Name      Size              Bytes  Class
  E         1x1                   8  double array
  a         1x1                   8  double array
  ans       1x19                 38  char array
  d         1x4                  32  double array
  g         2x3                  48  double array
Grand total is 31 elements using 134 bytes
>>
```

Typing a string.

The string is assigned to ans.

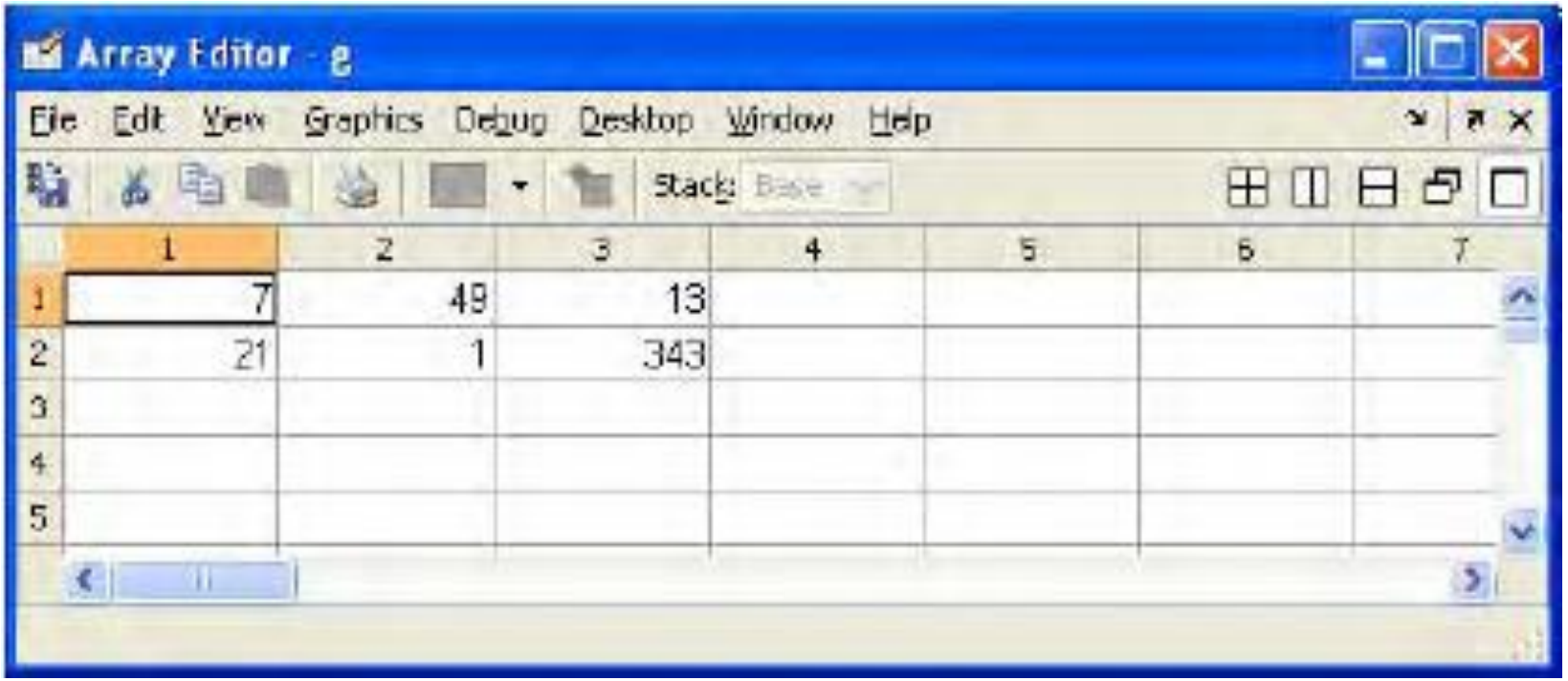
Creating the variables a, E, d, and g.

The who command displays the variables currently in the workspace.

The whos command displays the variables currently in the workspace, and information about their size.

Workspace

- The variables that are displayed in the Workspace Window can also be edited (changed). Double-clicking on a variable opens the Array Editor Window, where the content of the variable is displayed in a table.



Input to a Script

The assignment of a value to a variable can be done in three ways, depending on where and how the variable is defined:

- The variable is defined and assigned value in the script file.
- The variable is defined and assigned value in the Command Window.
- The variable is defined in the script file, but a specific value is entered in the Command Window when the script file is executed.

```
% This script file calculates the average points scored in three games.
% The assignment of the values of the points is part of the script file.
game1=75;
game2=93;
game3=68;
ave_points=(game1+game2+game3)/3
```

The variables are assigned values within the script file.

The Command Window when this file is executed looks like:

```
>> Chapter4Example2

ave_points =
   78.6667
>>
```

The script file is executed by typing the name of the file.

The variable `ave_points` with its value is displayed in the Command Window.

```
% This script file calculates the average points scored in three games.
% The assignment of the values of the points to the variables
% game1, game2, and game3 is done in the Command Window.

ave_points=(game1+game2+game3)/3
```

The Command Window for running this file is:

```
>> game1 = 67;
>> game2 = 90;
>> game3 = 81;
>> Chapter4Example3

ave_points =
    79.3333

>> game1 = 87;
>> game2 = 70;
>> game3 = 50;
>> Chapter4Example3

ave_points =
     69
>>
```

The variables are assigned values in the Command Window.

The script file is executed.

The output from the script file is displayed in the Command Window.

New values are assigned to the variables.

The script file is executed again.

The output from the script file is displayed in the Command Window.

```
% This script file calculates the average of points scored in three games.
% The point from each game are assigned to the variables by
% using the input command.
game1=input('Enter the points scored in the first game ');
game2=input('Enter the points scored in the second game ');
game3=input('Enter the points scored in the third game ');
ave_points=(game1+game2+game3)/3
```

The Command Window for running this file is:

```
>> Chapter4Example4
Enter the points scored in the first game   67
Enter the points scored in the second game   91
Enter the points scored in the third game   70

ave_points =
    76
>>
```

The computer displays the message. Then the value of the score is typed by the user and the **Enter** key is pressed.

The `input` command can also be used to assign a string to a variable. This can be done in one of two ways. One way is to use the command in the same form as shown above, and when the prompt message appears the string is typed in between two single quotes in the same way that a string is assigned to a variable without the `input` command. The second way is to use an option in the `input` command that defines the characters that are entered as a string. The form of the command is:

```
variable_name = input('prompt message','s')
```

where the `'s'` inside the command defines the characters that will be entered as a string. In this case when the prompt message appears, the text is typed in without the single quotes, but it is assigned to the variable as a string.

Output Commands

- MATLAB automatically generates a display when some commands are executed.
- In addition to this automatic display, MATLAB has several commands that can be used to generate displays. The displays can be messages that provide information, numerical data, and plots.

disp

- Every time the `disp` command is executed, the display it generates appears in a new line. One example is:

```
>> abc = [5  9  1;  7  2  4];
>> disp(abc)
     5     9     1
     7     2     4

>> disp('The problem has no solution.')

The problem has no solution.
>>
```

A 2×3 array is assigned to variable `abc`.

The `disp` command is used to display the `abc` array.

The array is displayed without its name.

The `disp` command is used to display a message.

```
% This script file calculates the average points scored in three games.
% The points from each game are assigned to the variables by
% using the input command.
% The disp command is used to display the output.

game1=input('Enter the points scored in the first game    ');
game2=input('Enter the points scored in the second game   ');
game3=input('Enter the points scored in the third game    ');
ave_points=(game1+game2+game3)/3;
disp(' ')
disp('The average of points scored in a game is:')
disp(' ')
disp(ave_points)
```

Display empty line.

Display text.

Display empty line.

Display the value of the variable ave_points.

fprintf

```
fprintf('text typed in as a string')
```

```
fprintf('The problem, as entered, has no solution. Please check the
input data.')
```

If this line is part of a script file, when the line is executed, the following is displayed in the Command Window:

```
The problem, as entered, has no solution. Please check the input data.
```

This is done by inserting \n before the character that will start the new line.

```
fprintf('The problem, as entered, has no solution.\nPlease
check the input data.')
```

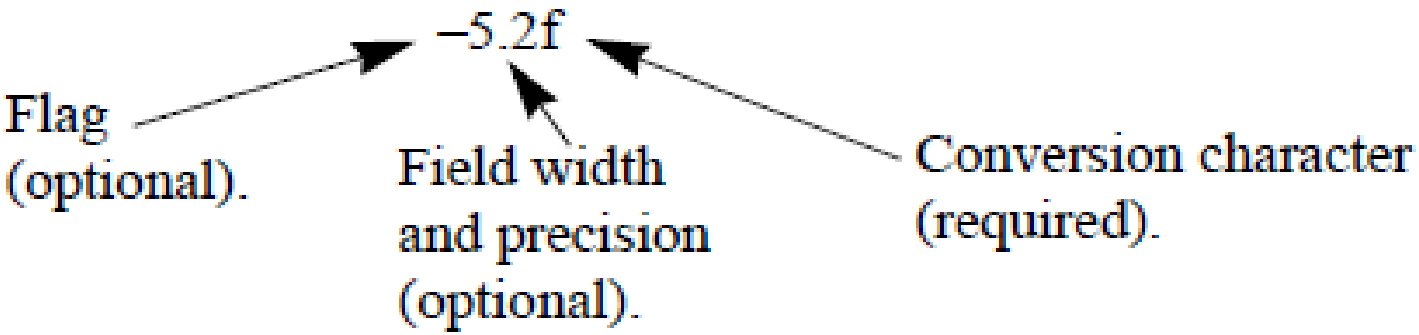
```
fprintf('text as string %-5.2f additional text',
                                    variable_name)
```

The % sign marks the spot where the number is inserted within the text.

Formatting elements (define the format of the number).

The name of the variable whose value is displayed.

The formatting elements are:



The flag, which is optional can be one of the following three:

Character used for flag	Description
– (minus sign)	Left justifies the number within the field.
+ (plus sign)	Prints a sign character (+ or –) in front of the number.
0 (zero)	Adds zeros if the number is shorter than the field.

```
% This program calculates the distance a projectile flies,
% given its initial velocity and the angle at which it is shot.
% the fprintf command is used to display a mix of text and numbers.

v=1584;  % Initial velocity (km/h)
theta=30;  % Angle (degrees)
vms=v*1000/3600;
t=vms*sind(30)/9.81;
d=vms*cosd(30)*2*t/1000;
fprintf('A projectile shot at %3.2f degrees with a velocity
of %4.2f km/h will travel a distance of %g km.\n',theta,v,d)
```

Changing velocity units to m/s.

Calculating the time to highest point.

Calculating max distance.

When this script file (saved as Chapter4Example7) is executed, the display in the Command Window is:

```
>> Chapter4Example7
A projectile shot at 30.00 degrees with a velocity of
1584.00 km/h will travel a distance of 17.091 km.
>>
```

```
x=1:5;
y=sqrt(x);
T=[x; y]
fprintf('If the number is: %i, its square root is: %f\n',T)
```

Create a vector x.

Create a vector y.

Create 2 × 5 matrix T, first row is x, second row is y.

The fprintf command displays two numbers from T in every line.

When this script file is executed the display in the Command Window is:

```
T =
    1.0000    2.0000    3.0000    4.0000    5.0000
    1.0000    1.4142    1.7321    2.0000    2.2361

If the number is: 1, its square root is: 1.000000
If the number is: 2, its square root is: 1.414214
If the number is: 3, its square root is: 1.732051
If the number is: 4, its square root is: 2.000000
If the number is: 5, its square root is: 2.236068
```

The 2 × 5 matrix T.

The fprintf command repeats 5 times, using the numbers from the matrix T column after column.

Using the fprintf command to save output to a file:

Writing output to a file requires three steps:

- Opening a file using the fopen command.
- Writing the output to the open file using the fprintf command.
- Closing the file using the fclose command.

```
fid = fopen('file_name','permission')
```

`fid` is a variable called the file identifier. A scalar value is assigned to `fid` when `fopen` is executed. The file name is written (including its extension) within single quotes as a string. The permission is a code (also written as a string) that tells how the file is opened. Some of the more common permission codes are:

`'r'`	Open file for reading (default).
`'w'`	Open file for writing. If the file already exists, its content is deleted. If the file does not exists, a new file is created.
`'a'`	Same as `'w'`, except that if the file exists the written data is appended to the end of the file.

If a permission code is not included in the command, the file opens with the default code `'r'`. Additional permission codes are described in the help menu.

```
fprintf(fid,'text %-5.2f additional text',variable_name)
```

`fid` is added to the `fprintf` command.

```
fclose(fid)
```

```
% Script file in which fprintf is used to write output to files.
% Two conversion tables are created and saved to two different files.
% One converts mi/h to km/h, the other converts lb to N.
clear all
Vmph=10:10:100;
Vkmh=Vmph.*1.609;
TBL1=[Vmph; Vkmh];
Flb=200:200:2000;
FN=Flb.*4.448;
TBL2=[Flb; FN];
fid1=fopen('VmphtoVkm.txt','w');
fid2=fopen('FlbtoFN.txt','w');
fprintf(fid1,'Velocity Conversion Table\n \n');
fprintf(fid1,'      mi/h            km/h    \n');
fprintf(fid1,'   %8.2f        %8.2f\n',TBL1);
fprintf(fid2,'Force Conversion Table\n \n');
fprintf(fid2,'     Pounds         Newtons    \n');
fprintf(fid2,'   %8.2f        %8.2f\n',TBL2);
fclose(fid1);
fclose(fid2);
```

Creating a vector of velocities in mph.

Converting mph to km/h.

Creating a table (matrix) with two rows.

Creating a vector of forces in lb.

Converting lb to N.

Creating a table (matrix) with two rows.

Open a txt file named VmphtoVkm.

Open a txt file named FlbtoFN.

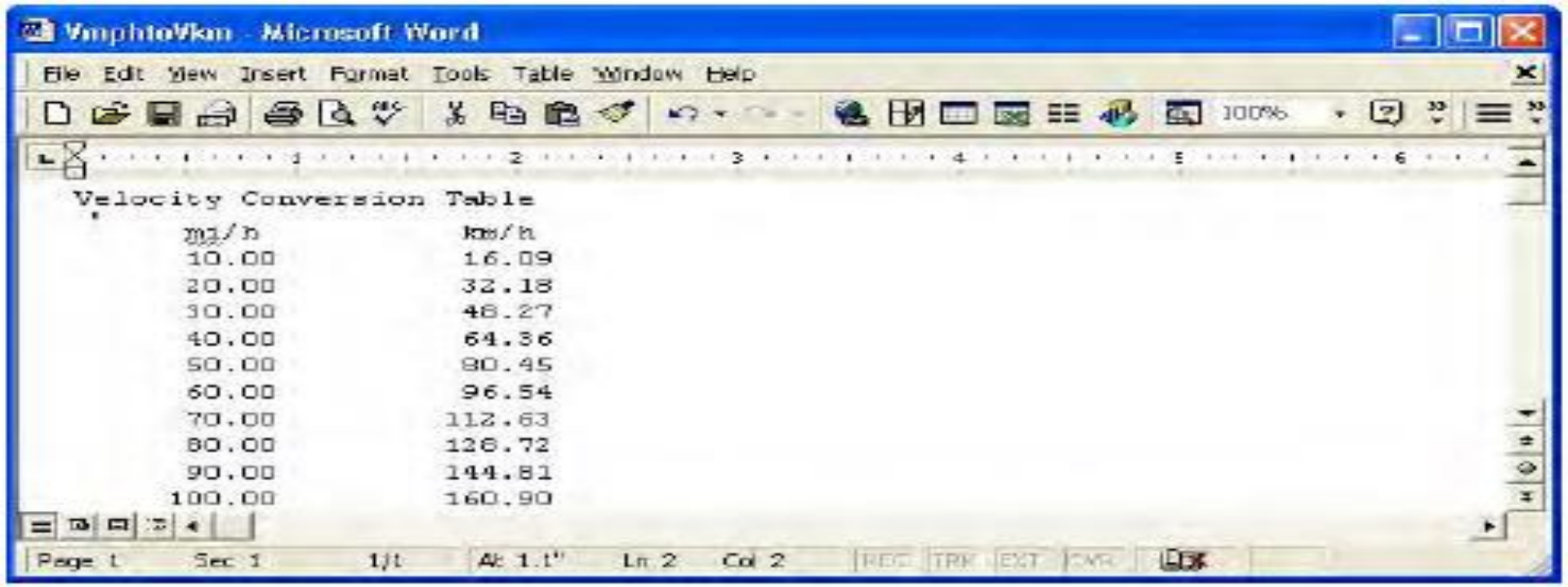
Writing a title and an empty line to the file fid1.

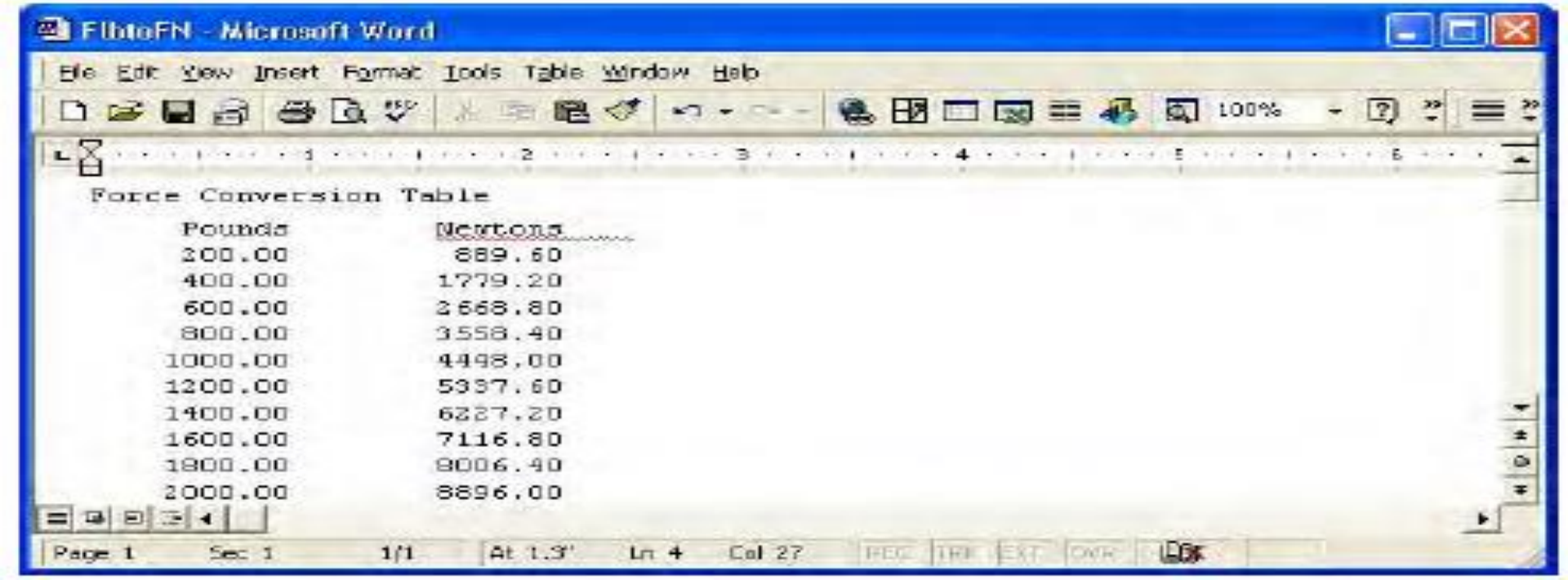
Writing two columns heading to the file fid1.

Writing the data from the variable TBL1 to the file fid1.

Writing the force conversion table (data in variable TBL2) to the file fid2.

Closing the files fid1 and fid2.


VmphtoVkm - Microsoft Word
File Edit View Insert Format Tools Table Window Help
100%
Velocity Conversion Table
mi/h km/h
10.00 16.09
20.00 32.18
30.00 48.27
40.00 64.36
50.00 80.45
60.00 96.54
70.00 112.63
80.00 128.72
90.00 144.81
100.00 160.90
Page 1 Sec 1 1/1 At 1.1" Ln 2 Col 2


FlbtoFN - Microsoft Word
File Edit View Insert Format Tools Table Window Help
100%
Force Conversion Table
Pounds Newtons
200.00 889.60
400.00 1779.20
600.00 2668.80
800.00 3558.40
1000.00 4448.00
1200.00 5337.60
1400.00 6227.20
1600.00 7116.80
1800.00 8006.40
2000.00 8896.00
Page 1 Sec 1 1/1 At 1.3" Ln 4 Col 27

Save

```
save file_name
```
or
```
save('file_name')
```

```
save file_name var1 var2
```
or
```
save('file_name','var1','var2')
```

- The save command can also be used for saving in ASCII format, which can be read by applications outside MATLAB.

```
>> V=[3 16 -4 7.3];
>> A=[6 -2.1 15.5; -6.1 8 11];
>> save -ascii DatSavAsci
```

Create a 1 × 4 vector V.

Create a 1 × 4 matrix A.

Save variables to a file named DatSavAs.

Load

- The load command can be used for retrieving variables that were saved with the save command back to the workspace, and for importing data that was created with other applications and saved in ASCII format or in text (.txt) files. Variables that were saved with the save command in .mat files can also be retrieved with the command.

```
load file_name
```

or

```
load('file_name')
```

```
load file_name var1 var2
```

or

```
load('file_name','var1','var2')
```

```
>> DfT=load('DataFromText.txt')

DfT =

   56.0000   -4.2000
    3.0000    7.5000
   -1.6000  198.0000

>> load DataFromText.txt

>> DataFromText

DataFromText =

   56.0000   -4.2000
    3.0000    7.5000
   -1.6000  198.0000
```

Load the file `DataFrom-Text` and assign the loaded data to the variable `Dft`.

Use the load command with the file `DataFromText`.

The data is assigned to a variable named `DataFromText`.

Importing Data from Excel

```
variable_name=xlsread('filename')
```

- `'filename'` (typed as a string) is the name of the Excel file. The directory of the Excel file must either be the current directory, or listed in the search path.
- If the Excel file has more than one sheet, the data will be imported from the first sheet.

When an Excel file has several sheets, the `xlsread` command can be used to import data from a specified sheet. The form of the command is then:

```
variable_name=xlsread('filename','sheet_name')
```

- The name of the sheet is typed as a string.

Another option is to import only a portion of the data that is in the spreadsheet. This is done by typing an additional argument in the command:

```
variable_name=xlsread('filename','sheet_name','range')
```

- The `'range'` (typed as a string) is a rectangular region of the spreadsheet defined by the addresses (in Excel notation) of the cells at opposite corners of the region. For example, `'C2:E5'` is a 4 by 3 region of rows 2, 3, 4, and 5 and columns *C*, *D*, and *E*.

Exporting Data to Excel

Exporting data from MATLAB to an Excel spreadsheet is done by using the `xlswrite` command. The simplest form of the command is:

```
xlswrite('filename',variable_name)
```

- `'filename'` (typed as a string) is the name of the Excel file to which the data is exported. The file must be in the current directory. If the file does not exist, a new Excel file with the specified name will be created.
- `variable_name` is the name of the variable in MATLAB with the assigned data that is being exported.
- The arguments `'sheet_name'` and `'range'` can be added to the `xlswrite` command to export to a specified sheet and to a specified range of cells, respectively.

Laboratory Session

Do sample applications in Chapter 4 of the textbook.

Homework #6

Not later than the next week:

Solve problems 3, 4, and 8 from the Chapter 4 of the textbook using Matlab.