MATLAB - DATA OUTPUT

Data export *oroutput* in MATLAB means to write into files. MATLAB allows you to use your data in another application that reads ASCII files. For this, MATLAB provides several data export options.

You can create the following type of files:

- Rectangular, delimited ASCII data file from an array.
- Diary *orlog* file of keystrokes and the resulting text output.
- Specialized ASCII file using low-level functions such as fprintf.
- MEX-file to access your C/C++ or Fortran routine that writes to a particular text file format.

Apart from this, you can also export data to spreadsheets.

There are two ways to export a numeric array as a delimited ASCII data file −

- Using the **save** function and specifying the **-ascii** qualifier
- Using the **dlmwrite** function

Syntax for using the save function is:

```
save my_data.out num_array -ascii
```

where, *my_data.out* is the delimited ASCII data file created, *num_array* is a numeric array and **−ascii** is the specifier.

Syntax for using the **dlmwrite** function is:

```
dlmwrite('my_data.out', num_array, 'dlm_char')
```

where, *my_data.out* is the delimited ASCII data file created, *num_array* is a numeric array and *dlm_char* is the delimiter character.

Example

The following example demonstrates the concept. Create a script file and type the following code −

```
num_array = [ 1 2 3 4 ; 4 5 6 7; 7 8 9 0];
save array_data1.out num_array -ascii;
type array_data1.out
dlmwrite('array_data2.out', num_array, ' ');
type array_data2.out
```

When you run the file, it displays the following result −

```
   1.0000000e+00   2.0000000e+00   3.0000000e+00   4.0000000e+00
   4.0000000e+00   5.0000000e+00   6.0000000e+00   7.0000000e+00
   7.0000000e+00   8.0000000e+00   9.0000000e+00   0.0000000e+00

1 2 3 4
4 5 6 7
7 8 9 0
```

Please note that the save -ascii command and the dlmwrite function does not work with cell arrays as input. To create a delimited ASCII file from the contents of a cell array, you can

- Either, convert the cell array to a matrix using the **cell2mat** function

- Or export the cell array using low-level file I/O functions.

If you use the **save** function to write a character array to an ASCII file, it writes the ASCII equivalent of the characters to the file.

For example, let us write the word 'hello' to a file −

```
h = 'hello';
save textdata.out h -ascii
type textdata.out
```

MATLAB executes the above statements and displays the following result. which is the characters of the string 'hello' in 8-digit ASCII format.

```
1.0400000e+02   1.0100000e+02   1.0800000e+02   1.0800000e+02   1.1100000e+02
```

Writing to Diary Files

Diary files are activity logs of your MATLAB session. The diary function creates an exact copy of your session in a disk file, excluding graphics.

To turn on the diary function, type −

```
diary
```

Optionally, you can give the name of the log file, say −

```
diary logdata.out
```

To turn off the diary function −

```
diary off
```

You can open the diary file in a text editor.

Exporting Data to Text Data Files with Low-Level I/O

So far, we have exported numeric arrays. However, you may need to create other text files, including combinations of numeric and character data, nonrectangular output files, or files with non-ASCII encoding schemes. For these purposes, MATLAB provides the low-level **fprintf** function.

As in low-level I/O file activities, before exporting, you need to open or create a file with the **fopen** function and get the file identifier. By default, fopen opens a file for read-only access. You should specify the permission to write or append, such as 'w' or 'a'.

After processing the file, you need to close it with **fclose***fid* function.

The following example demonstrates the concept −

Example

Create a script file and type the following code in it −

```
% create a matrix y, with two rows
x = 0:10:100;
y = [x; log(x)];

% open a file for writing
fid = fopen('logtable.txt', 'w');

% Table Header
fprintf(fid, 'Log     Function\n\n');
```

```
% print values in column order
% two values appear on each row of the file
fprintf(fid, '%f    %f\n', y);
fclose(fid);
% display the file created
type logtable.txt
```

When you run the file, it displays the following result –

```
Log     Function

0.000000    -Inf
10.000000    2.302585
20.000000    2.995732
30.000000    3.401197
40.000000    3.688879
50.000000    3.912023
60.000000    4.094345
70.000000    4.248495
80.000000    4.382027
90.000000    4.499810
100.000000    4.605170
```