ECCE3245 Computer Aided Tools

1. **File Input/output**

a. User Input

User input can be obtained with the following input command, which displays a prompt and waits for a user response.

```
x= input('Insert an integer value:');
```

There is another function that collects data via mouse clicks and it as follows:

```
[x, y] = ginput(5);
```

b. Output to Screen

When we omitted semi colon from MATLAB assignment the result will be displayed on screen. There is a function called 'disp' that display the value of variables and strings on the screen. The syntax is as follows:

```
a = rand(2);
disp(' The value of variable a is :')
disp(a)
```

Moreover, the value of a could be displayed by using fprintf command

```
fprintf('%d %d\n',a)
```

c. **File Opening**

To open a file, use a command similar to the following

```
fid = fopen(filename, 'mode');
```

Mode is divided into:

(a) Writing mode which is represented by ('w').
(b) Reading mode and represented by ('r').

fid is an integer file identifier obtained from FOPEN. fid will take an integer value. If the value is -1, that indicates that a major problem occurred during opening the file, such as the file could not be opened.

d. **File writing (Input to file)**

For such purpose we use the following command:

```
fid = fopen(filename,'w');
```

The 'w' mode will create a file with the specified name even it does not exist. Where

'r' = read only
'w' = write (create if needed)
'a' = append (create if needed)
'r+' = read and write (do not create)
'w+' = create for read and write
'a+' = read and append (create if needed)

Example for create file for writing

```
fid = fopen('datafile','w'); % create a file called datafile
x= rand(5,5); % create matrix of (5X5)
% Write formatted data to file.
fprintf(fid,'%6.2f  %12.8f %12.8f% 12.8f %12.8f\n',x);
fclose(fid) % close the created file
```

The %f means the input data has float format. The special formats \n,\r,\t,\b,\f can be used to produce linefeed, carriage return, tab, backspace, and form feed characters respectively.

Another example by using function

```
function f= create_file(n,m)
f= rand(n,m);
fid = fopen('testdata.dat','w+');
   %print row by row
      for i = 1:n
         fprintf(fid,'%8.4f', f(i,:));
         fprintf(fid,'\n');
      end
fclose(fid);
```

e. **Reading from a File (Output)**

For such purpose we use the following command:

```
fid = fopen(filename,'r');
```

The fid will retrieve value of -1 when we use 'r' mode if the specified file name does not exist. Therefore we should be aware of the filename as well as the path directory where it saved.

```
fid = fopen('datafile.m','r'); % create a file called datafile
A= fscanf(fid,'%f%f%f%f%f'); %reads 5-digit decimal integers.
x= vec2mat(A,5);% convert the displayed vector into matrix
disp(x);% display the value of x
fclose(fid) % close the created file
```

There another command for reading data from file rather than the above mentioned, one of them is text read command. Textread command is used for reading from text file. The syntax of the command is as follows:

```
A = textread('filename','format');
```

f. **Saving**

For saving workspace variables to disk we should use the following command

```
>> save filename  variable  -ascii;
```

g. **Other commands**

Display all .mat files in directory

```
ls *.mat
```

The "load" command loads the data stored in the named binary file into memory.

```
load data1.mat;
```

If we want to get rid of this file, we can use the "delete" command.

```
delete data1.mat;   ls *.mat
```

We can then change to another directory using the "cd" command.

```
cd ..
```

h. Various Examples with documentation

Example 1

```
%=====================================================================
% electrical.m -- Compute currents in an electrical circuit
%
% Matrices : Resist  = Rows represents resistors in each loop.
%          : Voltage = Voltage gain in each loop provided by battery.
%          : Current = Current in each loop.
%=====================================================================

% Setup matrix for "resistances in circuit loops"

  Resist = [ 11  -3   0;
             -3  13  -4;
              0  -4  14 ]

% Setup matrix for "voltage gains" in circuit loops

  Voltage = [ 10; 0; 0 ]

% Solve equations and print currents

  Current = Resist\Voltage
% ========================================================
% the end!
```

Example 2

```
%=====================================================================
%leastsq.m -Compute least squares polynomial fit on experimental data
%
% Experiment : x = displacement of spring (cm).
%              f = force in spring (N).
%
% Least squares fit : p(x) = a + b.x + c.x^2
%=====================================================================

% Store force-displacement relationship in matrix "data"
data = [ 5.0    0.0;
         5.5   47.5;
         6.0   90.0;
         6.5  127.5;
         7.0  160.0;
         7.5  187.5;
         8.0  210.0 ];

% Compute terms in least squares matrix and right-hand vector
N = 7;
sumx   = sum(data(:,1));
sumy   = sum(data(:,2));
sumxy  = sum(data(:,1).*data(:,2));
sumx2  = sum(data(:,1).*data(:,1));
```

```
sumx2y = sum(data(:,1).*data(:,1).*data(:,2));
sumx3  = sum(data(:,1).*data(:,1).*data(:,1));
sumx4  = sum(data(:,1).*data(:,1).*data(:,1).*data(:,1));

A = [      N   sumx   sumx2
        sumx  sumx2   sumx3
       sumx2  sumx3   sumx4 ]

B = [ sumy; sumxy; sumx2y]

% Compute and print constants a,b,c

Coefficients = A\B

% ============================================================
% the end!
```

Example 3

```
%  ====================================================
%  expt.m -- Statistical analysis of experimental data.
%
%  ====================================================

% Store experimental results in array

  load expt.dat

% Generate bar plot of experimental results

  bar(expt(:,1), expt(:,2),'b')
  xlabel('Day of Expt');
  ylabel('Pollen Count');

% Compute terms from experimental results.

  [xm, xd] = stat(expt(:,2))

% Create and display mean value of pollen count

  mean_minus = xm(1,1) - xd(1,1);
  mean_plus  = xm(1,1) + xd(1,1);

  data = [  1, xm(1,1), mean_minus, mean_plus;
           15, xm(1,1), mean_minus, mean_plus ];

  hold;
  plot (data(:,1), data(:,2), 'b');
  plot (data(:,1), data(:,3), 'b:');
  plot (data(:,1), data(:,4), 'b:');
  text(1,   xm(1,1) + 10,'Mean Pollen Count');
  text(1,mean_minus + 10,'Mean - Std');
  text(1, mean_plus + 10,'Mean + Std');
```

Example 4

```
%====================================================================
%temperature.m  Compute and plot a graph of Celsius versus Fahrenheit
%                  for the range -50 through 100 degrees Celsius.
%
%====================================================================

% Allocate arrays for Celsius and Fahrenheit temperatures.

  tempC = zeros(2,1);
  tempF = zeros(2,1);

% Compute temperatures at graph end-points.

  tempC(1)  = -50; tempF(1)  = 9*tempC(1)/5 + 32;
  tempC(2)  = 100; tempF(2)  = 9*tempC(2)/5 + 32;

% Plot and label the graph

  plot( tempC, tempF );
  grid;
  xlabel('Temperature (Celsius)');
  ylabel('Temperature (Fahrenheit)');
  title('Fahrenheit versus Celsius Temperature Conversion');
```

Example 5

```
%====================================================================
% Compute square root function
%
%====================================================================
function sqroot(x)
% SQROOT Compute square root by Newton's method

% Check that value of function argument is positive.

  if x <= 0,
     error('In sqroot() : argument x must be positive');
  end;

% Initial guess

  xstart = x/2;

% Iteration loop to compute square root
  for i = 1:100
      xnew = ( xstart + x/xstart)/2;  % new estimate of square root.
      disp(xnew);                     % print xnew.
      if abs(xnew-xstart)/xnew<eps,% check convergence of iterations.
         break,                          % iterations.
      end;
      xstart = xnew;               % update estimate of square root.
  end
```

Example 6

```
%=====================================================================
%                        Statistical Analysis
%
%=====================================================================

    function  [mean, stdev] = stat(x)

    % STAT  Mean and standard deviation
    % For a vector x, stat(x) returns the
    % mean and standard deviation of  x.
    % For a matrix x, stat(x) returns two row vectors containing,
    % respectively, the mean and standard deviation of each column.

    [m  n] = size(x);

    if m == 1
       m = n;     % handle case of a row vector
    end

    mean  = sum(x)/m;
    stdev = sqrt(sum(x.^ 2)/m - mean.^2);
```