Matlab1 hands-on

I. Run the commands listed below, then check the results.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Introducing Matlab (adapted from http://www.cns.nyu.edu/~eero)

clear all                       % clear all variables
close all                       % close all figures
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% the basic objects in matlab : scalars, vectors, and matrices...

N       = 5                         % a scalar
v       = [1 0 0]                   % a row vector
v       = [1;2;3]                   % a column vector
v       = v'                        % transpose a vector
                                            (row to column or column to row)
v       = [1:.5:3]                  % a vector in a specified range:
v       = pi*[-4:4]/4               %       [start:stepsize:end]
v       = []                        % empty vector


m       = [1 2 3; 4 5 6]            % a matrix: 1ST parameter is ROWS
                                    %            2ND parameter is COLS
m       = zeros(2,3)                % a matrix of zeros
v       = ones(1,3)                 % a matrix of ones
m       = eye(3)                    % identity matrix
v       = rand(3,1)                 % rand matrix (see also randn)

%--------------------------------------------------
matrix_data = [2 3 4; 5 6 7; 1 2 3];
                                    %       2     3     4
                                    %       5     6     7
                                    %       1     2     3

v       = [1 2 3];                  % access a vector element
v(3)                                % vector(number)


m       = [1 2 3; 4 5 6]
m(1,3)                              % access a matrix element
                                    % matrix(rownumber, columnnumber)
m(2,:)                              % access a matrix row (2nd row)
m(:,1)                              % access a matrix column (1st row)


size(m)                             % size of a matrix
size(m,1)                           % number rows
size(m,2)                           % number of columns

m1      = zeros(size(m))            % create a new matrix with size of m

who                                 % list of variables
whos                                % list/size/type of variables


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
% (3) Simple operations on vectors and matrices

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (A) Pointwise (element by element) Operations:

% addition of vectors/matrices and multiplication by a scalar
% are done "element by element"
a         = [1 2 3 4];                    % vector
2 * a                                     % scalar multiplication
a / 4                                     % scalar multiplication
b         = [5 6 7 8];                    % vector
a +  b                                    % pointwise vector addition
a - b                                     % pointwise vector addition
a .^ 2                                    % pointise vector squaring (note .)
a .* b                                    % pointwise vector multiply (note .)
a ./ b                                    % pointwise vector multiply (note .)

log( [1 2 3 4] )                  % pointwise arithmetic operation
round( [1.5 2; 2.2 3.1] )         % pointwise arithmetic operation


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% (B) Vector Operations (no for loops needed)
% Built-in matlab functions operate on vectors, if a matrix is given,
% then the function operates on each column of the matrix

a         = [1 4 6 3]                     % vector
sum(a)                                    % sum of vector elements
mean(a)                                   % mean of vector elements
var(a)                                    % variance
std(a)                                    % standard deviation
max(a)                                    % maximum


a         = [1 2 3; 4 5 6]                % matrix
mean(a)                                   % mean of each column
max(a)                                    % max of each column
max(max(a))                               % to obtain max of matrix
max(a(:))                                 %        or...


%%%%%%%%%%%%%%%%%%%%%%%%
% (C) Matrix Operations:

[1 2 3] * [4 5 6]'                        % row vector 1x3 times column vector 3x1
                                          % results in single number, also
                                          % known as dot product or inner product

[1 2 3]' * [4 5 6]                        % column vector 3x1 times row vector 1x3
                                          % results in 3x3 matrix, also
                                          % known as outer product

a         = rand(3,2)                     % 3x2 matrix
b         = rand(2,4)                     % 2x4 matrix
c         = a * b                         % 3x4 matrix

a         = [1 2; 3 4; 5 6]       % 3 x 2 matrix
b         = [5 6 7];                      % 3 x 1 vector
b * a                                     % matrix multiply
```

```
a' * b'                                   % matrix multiply


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%(4) Plotting

x         = [0 1 2 3 4];                  % basic plotting
plot( x );
plot( x, 2*x );
axis( [0 8 0 8] );

x         = [0 1 2 3 4];                  % basic stem
stem( x );
stem( x, 2*x );
axis( [0 8 0 8] );

x         = pi*[-24:24]/24;
plot( x, sin(x) );
xlabel( 'radians' );
ylabel( 'sin value' );
title( 'dummy' );

figure;                                   % multiple functions in separate graphs
subplot( 1,2,1 );
plot( x, sin(x) );
axis square;
subplot( 1,2,2 );
plot( x, 2.*cos(x) );

figure;                                   % multiple functions in single graph
plot( x,sin(x) );
hold on;
plot (x, 2.*cos(x), '--' );
legend( 'sin', 'cos' );
hold off;
```

II. Make the Matlab program as instructed below.

1. Make a 2D matrix A with the contents listed below.

    1 2 3

    4 5 6

    7 8 9

2. Using Matlab commands, get the size of A and list more detailed information on A. (size, whos)

3. Display the value of center element (5) of A in the 'Command Window'.

4. Display the last column of A.

5. Display the last row of A.

6. Transpose the last row into column vector then display it.

7. Append a row vector [10 11 12] at the end of last row of A.

8. Turn the rows upside down by using matrix index, i.e. last row will become first row and vice versa. (hint: use 'end')

9. Take 3x2 sub-array from A and move it into a new array B.

10. Create 8x8 array C and fill it with random numbers. The minimum value of C should be greater than or equal to 0, and its maximum value should be less than or equal to 255. Then convert the data type so that each element of the array be unsigned integer (1 byte). (hint: >> help uint8)

11. Get the size, maximum and minimum values of C.

12. Create a new array D so that it has the same size of array C, then fill the array D with zeros.

13. Find the elements in C whose values are greater than or equal to 128, and move those elements into D. (hint: find())

14. We obtained two sinusoidal signals x1(t) and x2(t) during 100 seconds ($0 \le t \le 100$).

$$x_1(t) = 7\cos(2\pi(0.5)t)$$
$$x_2(t) = 3\cos(2\pi(0.01)t + 0.25\pi)$$

1) Make a 2 by 2 subplot and display x_1(t) at the location of subplot1 (hint:subplot(2,2,1))

2) Display plot of signal in the subplot2 (hint: subplot(2,2,2))

3) Display x_1(t) + x_2(t) in the subplot3

4) Have all those three plots overlaid and display them together in subplot4. Each plot must be displayed in different colors. (hint: plot(x, y, ‘r’), hold on, hold off)