SOMETHING MORE ABOUT MATRICES

1. Generation of Random Numbers

```
% Generate a single number between 0 and 1
      >> rand
% Generate a single integer between m and n
      >> rand ([imin imax])
% generate n elements row vector of random numbers between 0 and 1
      >> rand (1,n)
% generate nxn matrix of random numbers between 0 and 1
      >> rand (n)
% generate mxn matrix of random integers between m and n
      >> rand ([imin imax], m, n)
% generate mxn matrix of random numbers between 0 and 1
      >> rand (m,n)
% generate row vector n elements that are random permutation of integers between 0 and 1
      >> randperm (n)
```

2. Inverse of a Matrix

```
      >> A=[2 1 4; 4 1 8; 2 -1 3]
      >> B = inv (A)
```

3. Solving System of Linear Equations

```
% Example (3.1) Page 71
          4x – 2y + 6z = 8
          2x + + 8y 2z = 4
          6x + + 10y 3z = 0
     by solving  AX = B    Let C = inv (A)
     so  X = A\B  or  X = B / A   or   X = C B

     >> A = [4 -2 6; 2 8 2; 6 10 3];
     >> B = [8; 4; 0];
     >> X = inv(A) * B        % execute

     % Example
      >>    A = [magic(3)   eye(3)]
      A =   8     1     6     1     0     0
            3     5     7     0     1     0
            4     9     2     0     0     1
     >>  R = rref (A)
      R =     1.0000         0         0         0.1472   -0.1444    0.0639
                0            1.000     0        -0.0611    0.0222    0.1056
                0            0         1.0000   -0.0194    0.1889   -0.1028
     >> inv_A = R (: , 4 : end)              % gives the inverse of A
```

4. Row Reduced Echelon form of a Matrix

```
     >>   A = magic (3);
     >>   A ( : , 4) = [1; 1; 1]
     A =  8     1     6     1
          3     5     7     1
          4     9     2     1
     >> R = rref (A)
     A =  1     0     0     0.0667
          0     1     0     0.0667
          0     0     1     0.0667
```

```
>> X = R ( : , end)     % this gives the solution of the system of linear equations
```

5. Gauss Jordan Elimination Method to Solve System of Linear Equations

$$x_1 + x_2 + 5x_3 = 6$$
$$2x_1 + x_2 + 8x_3 = 8$$
$$x_1 + 2x_2 + 7x_3 = 10$$
$$-x_1 + x_2 - x_3 = 2.$$

```
>> A = [1 1 5; 2 1 8; 1 2 7; -1 1 -1]
>> B = [6; 8; 10; 2]
>> M = [A B]
>> R = rref (M)
   R = 1     0     3     2
       0     1     2     4
       0     0     0     0
       0     0     0     0
% system has infinite solution
```

Two Dimensional Plots

1. Simple Line Plot

```
% Example 1
   X = [1 2 3 5 7 7.5 8 10];
   Y = [2 6.5 7 7 5.5 4 6 8];
   plot (X,Y)
% Example 2
   X = 0:pi/100:2*pi;   % 0 to 2*pi with steps pi/100
   Y = sin(x);
   Plot (X,Y)
% Example 3
   x=linspace(0,2*pi); % a lin vect between 0 to 2pi with built-in 100 points
   y=sin(x);
   plot(x,y)
   xlabel('x')
   ylabel('sin(x)')
   title('plot of sine function')

% Example 4    ( all three characteristics (line color, style, and marker))
   x = linspace(0,2*pi,50);  % a lin vect between 0 to 2pi with 50 points
   y = sin(x);
   plot(x,y,':')   % dotted line, colour built-in blue
   hold on
   y2 = cos(x);
   plot(x,y2,'--ro')  % dashed line with colour red and marker circle
   hold off
```

2. Additional Options

>> plot (x, y, 'line specifiers', 'Property Name', Property Value)

Line specifiers defines type and colour of the line and the marker
Property Name defines the line width
Property value defines the size of the marker

Line Specifier : solid '-', dashed '--', dotted ':', dash-dot '-.'
Line Colour : red 'r', green 'g', blue 'b', yellow 'y', black 'k', magenta 'm'
Marker Type : + , o, *, ., etc. square 's' diamond 'd' star 'p' etc.

Try to understand these commands

```
plot(x,y)
plot(x,y,'r')
plot(x,y,'--y')
plot(x,y,'*')
plot(x,y,'g:d')
```

```
%Example 5
   x = -pi:pi/10:pi;
   y = tan(sin(x)) - sin(tan(x));

   figure
   plot(x,y,'--gs',...
       'LineWidth',2,...
       'MarkerSize',10,...
       'MarkerEdgeColor','y',...
       'MarkerFaceColor',[0.5,0.5,0.5])
```