Sensors and Control for Mechatronics Systems
Tutorial 1
Quick Introduction to MATLAB

Section 1 : Introduction

1.1 MATLAB is a high level programming language for numerical computations, visualization and application development.

1.2 All variables are multidimentional arrays, regardless of the type of data.

Section 2 : Matrices and Arrays

2.1 An array can be created in multiple ways

Eg :

```
>> a = [1, 2, 3, 4]

a =

    1    2    3    4

>> b = zeros(3)

b =

    0    0    0
    0    0    0
    0    0    0

>> c = [1, 2, 3; 4, 5, 6; 7, 8, 9]

c =

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

2.2 Array indices start with 1 in MATLAB, which is different from programming languages like C, Java and Python.

Can use either row and column numbers as the indices.

```
>> c(2,3)

ans =

    6
```

Or linear indexing can be used (Traverse down the columns in order).

```
>> c(8)

ans =

     6
```

* Note that parenthesis are used to specify the index, not the square brackets.

Section 3 : Functions

3.1 Functions are called by their name, succeeded by argumets passed inside parenthesis.

```
>> d = max(a)

d =

     4
```

3.2 When there are multiple output arguments, use square brackets.

```
>> [maxA, index] = max(a)

maxA =

     4


index =

     4
```

3.3 If a function does not need input arguments, simply call it using its name.

```
>> pwd

ans =

    '/home/janindu/Documents/Tutorial 1'
```

3.4 Text arguments are enclosed with single quotes.

```
>> disp('Matlab tutorial')
Matlab tutorial
```

3.5 To view function documentation, use help

```
>> help mean
```

Section 4 : 2-D and 3-D Plots

4.1 Vector Y can be plotted against vector X using plot function.

4.2 Create vector X using colon operator.

```
>> X = 0 : pi/360 : 2*pi;
```

4.3 Create vector Y with sine value for corresponding X

```
>> Y = sin(X);
```

4.4 Plot Y against X.

```
>> plot(X,Y);
```

4.5 surf function can be used to create 3-D plots

4.6 Plot z = sin(x) – cos(y)

```
>> [x,y] = meshgrid(-pi:pi/360:pi);
>> z = sin(x) - cos(y);
>> surf(x,y,z);
```

Section 5 : Scripts

5.1 A script is a file with a .m extention that contains MATLAB commands and function calls. You can declare your own functions inside MATLAB scripts.

5.2 Create new script called matrixMultiplier.

```
>> edit matrixMultiplier
```

5.3 Declare a function called multiply which has two input arguments and one output argument

```
function Y = matrixMultiplier(A,B)

end
```

5.4 Use conditional statements to check if matrix dimensions match. If they do, multiply A and B and return it as Y. Otherwise return A.

```
function Y = matrixMultiplier(A,B)
[rowA, colA] = size(A);
[rowB, colB] = size(B);

if colA == rowB
   Y = A*B;
else
   msg = “Dimension mismatch”;
   error(msg);
end
end
```

5.5 Test your script. Declare the two matrices you want to multiply as A and B first.

```
>> matrixMultiplier(A,B)
```

5.6 Change the function body such that before multiplying, the program iterates through A and sets all zero valued elements to 1.

```
[rowA, colA] = size(A);
[rowB, colB] = size(B);

if colA == rowB
   for i = 1:rowA
      for j = 1:colA
         if A(i,j) == 0
            A(i,j) = 1;
         end
      end
   end
   Y = A*B;
else
   Y = B;
end
```

Note : A comprehensive tutorial on MATLAB can be found on the MathWorks website : https://au.mathworks.com/support/learn-with-matlab-tutorials.html