LEBANESE AMERICAN UNIVERSITY
School of Engineering
Department of Electrical and Computer Engineering

ELE443 Control System LAB

Fall 2013

Lecture 1: Introduction to MATLAB

Joe Khalifeh

Introduction

- MATLAB=Matrix Laboratory
- MATLAB is a high-performance language for technical computing.
- It integrates computation, visualization, and programming in an easy-to-use environment.

- Typical uses are:
  - Math and computation
  - Algorithm development
  - Data acquisition
  - Modeling
  - simulation, and prototyping
  - Data analysis, exploration, and visualization
  - Scientific and engineering graphics
  - Application development, including graphical user interface building

MATLAB System

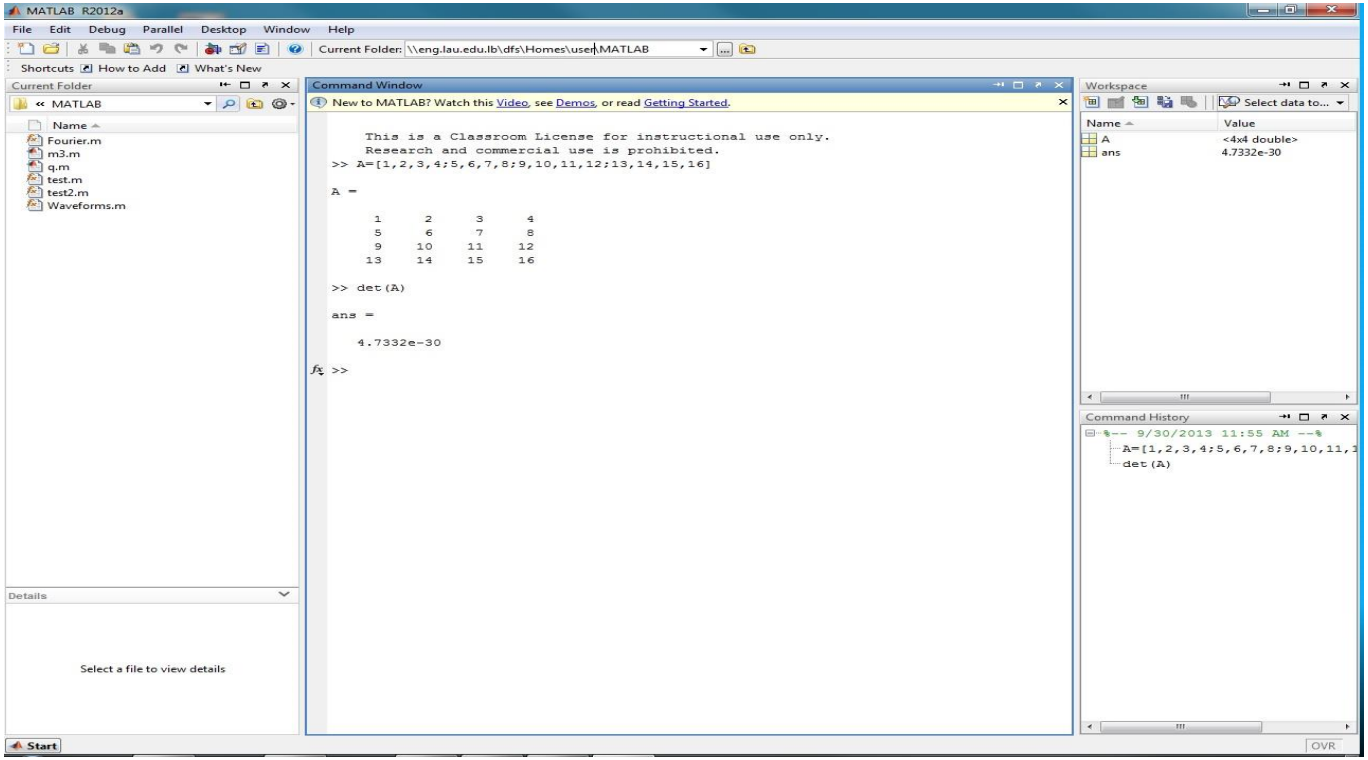
- MATLAB System is formed of 5 main parts:
  - Desktop Tools and Development Environment
  - The MATLAB Mathematical Function Library
  - The MATLAB Language
  - Graphics
  - The MATLAB External Interfaces/API

MATLAB Toolboxes

- There are many toolboxes in MATLAB:
  - Control Systems Toolbox
  - Communication Toolbox
  - Curve Fitting Toolbox
  - Filter Design Toolbox
  - Statistics Toolbox
  - …
- In addition to Simulink which simulates systems using block diagrams

MATLAB Windows

- Command Window
- Current Directory
- Workspace
- Command History



Basic Notations

- Semicolon(;): If a semicolon is typed after a command, then the command is executed without displaying the output.
- Comments(%): Similarly to high level programming languages, comments in MATLAB codes are written after typing percent sign %
- clear: It clears all variables in workspace
- clear A B: Clears variables A and B from workspace
- clc: Clears the command window and homes the cursor. It doesn't affect workspace variables
- close: Closes the current figure window
- help plot: Gives information about the use and the arguments of a function. In this case, it gives information about the function "plot"
- exit: Exit MATLAB

Arithmetic Operators

Symbol	Operation
+	Addition
-	Subtraction
*	Multiplication
/	Division
\	Left division
^	Power
'	Complex conjugate transpose
( )	Specify evaluation order

Arithmetic Operations

- MATLAB as a calculator:
  - Simplest way to use MATLAB
    - Type command (mathematical expression)
    - Press Enter Key
    - Command executed and then is displayed
      - ans= (result)
  - Example
    - cos(pi/2)

      ans = 0

Display Format

- The number format in command window can be modified using the command format
- The default format in the command window is the short representation of numbers.

Command	Description	Example
format short	Scaled fixed point format, with 5 digits	3.1416
format long	Scaled fixed point format, with 15 digits for double; 7 digits for single.	3.14159265358979

Display Format

Command	Description	Example
format short eng	Engineering format that has at least 5 digits and a power that is a multiple of three	3.1416e+000
format long eng	Engineering format that has exactly 16 significant digits and a power that is a multiple of three	3.14159265358979e+000
format short e	Floating point format, with 5 digits.	3.1416e+000
format long e	Floating point format, with 15 digits for double; 7 digits for single.	3.14159265358979e+000

Elementary functions

- MATLAB has built-in useful elementary functions, and extended lists of elementary functions is provided by MATLAB toolboxes.

- Some useful elementary functions:
  - sqrt, exp, log, log10, log2, cos, ceil, sign

MATLAB Variables

- The MATLAB language works with only a single object type: the MATLAB array.
- All MATLAB variables, including scalars, vectors, matrices, strings, cell arrays, structures, and objects are stored as MATLAB arrays.
- Variables are shown in Workspace.
- Variables can have different data types such as:
  - Complex Double-Precision Matrices
  - Numeric Matrices
  - Logical Matrices
  - MATLAB Strings
  - Empty Arrays

MATLAB Variables

- Variable
  - Name made of a combination of letters and/or digits:
    - Memory location
  - Scalar variables are assigned a numerical value:
    - Stored in memory location
  - Can be used in any MATLAB statement or command

- Variables are assigned using equal operator (=). It assigns a value to a variable
- Example:

```
x=pi/2          f=sin(x)
x =1.5708       f=1
```

MATLAB Variables

- Rules about variable names:
  - Up to 63 characters in MATLAB 7 (31 in MATLAB 6.x).
  - Can contain letters, digits and underscore.
  - Must begin with a letter.
  - MATLAB is case sensitive.
  - Avoid using names of built-in functions or predefined variables.
- Predefined variables
  - pi = the number π
  - Inf =Infinity
  - realmax=Largest positive floating point number
  - realmin=Smallest positive floating point number
  - i = sqrt(-1)
  - j = I
  - NaN= (Not a Number) used by MATLAB when it cannot define a valid numerical value, such as 0/0.
  - Eps = Spacing of floating point numbers = 2^{-52}

MATLAB Variables

- Useful commands for managing variables

Command	Description
clear	Clear variables and functions from memory.
clear x y	Clear the variables specified.
who	List current variables.
whos	List current variables, long form.
load	Load workspace variables from disk.
save	Save workspace variables to disk.

Creating Arrays in MATLAB

- Array:
  - Fundamental form used to store and manipulate data.
  - Arranged in rows and/or columns.
  - Include data of different types.
  - Arrays are n-dimensional:
    - One-Dimensional (Vector)
    - Two-Dimensional (Matrix)
    - N-Dimensional

Arrays

- Array constructor [ ]
  - An array of elements (Vector or Matrix) is created using brackets [ ]
- Example:
  - V=[1 2 3 5] creates a horizontal vector
  - Similarly, V=[1,2,3,5]
- A Comma or a Blank separate between elements in two columns of a matrix or vector

Creating Vectors

- When vector elements are specified element by element, a vector can be defined as follows:

- Row vector:
  - V=[1 2 3 5]

    V = 1 2 3 5
- Column Vector:
  - Elements in a column vector are separated using semicolon(;)
  - U=[5;2;1]

    U = 5

        2

        1

Creating Vectors

- Vectors with constant spacing:
  - V=start: space :end
  - start= first element, end=last element
  - space= spacing between two consecutive elements

  - V=1:3:13
    V = 1 4 7 10 13

- When space is omitted, default spacing is 1.

Creating Vectors

- Vector with constant spacing of a desired number of elements:

- V=linspace(start,end,# of elements)
  - V=linspace(1,5,3)

    V =1 3 5

- When # of elements is omitted, 100 is used as a default number.

Creating Matrices

- Matrices are two-dimensional arrays.
  - An m-by-n matrix has m rows and n columns
  - All rows must have the same number of elements.
  - In square matrices, m=n.

- Example:

```
A=[1 5 7;8 2 6;4 -2 9]
A = 1    5    7
    8    2    6
    4   -2    9
```

Creating Matrices

- Variables or functions with adequate output size can be used to define matrix elements.

  - x=0;
    y=pi/6;
    z=pi/2;

  - A=[x,y,z]
    A = 0 0.5236 1.5708

  - B=[A;sin(A)]
    B = 0 0.5236 1.5708
        0 0.5 1

Useful matrices

- zeros(M,N)
  - Creates an M-by-N matrix of zeros.
- ones(M,N)
  - Creates an M-by-N matrix of ones.
- eye(N)
  - Creates the N-by-N identity matrix.

3-Dimensional Arrays

- A 3D array may be constructed by *superposition* of 2D arrays.

- Example:
  - A=[1 2 5;7 8 6];
  - B=[8 2 6;7 3 1];

  - C(:,:,1)=A

    C = 1 2 5
        7 8 6
  - C(:,:,2)=B

C(:,:,1) =1 2 5
          7 8 6

C(:,:,2) = 8 2 6
           7 3 1

The Transpose Operation

- In vectors: Switches row (column) to column (row)
- In matrices: Switches columns (rows) to rows (columns)
- Applied by typing ‘ next to a variable.
- Transpose is not defined for N-Dimensional arrays where N>2

- Example:

```
A = [1 2 5; 7 8 6]
A = 1    2    5
    7    8    6
>> A'
ans = 1    7
      2    8
      5    6
```

Array Addressing

- Elements in arrays can be addressed individually or in subgroups.
- In vectors, elements are addressed by their index.
- Vector indices start from 1.

- For example:

  - V=[5 4 8 3 7];

```
V(1)              a=V(5)
ans =5            a = 7
```

Array Addressing

- Elements of N-Dimensional arrays are addressed using N coordinates (arguments).
- Matrices are 2D arrays.
  - A=[5 6 3;8 2 -9]
    A = 5 6 3
    8 2 -9
- The element "-9" is in the 2nd row and 3rd column can be addressed by:
  - A(2,3)
    ans =-9

Array Addressing

- To address sub-matrices in a matrix, we use the colon (:) notation. Consider the following matrix:
  - A=[5 6 9;3 2 7;1 4 8]

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

- The elements of the sub-matrix are in rows (2 to 3), and in columns (1 to 2), this sub-matrix is addressed such that:
  - A(2:3,1:2)

```
ans =  3    2
       1    4
```

Array Addressing

- >> A=[5 6 9;3 2 7;1 4 8]

A =

5	6	9
3	2	7
1	4	8

elements from (1st and 3rd row) and (1st and 2nd column)

- A([1 3],[1 2])

ans =

5	6
1	4

Array Addressing

- To address all elements from a column(s) or a row(s):

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

- Using (:) in the i^{th} dimension selects all elements belonging to this dimension.

```
A(:,2)
ans = 6
      2
      4
```

Modifying array elements

- Modifying array elements can be done by assigning new elements to sub-parts of the array.

- A(2:3,1:2)=[5 8;6 3]

A = 5 6 9
5 8 7
6 3 8

Adding elements to arrays

- Adding new elements to a matrix:
  - Assigning matrices to new positions in a matrix (at positions “outside” matrix dimension)
  - Appending two matrices
  - The added and original matrices should have the same number of rows (columns) if we are appending elements horizontally (vertically).

Adding elements to arrays

- Example: A = 5 6 9
  3 2 7
  1 4 8

- Adding a column to A as a 5th column:

  A(:,5)=[3;7;2]
  A = 5 6 9 0 3
  3 2 7 0 7
  1 4 8 0 2

- Note that the 4th column is automatically created and set to 0, and in this horizontal appending, the number of rows of the original and added matrices are equal.

Adding elements to arrays

- Adding a single element to an array is always allowed (without constraints on the size of the matrix).

A = 5 6 9
3 2 7
1 4 8

- A(5,4)=2

A = 5 6 9 0
3 2 7 0
1 4 8 0
0 0 0 0
0 0 0 2

- New elements are created accordingly to satisfy the new matrix dimension (and are set to 0).

Adding elements to arrays

- Another method of appending elements of two arrays is by assigning a new array whose elements are arrays and not scalars.
  - A=[1 2;5 6]

    A = 1 2
    5 6
  - B=[7;8]

    B = 7
    8
  - C=[A B]

    C = 1 2 7
    5 6 8

Deleting elements from arrays

- Deleting columns or rows from a matrix can be done by assigning the null matrix [ ] to a sub-part of the matrix.
  - A = 5 6 9
    3 2 7
    1 4 8
  - A(:,2)=[ ]
    A = 5 9
    3 7
    1 8

Array Functions

Function	Description	Example
reshape(X,M,N)	Returns the M-by-N matrix whose elements are taken column wise from X.	X=[1 2;3 4]; Y=reshape(X,1,4) Y = 1 3 2 4
diag(v)	Returns a matrix and puts the elements of v in the main diagonal	v=[1 2 3]; A=diag(v) A = 1 0 0 0 2 0 0 0 3
reshape(X,M,N)	Returns the M-by-N matrix whose elements are taken column wise from X.	X=[1 2;3 4]; Y=reshape(X,1,4) Y = 1 3 2 4
[M,N]=size(X)	for matrix X, returns the number of rows and columns in X as separate output variables.	X=[1 2 3;7 5 9]; [M N]=size(X) M=2 & N=3

Simple plot with MATLAB

- Use "plot" command:
  - plot(t, x)
  - Plots the vector "x" against the vector "t"

- Example:
  Plot the function x=2exp(-2t) over the range [0;2]
  - t=0:0.1:2;
  - x=2*exp(-2*t);
  - plot(t,x)
  - grid

Simple plot with MATLAB