MATLAB Tutorial 01

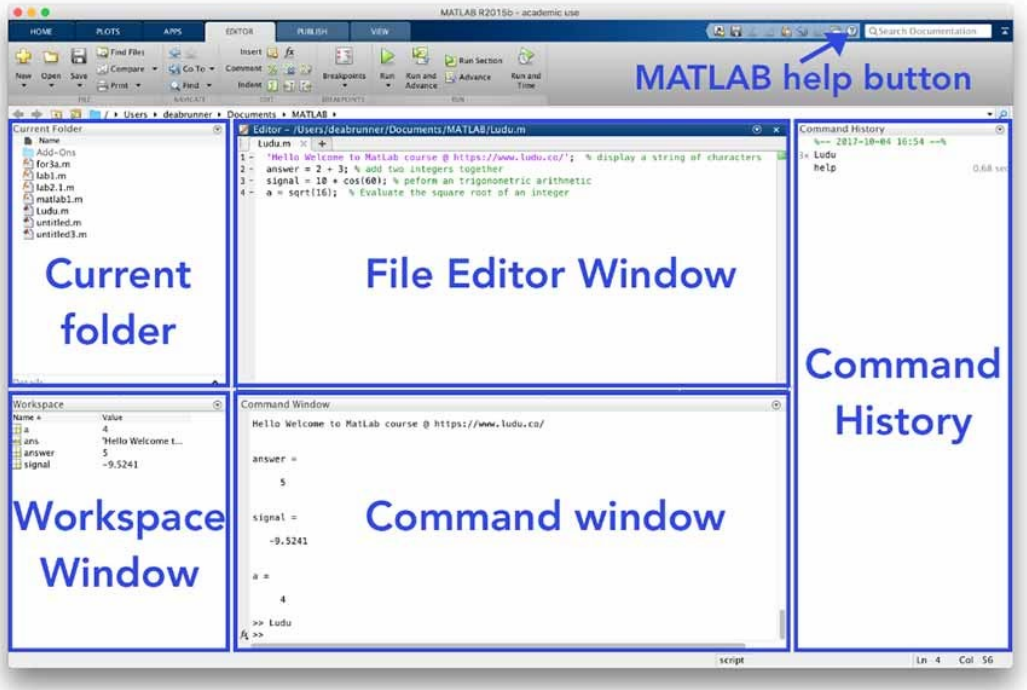
ENME 303 Computational Methods for Engineers

Parham Oveissi

MATLAB Installation Instructions

1. Sign in to myUMBC and then click here.
    1. **If you don't have a UMBC MathWorks account:** Click **Sign In**. In the new page that opens, just below the email text box, click **Create one!** and then proceed to step 2.
    2. **If you already have a UMBC MathWorks account:** Click **Sign In** and type in your credentials. Once you are signed in, proceed directly to step 6.
2. Type in your UMBC email address and fill in all the requested information that is displayed on the page. Click **Create** when you are finished.
3. MathWorks will ask you to verify the UMBC email address you provided in step 2. Follow the directions that are displayed on the page to verify the email address.
4. Fill in the required information that is displayed on the page and click **Create** when you are finished. In the **School/University** section, type in **University of Maryland Baltimore County**. For **Associate Your Account to a License**, type in the appropriate faculty/staff or student stand-alone Activation Key.
5. Click **Downloads application**.
6. Download the MATLAB installer. After downloading the installer, click on it to run it.
7. Enter the credentials for your UMBC MathWorks Account, click **Sign In**, and select **Yes** to accept the license agreement and then click **Next** to continue the setup.
8. Select the appropriate license, click **Next**, and follow the remaining onscreen instructions to complete the installation.

MATLAB Environment



MATLAB Variables

- Is a region of memory containing an array, which is known by a user-specified name.
- Variable names must begin with a letter.
- The MATLAB language is case-sensitive, which means that uppercase and lowercase letters are not the same.
- Use meaningful names for variable to make a program much easier to read and to maintain.
- The simplest way to initialize a variable is to assign it one or more values in an assignment statement:
  - var = 5;

Useful Commands

- clc - clears the Command Window
- clear - clears the workspace
- close all - closes all figures
- who - lists all variables
- whos - detailed list of all variables
- doc - opens the documentation of a command
- help <function>

Useful Built-in Variables

Function	Purpose
`pi`	Contains π to 15 significant digits.
`i, j`	Contain the value i ($\sqrt{-1}$).
`Inf`	This symbol represents machine infinity. It is usually generated as a result of a division by 0.
`NaN`	This symbol stands for not-a-number. It is the result of an undefined mathematical operation, such as the division of zero by zero.
`clock`	This special variable contains the current date and time in the form of a six-element row vector containing the year, month, day, hour, minute, and second.
`date`	Contains the current data in a character string format, such as `24-Nov-1998`.
`eps`	This variable name is short for "epsilon." It is the smallest difference between two numbers that can be represented on the computer.
`ans`	A special variable used to store the result of an expression if that result is not explicitly assigned to some other variable.

Arithmetic Operations

- + Add numbers, append strings — 7+5 = 12
- - Subtraction — 7-2 = 5
- .* Element-wise Multiplication — $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} .* \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} = \begin{bmatrix}1 & 4\\9 & 16\end{bmatrix}$
- * Matrix multiplication — $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} * \begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} = \begin{bmatrix}7 & 10\\15 & 22\end{bmatrix}$
- .^ Element-wise power — $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} .\text{^}2 = \begin{bmatrix}1 & 4\\9 & 16\end{bmatrix}$
- ^ Matrix power — $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix} \text{^}2 = \begin{bmatrix}7 & 10\\15 & 22\end{bmatrix}$
- ‘ Transpose — $\begin{bmatrix}1 & 2\\3 & 4\end{bmatrix}' = \begin{bmatrix}1 & 3\\2 & 4\end{bmatrix}$

Trigonometric Functions

Function	Remark
`cos(x)`	
`sin(x)`	
`tan(x)`	
`acos(x)`	$\cos^{-1}(x)$
`asin(x)`	$\sin^{-1}(x)$
`atan(x)`	$-\pi/2 \leq \tan^{-1}(x) \leq \pi/2$
`atan2(y,x)`	$-\pi \leq \tan^{-1}(y, x) \leq \pi$

Function	Remark
`cosh(x)`	$(e^x + e^{-x})/2$
`sinh(x)`	$(e^x - e^{-x})/2$
`tanh(x)`	$(e^x - e^{-x})/(e^x + e^{-x})$
`acosh(x)`	$\cosh^{-1}(x)$
`asinh(x)`	$\sinh^{-1}(x)$
`atanh(x)`	$\tanh^{-1}(x)$

Arithmetic Functions

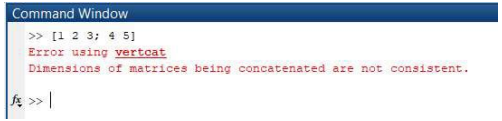
Function	Remark
`exp(x)`	Exponential function
`log(x)`	Natural logarithm
`log10(x)`	Common logarithm
`abs(x)`	Absolute value
`angle(x)`	Phase of a complex number [rad]
`sqrt(x)`	Square root
`real(x)`	Real part

Function	Remark
`imag(x)`	Imaginary part
`conj(x)`	Complex conjugate
`round(x)`	The nearest integer (round-off)
`fix(x)`	The nearest integer toward 0
`floor(x)`	The greatest integer $\leq x$
`ceil(x)`	The smallest integer $\geq x$
`sign(x)`	1(positive)/0/-1(negative)
`mod(y,x)`	Remainder of y/x

Vectors and Matrices

`[3.4]`	This expression creates a 1×1 array (a scalar) containing the value 3.4. The brackets are not required in this case.
`[1.0 2.0 3.0]`	This expression creates a 1×3 array containing the row vector $[1 \quad 2 \quad 3]$.
`[1.0; 2.0; 3.0]`	This expression creates a 3×1 array containing the column vector $\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$.
`[1, 2, 3; 4, 5, 6]`	This expression creates a 2×3 array containing the matrix $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$.
`[1, 2, 3` `4, 5, 6]`	This expression creates a 2×3 array containing the matrix $\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$. The end of the first line terminates the first row.
`[]`	This expression creates an **empty array**, which contains no rows and no columns. (Note that this is not the same as an array containing zeros.)

- The number of elements in every row of an array must be the same, and the number of elements in every column must be the same.



Vectors and Matrices

- Create 1-dimensional arrays using:
  - first:increment:last;
  - linspace(first,last,number);
    - Step = $\frac{last - first}{number - 1}$
  - logspace(first,last,number);

Special Matrices

- I_n = eye(n)
- z_n = zeros(n)
- $z_{m,n}$ = zeros(m,n)
- magic(n)
- ones(n)
- ones(m,n)
- diag(vector)

Useful Functions

- max: maximum and its index
- min: minimum and its index
- sort: sort in the ascending order
- sum: sum up all the elements in a vector (each column in a matrix)
- length: number of the elements in a vector
- size: [rows, cols] = size(matrix);
- numel: number of the elements in a vector
- prod: Product of array elements

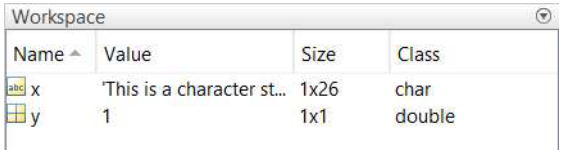
Useful Functions

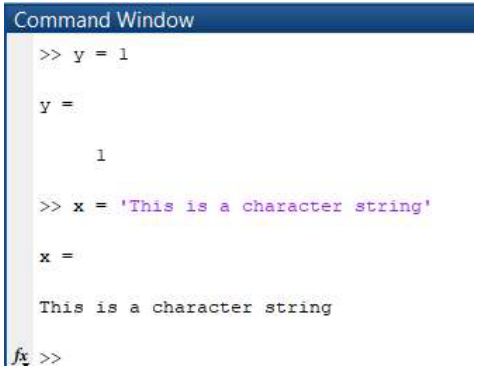
- det(A) — Determinant
- inv(A) — Inversion
- rank(A): — Rank of matrix
  - The rank of a matrix is equal to the number of linearly independent rows (or columns) in it.
- [V, D] = eig (A)
- trace(A): — Sum of diagonal elements
- rand(m, n): — Uniformly distributed random numbers in the interval [0 1]
- randn(m, n): — Normally distributed random numbers

String

- You can represent text in MATLAB using string arrays. Each element of a string array stores a sequence of characters. The sequences can have different lengths without padding, such as "yes" and "no". A string array that has only one element is also called a string scalar.

- num2str
- str2num






```
>> x=[num2str(a) ' is a real number']

x =

2 is a real number
```

Displaying Output Data

- semicolon off
- disp
- fprintf(format, data)

test.m

```
1   % This ia a test program
2 - clc
3 - format short
4 - x = 100.11
5 - y = 1001.1
6 - z = 0.000100112
7 - disp(x)
```

Command Window

```
x =

  100.1100


y =

   1.0011e+03


z =

   1.0011e-04

  100.1100

fx >>
```

test.m

```
1   % This ia a test program
2 - clc
3 - format short
4 - x = 100.11;
5 - y = 1001.1;
6 - z = 0.000100112;
7 - fprintf('The value of x is %f \n',x)
8 - fprintf('The value of x is %e \n',x)
9
10
11
```

Command Window

```
The value of x is 100.110000
The value of x is 1.001100e+02
fx >>
```

Format String	Results
%d	Display value as an integer.
%e	Display value in exponential format.
%f	Display value in floating-point format.
%g	Display value in either floating-point or exponential format, whichever is shorter.
\n	Skip to a new line.

Solving Systems of Linear Equations

$$Ax = B$$

?

1. $inv(A) * B$
2. A^(-1)*B
3. A\B

Thanks!