

# Matlab 1

## Order of operations

1. Parentheses ()
2. Exponentiation ^
3. Negation -
4. Multiplication and division \* /
5. Addition and subtraction + -

## Default Functions

- cos, sin, tan- radian
- Cosd, sind, tand- degrees
- pi- 3.141592...
- Log()-ln
- LOG10()- log

## Var names

- Use descriptive names
- Don't use special characters (other than \_)
- Don't start with number
- Do not overwrite MATLAB built-in function and key words
  - to avoid this put an underscore
- Case sensitive

## Arrays

- collection of data whose elements are arranged in rows and or columns (stored and manipulated)
- scalar- one element array
- vector- one dimensional array (either 1 row or 1 column)
- matrix- two dimensional array (multiple rows and columns)
- Items in rows separated by spaces or commas
- Items in columns separated by semicolons or a new line
- Start:step:end
- linspace(start, end) or linspace(start, end, number of values)

## Selecting array elements

- One element array(row, column)
- Grid selection array(startrow:endrow, startcol:endcol)
- Entire column array(:,col)
- Entire row array(row,:)
- End can be used as an array index

# Matlab 2

## Fprintf

- Can display text (no new line)
- Fprintf("Hello there") —> Hello there(no new line)
- \n new line

- Apostrophe “
- %s- insert string into text
- %0.1f
  - #- is the width
  - .#- is the precision
  - F- floating point

## Concatenate

- link things together in a series or chain
- To concatenate side by side arrays must have the same number of rows
  - [A,B] or [A B]
  - horzcat(A, B)
- To concatenate one on top of another they must have the same number of columns
  - [A;B]
  - Vertcat(A,B)


## Array Manipulation

- A+B adds each element of Array A to Array B
- a\*B multiplies each element of array B by scalar a
- A/b divides each element of A by the scalar b
- A.\*B multiplies each element of Array A to Array B
- A./B divides each element of Array A to Array B
- A.^B takes each element of Array A to Array B power
- Division of a scalar by an array a./B
- Exponential involving scalar a.^B or B.^a

## Conditionals

- < less than
- > greater than
- <= less than or equal to
- >= greater than or equal to
- == equal to
- ~ not
- ~= not equal to
- & and
- | or

## Order of operations

Parentheses	( )	 <p>HIGHEST PRIORITY</p> <p>LOWEST PRIORITY</p>
Transpose and Exponentiation	.', .^, ^	
Negation and Logical Negation (NOT)	~, ~	
Multiplication and Division	.*, ./, *, /	
Addition and Subtraction	+, -	
Relational Operators	<, <=, >, >=, ==, ~=	
Logical AND	&	
Logical OR		
Logical Short-Circuit AND	&&	
Logical Short-Circuit OR		

- & evaluated before |

## Find Function

- determines the linear index location of all non zero elements in array

## If statements

<pre>If condition     Body End</pre>	<pre>If condition     Body Else     Body End</pre>	<pre>If condition     Body Elseif condition     Body Else     Body</pre>
--------------------------------------	--	--

## loops

- Need to initialize variable first

<pre>While condition     Body end</pre>	<pre>for i=f:s:t     body end f=first value s=increment t=the value that is last passed</pre>
---	---

## Array Selection

- A(r,:)- gets row r in the matrix
- A(:,c)- gets column c in the matrix
- A(r,c)- gets item (r,c) in matrix
- A(r1: r2,c1: c2)- gets selection between r1 and r2 and c1 and c2

# Matlab 3

## Functions

Function fun_name(inputs) %body of the function End  Fun_name(invalue)	Function out_name=fun_name(inputs) %body of the function End  Value_out=fun_name(in_valu	Function [out_1,out2]=fun_name(inputs) %body of the function End  [out_1,out2]=fun_name(in_value)
--	---	--

### Python

- Begin with “def”
- Define before call
- Finishes with indenting
- Inputs have ( )
- Outputs separated by ,
- Variable names within function stay within function

### MATLAB

- Begin with “function”
- Define after call at end of program
- Finishes with end statement
  - Indenting does not matter
- Inputs have ( )
- Outputs have [ ]
- Variable names within function stay within function

Matlab Script	Matlab Function
File extension is .m ( <b>filename.m</b> )	File extension is .m ( <b>filename.m</b> )
First line of executable code can be anything!	First line of executable code <b>must</b> be the <b>function definition line!</b> <ol style="list-style-type: none"> <li>1. Keyword <b>function</b></li> <li>2. Outputs in square brackets [ ]</li> <li>3. Function name (matching filename!)</li> <li>4. Inputs in parentheses ( )</li> </ol>
Runs as if the code was typed at the Matlab prompt: <b>All variables</b> created within the script are available in the Matlab <b>workspace</b> after running	Run as if it was a Matlab built-in function: <b>Only input variables</b> can be used <b>inside</b> the function and <b>only output variables</b> create by function are available in the <b>workspace</b> after running
The green arrow button (F5) to run the code from the Editor Window will work without issues	The green arrow button (F5) to run the code from the Editor Window <b>will not work</b> unless you call it with required input

File Extension	File Format	Description	MATLAB Importing Function
.csv	Comma separated values	<ul style="list-style-type: none"> <li>Comma separates each entry (i.e., column) in a row</li> <li>Carriage return separates each row</li> </ul>	csvread
.txt	Delimited values	<ul style="list-style-type: none"> <li>Delimiter between entries can be space, tab, comma, or semicolon</li> <li>Carriage return separates each row</li> </ul>	load

- Change folder to the location of the file
- Txt files
  - Load("name.txt") or load('name.txt')
  - Can only load numeric data
  - % at the beginning of line to hide unwanted data from matlab
  - % headers and any text
- Csv files
  - csvread("name.txt") or tableread('name.txt')
  - Can specify row and data of the numeric data
  - A1 is 0,0
  - csvread("name.txt",r,c)
    - start row and column at zero because matlab is stupid

Ensure that MATLAB's current folder contains the data file

Import numeric data using the `load` and `csvread` commands

Import .txt files using the `load` command

Manage any text within the file using % or by deleting the text

Place the filename, with the extension, within matching quotes

Import .csv files the `csvread` command

Ignore text rows by assigning the starting point of the numeric data, remembering that MATLAB counts from 0

Place the filename, with the extension, within matching quotes

linear indexing goes down a column then dominates down the next one etc.

Plot(x,y,LineSpec)

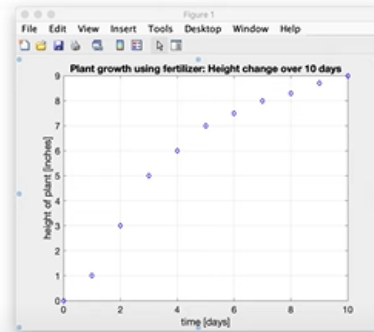
Technical presentation includes:

- **Title:**

- Descriptive of context
- References independent and dependent variables
- **MATLAB syntax:** `title('Title of plot')`

- **Label axes:**

- Clearly describes variables
- Units included
- **MATLAB syntax:** `xlabel('x_variable_description [units]')`  
`ylabel('y_variable_description [units]')`



**Grid on**  
MORE VIDEOS

- **MATLAB syntax:** `grid on` (to turn grid back off, use `grid off`)

Identify the independent and dependent variables

Use data markers with no lines when plotting measured data

Use lines with no markers when plotting theoretical or modeled data

Create plot titles that clearly describe the context of the data and mention of the independent and dependent variables

**hold ON** holds the current plot and all axis properties, including the current color and linestyle, so that subsequent graphing commands add to the existing graph without resetting the color and linestyle. **hold OFF** returns to the default mode whereby PLOT commands erase the previous plots and reset all axis properties before drawing new plots. **hold**, by itself, toggles the hold state. **hold** does not affect axis autoranging properties.

**hold ALL** is the same as **hold ON**. This syntax will be removed in a future release. Use **hold ON** instead.

```
legend("label_1", 'label_2', 'label_3', ...)
```

label for  
1<sup>st</sup> data  
set

label for  
2<sup>nd</sup> data  
set

label for  
3<sup>rd</sup> data  
set

**MATLAB command:** figure(H)

Figure handle

## Matlab 5

### Different Coefficient Solutions

**PURDUE**  
UNIVERSITY

Engineering Education

The equation on the right are an equivalent solution to the two least squares differential equations

Equations described earlier	Simplified equations
$a \sum_{i=1}^n x_i^2 + b \sum_{i=1}^n x_i = \sum_{i=1}^n x_i y_i$	$a = \frac{\bar{x} \bar{y} - \overline{xy}}{(\bar{x})^2 - \overline{x^2}}$
$a \sum_{i=1}^n x_i + bn = \sum_{i=1}^n y_i$	$b = \bar{y} - a\bar{x}$

# How good is my model line?

Hooke's Law:  $F = 20x$

Conventional models of estimating "goodness of fit" for linear regression

Abbreviation	Name	Description	Equation
SSE	Sum of Squared Errors	A measure of how well we can predict the dependent variable	$SSE = \sum_{i=1}^n [y_i - f(x_i)]^2$
SST	Sum of Squared Deviations (Total)	A measure of how much variability is present in the original dependent variable	$SST = \sum_{i=1}^n [y_i - \bar{y}]^2$
$r^2$ ("r-squared")	Coefficient of Determination	<b>A measure of the percent of the variability in the data can be explained by the model</b>	$r^2 = 1 - \frac{SSE}{SST}$