ENGR 1100 Exam #1 Review

- How to save plots and insert in a word document?
- Be able to create simple MATLAB function

1. [Trigonometry Functions]

- Understand the right triangle definition of basic trig/inverse trig functions and their applications

Function	**Abbreviation**
sine	sin
cosine	cos
tangent	tan
cosecant	csc
secant	sec
cotangent	cot

- Evaluate the trig/inverse trig functions in MATLAB. Remember that MATLAB regular trig functions only take radian as inputs! (**sind()** take degree as inputs)
- Understand and apply the fundamental identities
  - Reciprocal identity
  - Quotient Identity

Reciprocal Identities

$$\sin\theta = \frac{1}{\csc\theta} \qquad \csc\theta = \frac{1}{\sin\theta}$$
$$\cos\theta = \frac{1}{\sec\theta} \qquad \sec\theta = \frac{1}{\cos\theta}$$
$$\tan\theta = \frac{1}{\cot\theta} \qquad \cot\theta = \frac{1}{\tan\theta}$$

Quotient Identities

$$\tan\theta = \frac{\sin\theta}{\cos\theta} \qquad \cot\theta = \frac{\cos\theta}{\sin\theta}$$

  - Pythagorean Identity

$$\sin^2\theta + \cos^2\theta = 1 \qquad 1 + \tan^2\theta = \sec^2\theta \qquad 1 + \cot^2\theta = \csc^2\theta$$

  - Law of cosine

Example: find the values for following trig functions

a. csc 1.5 (radian mode)
b. sin 72.8° (degree mode)
c. $cos^{-1}(0.6)$

2. [Solve Linear Equations]

For linear equation $Ax = b$,

- Be able to solve it using Gaussian elimination method by hand.
- Understand Cramer's rule and the conditions that we can apply it (**optional**)
- Understand the matrix inversion method
- Use MATLAB to solve the linear equations
  - $x = A^{-1}b$
  - linsolve(A, b)

3. **[Array]** The following matrix is defined in MATLAB

S=

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24

By hand write what will be displayed if the following commands are executed by MATLAB. Check your answer by executing the commands with MATLAB

a) A=S(2, [2,4])
b) B=S(3, [3:5])
c) C=S([2:4], [4:6])
d) D=S(:, [1:3])

4. [Mathematical Operations]

- Mathematical operations (**^, ***) follow the rules of linear algebra for arrays.
- Element-by-element operations (**.^, .*** and **./**) can be carried out with arrays of the same size

A=

2	6	3
5	8	4
1	2	4

B=

1	4	10
3	2	7
5	6	2

a) Compare A^2 and A.^2, understand how these two are different
b) Compare A*B and A.*B
c) Compute A./B

5. [Managing Data]

- Input Command: **input()**

  myVar=input('Please input the data ')
  myStr=input('Please input the string ', 's')

- Output Command:
  - **disp()**

  disp(myVar)
  disp('Here is the output')

  - **fprintf()** can be use to display a mix of text and numerical data

  fprintf('The new result is %5.2f better than the original one \n', myVar)

Remark: The **fprintf()** command is vectorized. This means that when a variable is a vector or a matrix is included in the command, the command repeats itself until all the elements are displayed. If the variable is a matrix, the data is used column by column.

```
X=1:5;
y=sqrt(x);
T=[x; y]
fprintf('If the number is: %i, its square root is: %f \n', T)
```

- Use fprintf() to save output to a file

  Step1: open the file that you want to write to

  ```
  fid=fopen('fileName.txt', 'permission')
  ```

  Step2: use fprintf() to write outputs to the file

  ```
  fprintf(fid, 'text information', variables)
  ```

  Step3: close the file once writing data is complete

  ```
  fclose(fid)
  ```

- The **save** and **load** command

  These two commands are useful for saving and retrieving data for use in MATLAB. The save command can be used for saving the variables that are currently in the workspace, and the load command is used for retrieving variables that have been previously saved, to the workspace.

  ```
  Save('fileName', list of variables)
  load('fileName', lise of variables)
  ```

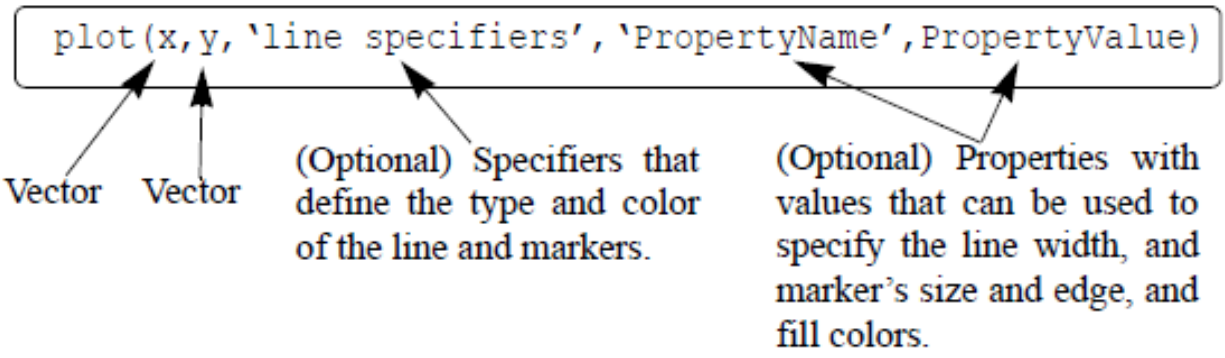
- Importing & Exporting data with Excel

  ```
  varibleNames=xlsread('fileName','sheetName','range')
  xlswrite('fileName', variableNames)
  ```

6. [Plots and Graphs]

In addition to plotting, you should be able to add labels, title and legends to the plot.

- **2D plot:** plot() command is used to make basic 2D plots



7. [Conditional Statements]

- **Relational operator**

Relational operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
~=	Not Equal to

**Result of comparing with a relational operator is always "true" or "false"

- If "true", MATLAB gives the comparison a value of one (1)
- If "false", MATLAB gives the comparison a value of zero (0)

- **Logical operator**

MATLAB has three logical operators: &, |, ~

- a&b does the logical AND operation on a and b,
-
- a|b does the logical OR operation on a or b,
- ~a does the logical NOT operation on a,

**Arguments to all logical operators are numbers

- Zero is "false"
- Any non-zero number is "true"

**Result (output) of logical operator is a logical one (true) or zero (false)

Remarks: Both the relational operator and logical operator can be applied on arrays (element-wise comparison).

Logical	Operator	MATLAB Funciton

A and B	A & B	and(A, B)
A or B	A | B	or(A, B)
Not A	~A	not(A)

Other MATLAB Boolean functions: any(A), all(A), find(A>x), etc.

Conditional statement is a command that allows MATLAB to decide whether or not to execute some code that follows the statement

- Conditional statements almost always part of scripts or functions
- They have three general forms
  - if-end
  - if-else-end
  - if-elseif-else-end

- Switch-case slightly different because choose code to execute based on value of scalar or string, not just true/false

  Example:

```
switch name
case 'Bobby'
    talk for a long time
case 'Susan'
    talk for a little bit and then set a time to meet
case 'Hubert'
    talk until you get the answer to the hard problem
case 'Mom'
    say you're busy and can't talk
otherwise
    have your room-mate say you'll call back later
end
```

8. [Loops]

A *loop* is another method of flow control. A loop executes one set of commands repeatedly.

MATLAB has TWO ways to control number of times loop executes commands

a) loop executes commands a specified number of times—for loops
b) loop executes commands as long as a specified expression is true—while loops

For loops: the index k can also be assigned specific value (typed in as a vector), for example: for k = [7 9 –1 3 3 5]

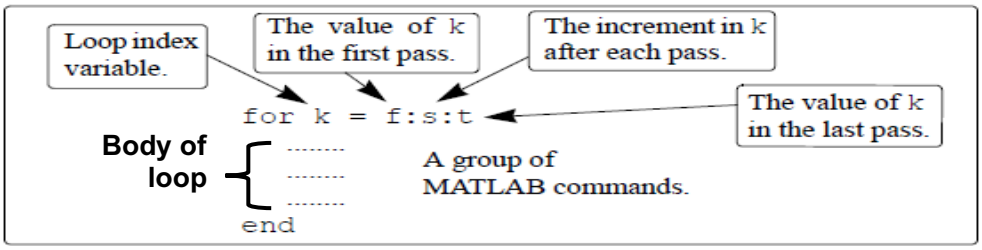


Figure 6-5: The structure of a `for-end` loop.

While loop: Loop evaluates conditional-expression

- If conditional-expression is true, executes code in body, then goes back to check the condition;
- If conditional-expression is false, skips code in body and goes to code after end-statement

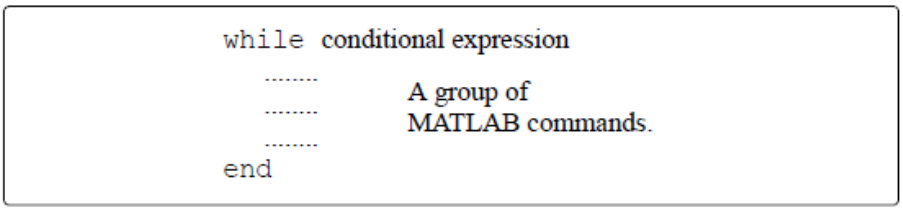


Figure 6-6: The structure of a `while-end` loop.

Remarks: Be careful not to create infinite loops!

9. Other useful MATLAB command: clear/clc/whos/diary