THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

ENG 101

Matlab Variables

Dr. Emil Jovanov
Electrical and Computer Engineering
University of Alabama in Huntsville
emil.jovanov@uah.edu
http://www.ece.uah.edu/~jovanov

Variables

THE UNIVERSITY OF ALABAMA IN HUNTSVILLE

A *variable* is a name that is assigned a numerical value

- Once assigned, can use variable in expressions, functions, and MATLAB statements and commands
- Can *read* the variable (get its value)
- Can *write to* the variable (set its value)

= (equals sign) is MATLAB's *assignment operator*. It evaluates the expression on its right side and stores the resulting value in the variable on its left side

```
>> a = 3
a =
     3
```

Create the variable called "a" and store the value 3 in it

MATLAB acknowledges that it has created "a" and set it to 3

EXAMPLE

```
>> a = 3
a =
     3
>> b = 10*a + 5
b =
     35
```

Make a variable and store a number in it

Make a variable and store the value of an expression made up of a variable, numbers, and addition and multiplication

Think of = as meaning "assign to" or "store in" but not meaning "equals"!

Why?

x = *x* + *6* has no meaning in math because it implies that *0* = *6*

x = x + 6 is perfectly fine in MATLAB because it means "take whatever is in x, add 6 to that and store the result back into x"

EXAMPLE

```
>> x = 3;
>> x = x + 6
x =
     9
>> x = 2 * x
x =
    18
```

; at end prevents MATLAB from displaying value of x

takes what's in x (3), adds 6 to it to get 9, then stores 9 back into x

now x's value is 9

takes what's in x (9), multiplies it by 2 to get 18, then stores 18 back into x

now x's value is 18

A variable must have a value before you use it in an expression

```
>> x = 3;
>> x+2
ans =
     5
>> x + y % assume y undefined
??? Undefined function or variable
'y'
```


To find out the value of a variable, just type it and press ENTER

```
>> x = 3;
>> y = 10 * x;
>> z = y ^ 2;
>> y
y =
     30
>> z
z =
     900
```

Can do multiple assignments on one line by separating with a comma or semicolon. If semicolon, no display for that assignment

```
>> a=12, B=4; C=(a-B)+40-a/B*10
a =
     12
C =
     18
```


To change the value of a variable, just assign it the new value

```
>> ABB=72;
>> ABB=9;
>> ABB
ABB =
      9
```

You must define a variable (give it a value) before you can use it in an argument of a function

```
>> sqrt( x ) % assume x undefined
??? Undefined function or variable 'x'
>> x = 144;
>> sqrt( x )
x =
     12
```


A variable name

- Must begin with a letter
- Can be up to 63 characters long
- Can contain letters, digits, and underscores (_)
- Can't contain punctuation, e.g., period, comma, semicolon

Avoid using the name of a built-in function as the name of a variable, e.g., don't call a variable `exp` or `sqrt`

MATLAB is *case-sensitive*, and does not consider an upper-case letter in a variable name to be the same as its lower-case counterpart,
e.g., `MTV`, `MTv`, `mTV`, and `mtv` are four different variable names

A variable name cannot contain a space.
Two common alternatives:

1. Use an underscore in place of a space, e.g., `speed_of_light`
2. Capitalize the first letter of every other word, e.g., `speedOfLight` (This is known as *camel case*!)

Keywords

A *keyword* is a word that has special meaning to MATLAB

- There are 20 keywords (see book)
- Appear in blue when typed in the Editor Window
- Can't be used as variable names

MATLAB has pre-defined variables for some common quantities

`pi` the number π

`eps` the smallest difference between any two numbers in MATLAB

`inf` or `Inf` infinity

`i` $\sqrt{-1}$

`j` $\sqrt{-1}$ (same as `i`) but commonly used instead of `i` in electrical engineering

More pre-defined variables

`ans` the value of the last expression that was not assigned to a variable

`NaN` or `nan` not-a-number. Used to express mathematically undefined values, such as `0/0`

Some commands for managing variables

Command	**Outcome**
`clear`	Removes all variables from memory
`clear x y z`	Removes only variables `x`, `y`, and `z` from memory
`who`	Displays a list of the variables currently in memory
`whos`	Displays a list of the variables currently in memory and their size, together with information about their bytes and class (see Section 4.1)