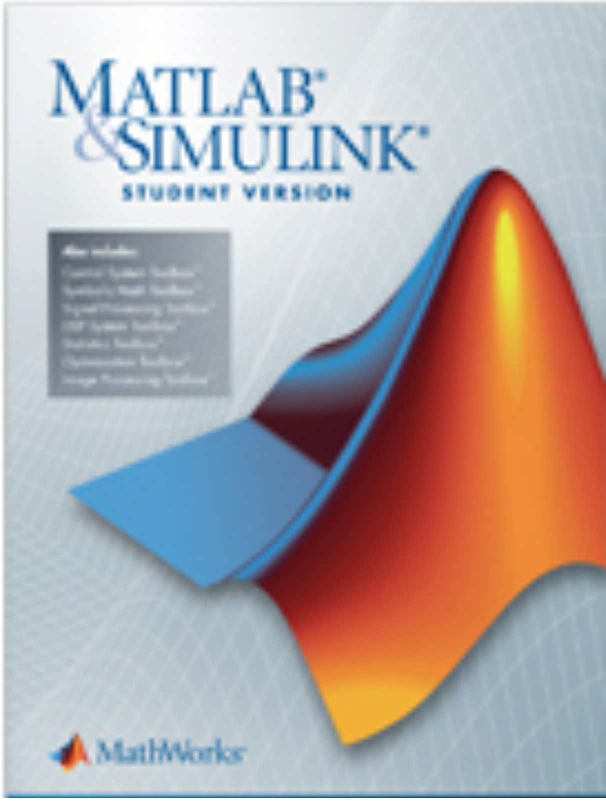
MATLAB in ECE 201-202



MATLAB is a powerful and widely used application for mathematics and engineering. It is commonly used by electrical and computer engineers and it is highly likely that electrical and computer engineering students will use MATLAB in later courses and as working engineers.

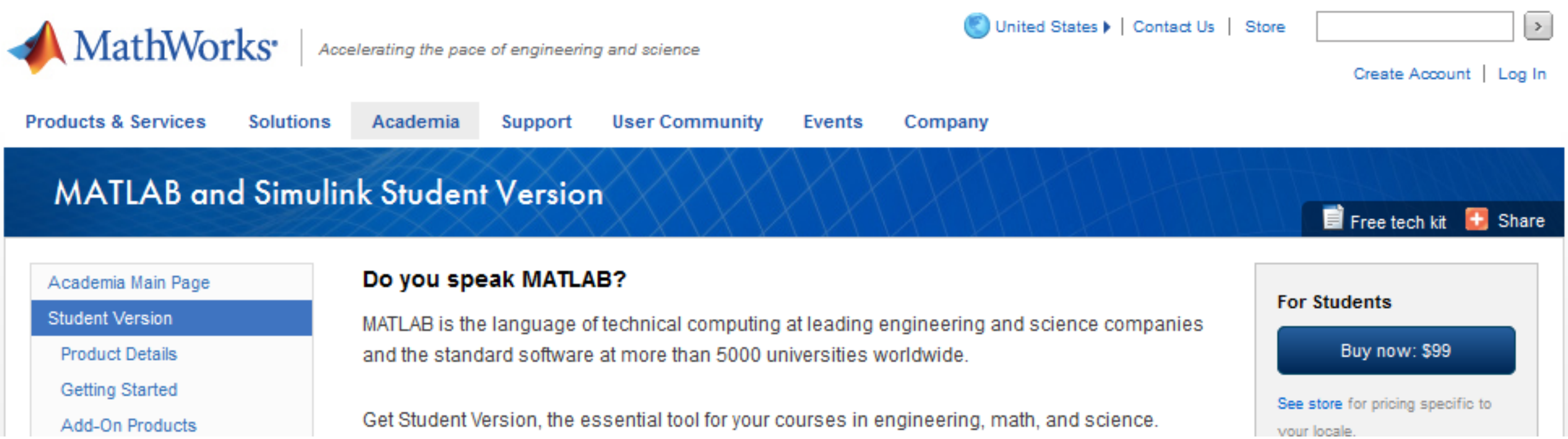
MATLAB will be used in ECE 201-202 in many areas, including:

- Solving systems of equations
- Performing symbolic calculations
- Writing interactive programs
- Determining graphical results
- Analyzing AC circuits using complex numbers
- Analyzing circuits and systems using with Laplace transforms

MATLAB Student Edition

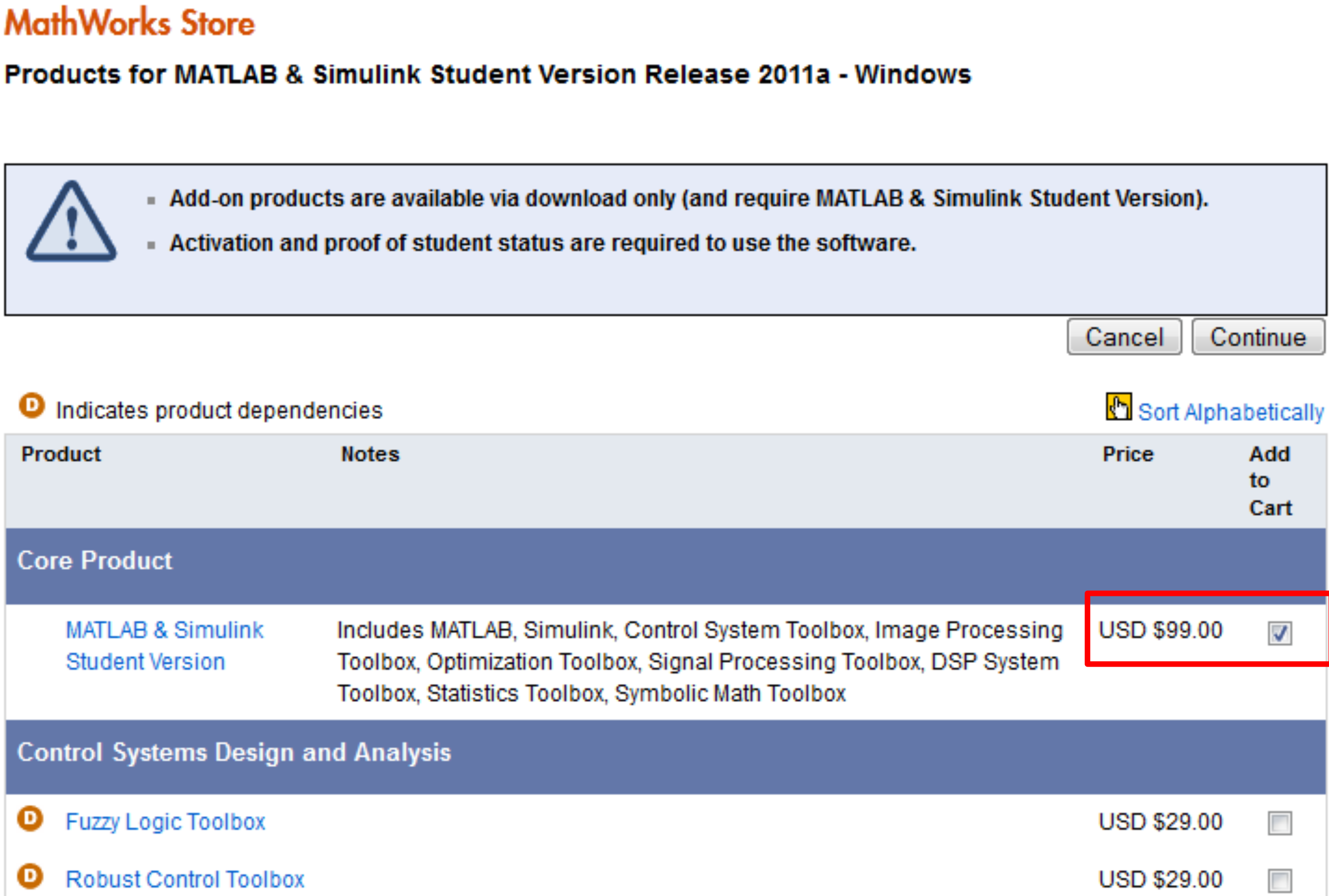
We will use MATLAB routinely in this course. Although MATLAB is available on campus, it would be advantageous to purchase a copy of the student edition of MATLAB. The student version is full-featured and will be useful in later courses as well. To purchase the student edition, do a quick online search for "MATLAB student edition" or go to:

http://www.mathworks.com/academia/student_version/



MATLAB Student Edition

If you purchase the student edition of MATLAB you will need to set up a new account using your TCC email address. Note that there are additional add-on products available with MATLAB, but we will only need the standard student version as indicated below.



MathWorks Store

Products for MATLAB & Simulink Student Version Release 2011a - Windows

- Add-on products are available via download only (and require MATLAB & Simulink Student Version).
- Activation and proof of student status are required to use the software.

Cancel Continue

Indicates product dependencies

Sort Alphabetically

Product	Notes	Price	Add to Cart
Core Product			
MATLAB & Simulink Student Version	Includes MATLAB, Simulink, Control System Toolbox, Image Processing Toolbox, Optimization Toolbox, Signal Processing Toolbox, DSP System Toolbox, Statistics Toolbox, Symbolic Math Toolbox	USD $99.00	☑
Control Systems Design and Analysis			
Fuzzy Logic Toolbox		USD $29.00	☐
Robust Control Toolbox		USD $29.00	☐

Check this box only

MATLAB Environment

When MATLAB is launched, several windows will appear. An important part of learning MATLAB is learning how to use each of these windows.

As we will see later, MATLAB uses a number of other windows as well, including:

- Editor – for creating and editing MATLAB programs
- Array editor
- Help window
- Properties windows

Vectors and scalars in MATLAB

MATLAB is designed to work with vector quantities or matrices, where the elements of the vector or matrix are placed inside brackets [ ] with the elements in each row separated by spaces and the rows separated by semicolons. However, MATLAB can also be used to work with variables defined by a single value called scalars.

Example:

Type of quantity	Mathematical notation:	MATLAB notation:
Vector	$A = \begin{bmatrix} 2 & 3 & 4 \\ 7 & 9 & -1 \end{bmatrix}$	A = [2 3 4;7 9 –1]
Scalar	x = 2	x = 2

Variable names in MATLAB

- are case-sensitive
- can contain up to 63 characters (any characters beyond the 63rd are ignored)
- must start with a letter, followed by any comb. of letters, numbers, and underscores
- cannot contain spaces

Exercise: Indicate whether each variable used below is valid or invalid.

Inductive Reactance = 12.5;	Valid/Invalid
2nd _derivative = 10;	Valid/Invalid
X1X2X3 = 54;	Valid/Invalid
Energy-Delivered = 1.25E6;	Valid/Invalid
Fifteen = 16;	Valid/Invalid
Problem_5.2_Answer = 7;	Valid/Invalid
Capacitor_Voltage = 125;	Valid/Invalid

Expressions in MATLAB

MATLAB, like most programming languages, requires that expressions contain a single variable to the left of the assignment operator (equals sign).

Example:

$$\underbrace{X}_{\text{Single variable}} \; \underbrace{=}_{\text{Assignment operator}} \; \underbrace{2*A + 3*B - 15}_{\text{Expression involving variables, functions, numbers, etc}}$$

Additionally, MATLAB performs the calculations to the right of the assignment operator (=) using existing values of variables and then assigns the result to the variable to the left of the equals sign.

So the expression **$X = X + 2$** should be interpreted as **$X_{new} = X_{old} + 2$**

Note the result of the following sequence of instructions in MATLAB (try it!):

Command Window

```
>> x=2;
>> x=x+3

x =

     5
```

Note: a semicolon (;) after a command suppresses the value from being printed

MATLAB's Basic Scalar Arithmetic Operators

Operation	Symbol	Example
Addition	+	X = A + B
Subtraction	-	X = A - B
Multiplication	*	X = A*B
Division	/	X = A/B
Exponentiation	^	X = A^2 + B^2

MATLAB's Order of Operations (for scalars)

Operators are executed left to right using the following precedence.

Operation	Symbol	Precedence
Parentheses	( )	Highest
Exponentiation	^	
Negation	-	
Multiplication or division	* or /	
Addition or subtraction	+ or -	Lowest

Exercise:

The expression $F_1 = \dfrac{B}{A \cdot C}$ can be evaluated using which two of the three expressions below?

F1 = B/A * C

F1 = B/(A * C)

F1 = B/A/C

Function	Description
pi	returns π to 15 significant digits
cos(x)	returns cosine of x, with x in radians
sin(x)	returns sine of x, with x in radians
tan(x)	returns tangent of x, with x in radians
acos(x)	returns arccosine of x, in radians
asin(x)	returns arcsine of x, in radians
atan(x)	returns arctangent of x, in radians
exp(x)	returns the value of e^x
sqrt(x)	returns the square root of x
factorial(x)	returns factorial of x
log(x)	returns ln(x)
log10(x)	returns $\log_{10}(x)$
abs(x)	returns absolute value of x
sinh(x)	returns the hyperbolic sine of x
round(x)	rounds x off toward nearest integer
ceil(x)	rounds x toward positive infinity
floor(x)	rounds x toward negative infinity

MATLAB built-in functions

MATLAB has hundreds of built-in functions. A few of the functions commonly used with scalars are shown in the table.

More functions

Open MATLAB and select ***Help – Function Browser*** and check out the vast number of functions available in MATLAB. What do the following functions do?

sind(A)

hypot (a,b)

cot(A)

mod(N,M)

atan2(y,x)

MATLAB Command	Description	Example X = 2/3 displayed as:
format short	express using 4 digits after decimal point (default)	0.6667
format long	express using 14 digits after decimal point (d.p.)	0.66666666666667
format rat	express using a ratio of two integers	2/3
format bank	express using 2 digits after d.p.	0.67
format short e	express in scientific notation w/ 4 digits after d.p.	6.6667e-001
format long e	express in scientific notation w/ 14 digits after d.p.	6.666666666666667e-001
format compact	suppresses blank lines in the output **(Nice option!)**	(see example below)

MATLAB Format Commands

MATLAB offers a few commands for controlling the format of the output. Additional useful commands are provided on the next page.

```
>> y=x^3

y =

   125

>> format compact
>> y=x^3
y =
   125
>>
```

MATLAB Command	**Description**
clc	clear the Command Window
clear	clear values of all variables
disp(x)	display value of a single variable or expression (see examples below)
fprintf(a,b,c,…)	Display values of variables and text with a desired number of digits (see examples on the next page). Formatting can include: '\n' – linefeed (carriage return) '\t' – tab '%0.4f' – use fixed format w/4 digits after the decimal point (dp) '%9.4f – use fixed format w/4 digits after the dp and 9 total spaces '%0.4e' – use scientific notation w/4 digits after dp '%0.4g' – use 4 significant digits (in fixed or scientific)

Example using fprintf( )

```
IA = 2.62;      % Current in mA
fprintf('I = %0.2f  mA\n', IA)
```

Text

Text

Variable to be printed with the formatter

% - start of formatter
0 - use as many spaces as needed
2 - use 2 digits after dp
f - use a fixed (f) format (i.e., not scientific notation)

\n = "newline" character (go to a new line)

Output:

```
I = 2.62  mA
```

Additional examples using fprintf( )

```
IA = 2.62;  % Current in mA
IB = 3.74;  % Current in mA
fprintf('I = %12.2f  mA\n', IA);       % Use 12 total spaces
fprintf('I = %0.2e  mA\n', IA);        % Use scientific notation
fprintf('I = %0.2f  mA\n', IA, IB);    % Two variables, so repeat the format
fprintf('IA = %0.2f  mA,  IB = %0.2f  mA\n', IA,IB); % Two variables on one line
fprintf('IA = %0.2f  mA\nIB = %0.2f  mA\n', IA,IB); % Two variables on two lines
```

Output:

12 spaces

```
I =         2.62  mA
I = 2.62e+00  mA
I = 2.62  mA
I = 3.74  mA
IA = 2.62  mA,  IB = 3.74  mA
IA = 2.62  mA
IB = 3.74  mA
```

MATLAB Script Files (.m files)

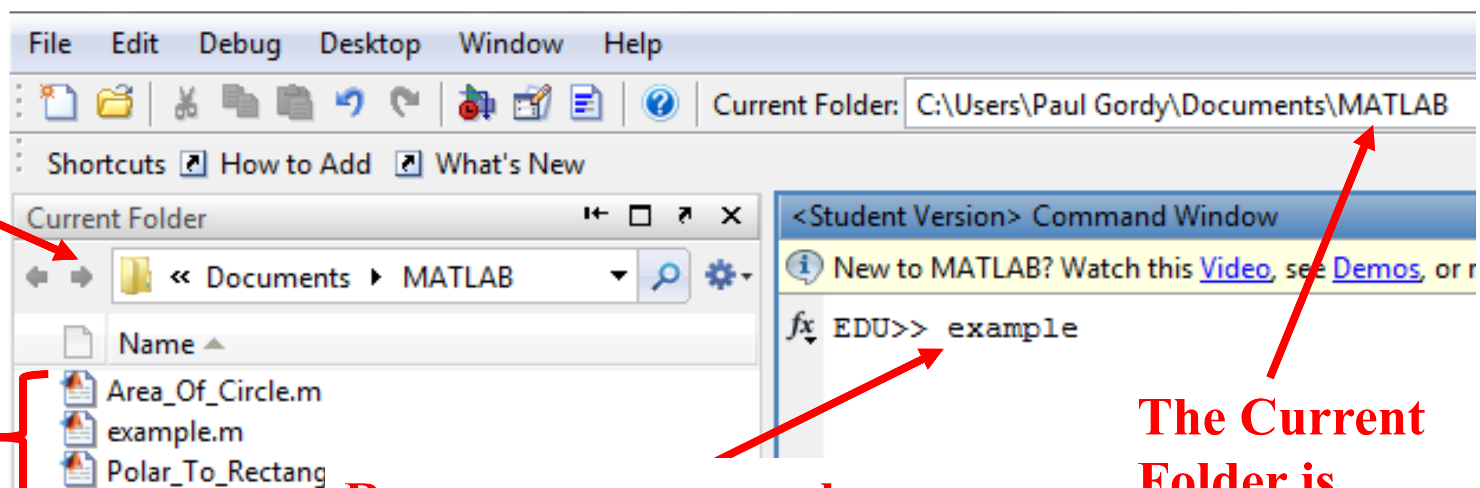
We have seen how to execute MATLAB commands in the Command Window. We will now see how to write a program in MATLAB. The program (called a script file or .m file) is a file containing a series of statements much like those used previously in the Command Window. A program has the advantage of being reusable. You can easily try a sequence of commands and then modify them if the output isn't correct. The program can be executed by typing its name in the Command Window.

The Current Directory

When the name of a program is entered into the Command Window, MATLAB looks for the program in the Current Directory and executes it if it is located there.

Change current folder here

MATLAB files in current folder

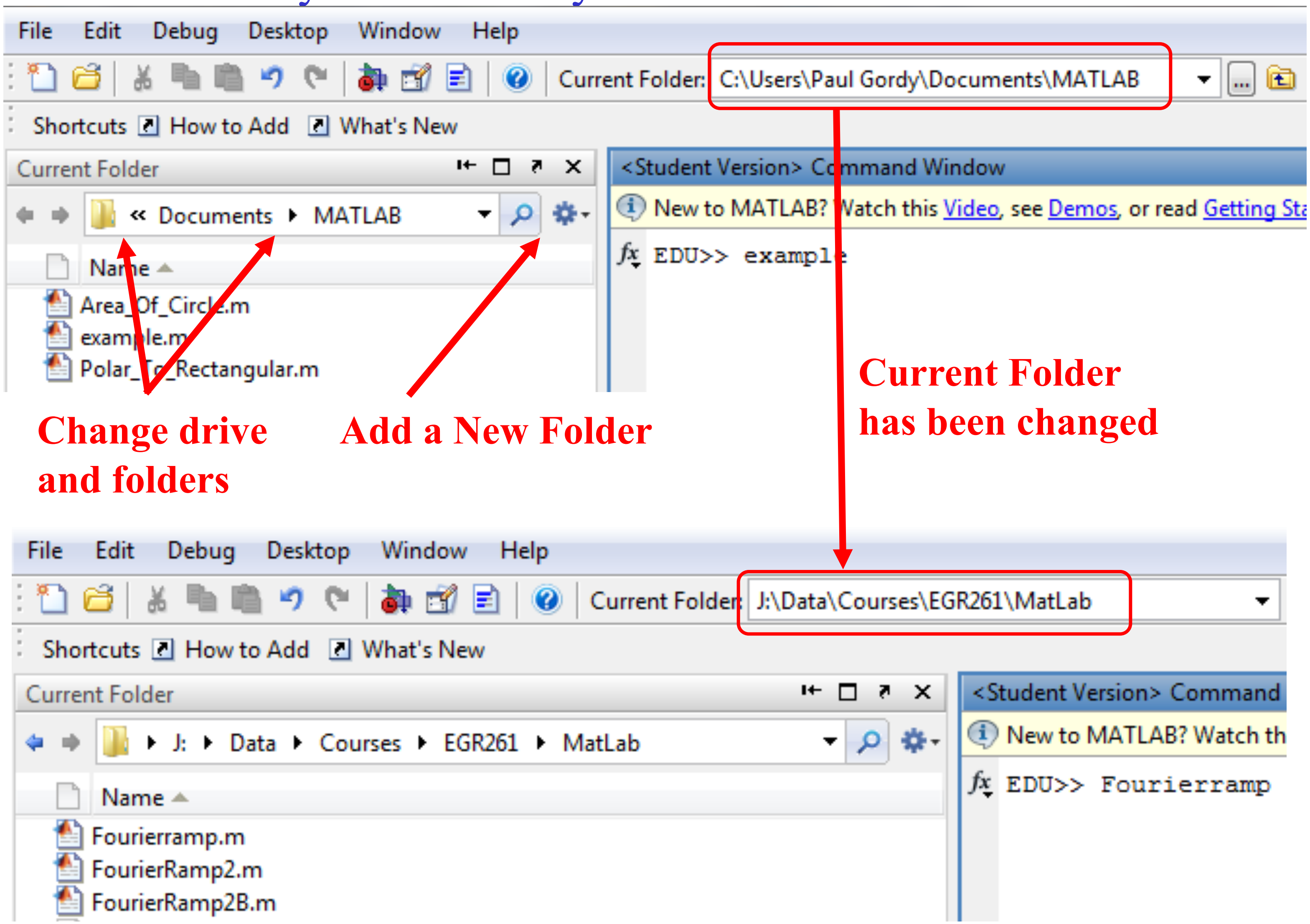


Run program example.m in the current folder

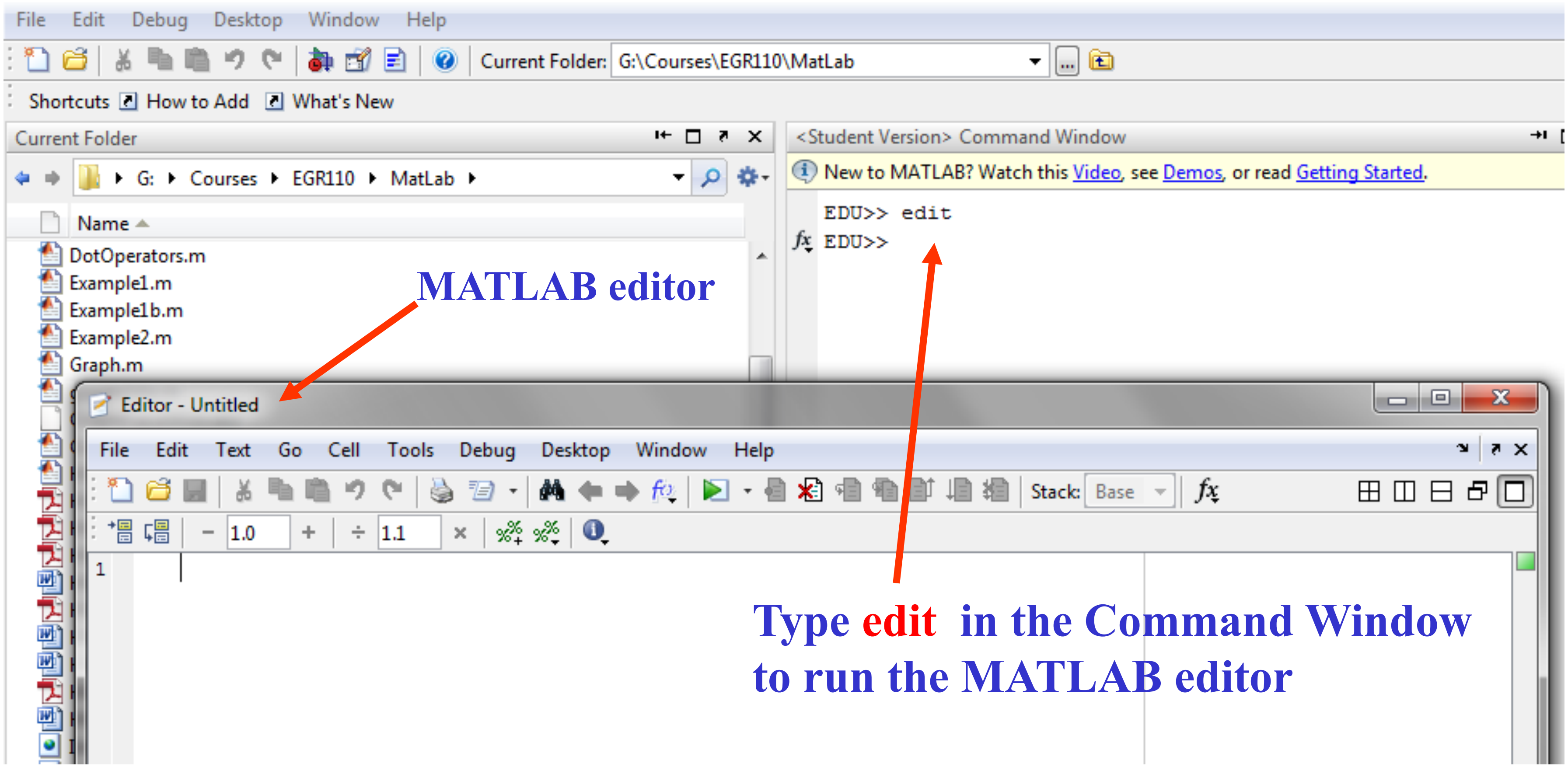
The Current Folder is shown here.

Changing the Current Folder

It is recommended that you change the current folder to the folder on your personal drive where you will save your files.



Creating a script (.m) file



Notes:

- Type edit to open the editor
- Type edit MyFile.m to create a new file named MyFile.m (if it does not exits)
- Type edit MyFile.m to open an existing file named MyFile.m

Using MATLAB's INPUT function

The following MATLAB function is used to prompt the user to enter inputs:

input ('message') % returns a value entered from the keyboard.

Example:

R = input('Enter the resistance in ohms');

Example:

- Write a MATLAB program to calculate the power absorbed by a resistor.
- Prompt the user to enter the resistance and the voltage.
- Display the input values and the output (with units).

Solution – next page

Sample MATLAB program


```
% EGR 271 - Circuit Theory I
% Filename:  Resistor_Power.m
% Description:  Prompt the user to enter a resistance and the voltage
% across the resistor and then calculate the power absorbed by the
% resistor.
clc
format compact
R = input('Please enter the resistance (in ohms): ');
V = input('Please enter the voltage (in volts): ');
P = V^2/R;
fprintf('Resistance:  %0.1f ohms\n', R);
fprintf('Voltage: %0.1f',V); disp(' volts'); % alternate way to show units
fprintf('Power:  %0.1f watts\n', P);
```

Run the program by selecting **Run** in the editor or by entering ***Resistor_Power*** in the Command Window.

Sample Output:

```
Please enter the resistance (in ohms): 10
Please enter the voltage (in volts): 25
Resistance:  10.0 ohms
Voltage: 25.0 volts
Power:  62.5 watts
```

Symbolic Expressions in MATLAB

MATLAB is a powerful tool for working with symbolic expressions.

Useful operations with symbolic expressions include:

- Solving for variables as functions of other variables (algebraic manipulation)
- Solving simultaneous equations
- Finding derivatives and integrals
- Solving differential equations (useful in ECE 202)
- Finding Laplace and inverse Laplace transforms (useful in ECE 202)
- Much more!

```
% Filename:  symbolic1.m
% Working with symbolic expressions in MATLAB
syms x y;  % Define symbolic variables (use spaces, not commas)
F = 2*x^2 + 3*x;     % Define a symbolic relationship.  Note that F did
                     % not need to be listed in syms
dF = diff(F)         % Find the derivative of F
```

Output:

```
>> symbolic1
dF =
4*x + 3
```

Example:

Use MATLAB to find a symbolic solution to the quadratic equation.
Use the symbolic function **solve( )**.

```
EDU>> symbolic_quadratic
ans =
 -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
 -(b - (b^2 - 4*a*c)^(1/2))/(2*a)
EDU>>
```

Editor

File Edit Text Go Cell Tools Debug Desktop Window Help

J:\Data\Courses\EGR267\Matlab\symbolic_quadratic.m

```
% Example:  Solving a quadratic equation symbolicly
% Filename:  symbolic_quadratic.m
syms a b c x;  % define symbolic variabls
solve('a*x^2 + b*x + c = 0', x)  % solve for x in terms of the other variables
```

Example: - Some modifications to the previous example:

```
% Filename: symbolic2.m
syms a b c x;
F = a*x^2 + b*x + c;      % No quotes needed
solve(F,x)                % Same as solve (F=0,x);
```

Output:

```
>> symbolic2
ans =
 -(b + (b^2 - 4*a*c)^(1/2))/(2*a)
 -(b - (b^2 - 4*a*c)^(1/2))/(2*a)
```

```
% Filename: symbolic3.m
syms a b c x;
F = a*x^2 + b*x + c;
pretty(solve(F,x))        % Use pretty() for nicer format
```

Output:

```
>> symbolic3

  +-                          -+
  |          2          1/2    |
  |     b + (b  - 4 a c)       |
  |   - ------------------     |
  |            2 a             |
  |                            |
  |          2          1/2    |
  |     b - (b  - 4 a c)       |
  |   - ------------------     |
  |            2 a             |
  +-                          -+
```

Symbolic versus numeric result

- Note that there is often no difference between solving equations that have a numeric result and equations that have a symbolic result. It is simply a matter of the number of equations versus the number of unknowns.
- Example: 2 equations, 2 unknowns → numeric solution
- Example: 2 equations, 3 unknowns → symbolic solution

```
% Filename:  symbolic_unknowns.m
clear;  % clear all variables
syms x y z;
[x] = solve('2*x - 40 = 0',x)     % 1 equation, 1 unknown
[y] = solve('2*y + 3*z = 10',y)   % 1 equations, 2 unknowns
```

```
EDU>> symbolic_unknowns
x =
20
y =
5 - (3*z)/2
```

Example – solving simultaneous equations:

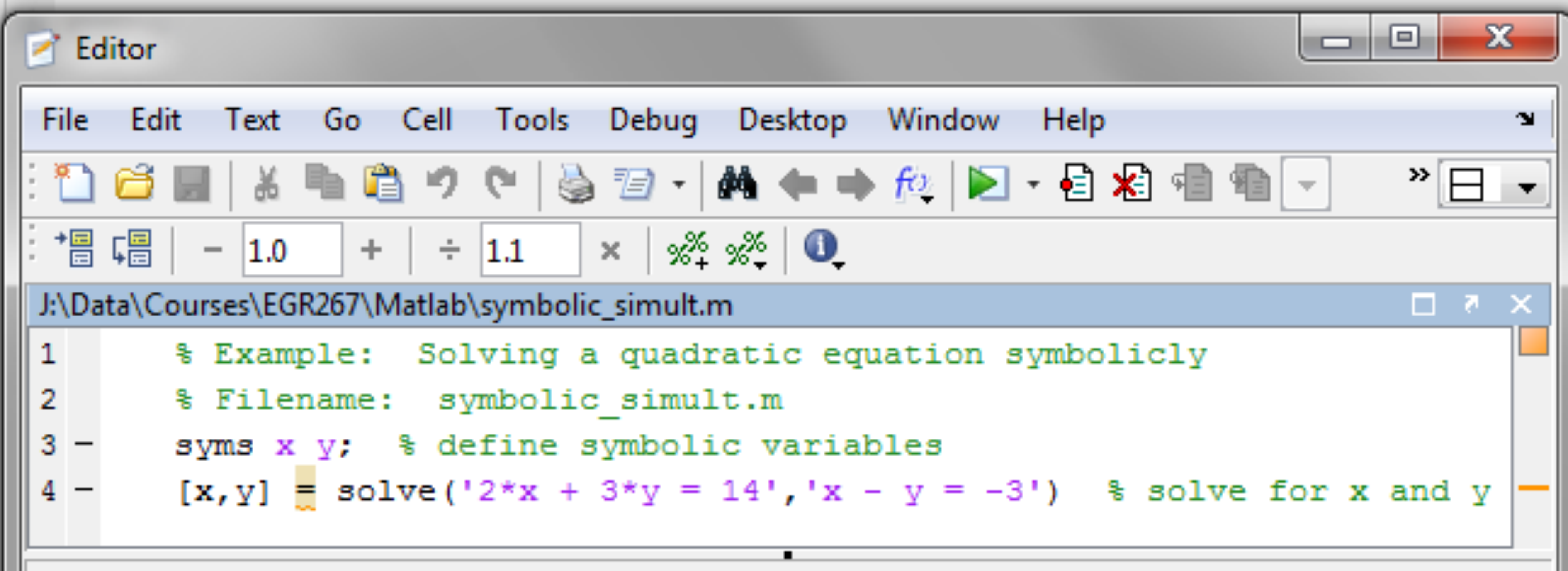
Use MATLAB to solve the following two simultaneous equations (with numeric results): $2x + 3y = 14$

$x - y = -3$

Use the symbolic function **solve( )**.

> Note: We will introduce simpler methods for solving simultaneous equations with numeric results later.

```
EDU>> symbolic_simult
x =
1
y =
4
```



Symbolic substitution:

If we would like to evaluate a symbolic expression for specific values, we can use the symbolic substitution function in MATLAB: **subs( )**

Example: If $V_1 = 10e^{-10t}$, find symbolic expressions for V_1 and $P_1 = (V_1)^2/R_1$. Then use ***subs( )*** to find numeric values for V_1 and P_1 when t = 50 ms and R_1 = 20 ohms.

```
% Symbolic substitution in MATLAB
syms V1 R1 t
V1 = 10*exp(-10*t);
P1 = V1^2/R1    % display symbolic expression for P1
t = 50e-3;      % 50 ms
subs(V1)        % evaluate V1 using t above
R1 = 20;         % ohms
subs(P1)        % evaluate P1 using t and R1 above
```

```
P1 =
(100*exp(-20*t))/R1
ans =
    6.0653
ans =
    1.8394
```

V_1 = 6.0653 V when t = 50 ms

P_1 = 1.8394 W when t = 50 ms and R_1 = 20 ohms

Finding derivatives in MATLAB

diff(f) – finds the derivative of symbolic function f

diff(f, 2) – finds the 2nd derivative of f
(same as diff(diff(f))

If f is a function of both x and y,

diff(f, x) – partial derivative of f w.r.t. x

diff(f, y) – partial derivative of f w.r.t. y

Examples: (shown to the right)

Before proceeding with more advanced derivatives and applications, it may be useful to discuss different ways of ***representing functions in MATLAB***.

```
EDU>> syms x
EDU>> f1 = 2*x^3+4*x
f1 =
2*x^3 + 4*x
EDU>> df1 = diff(f1)
df1 =
6*x^2 + 4
EDU>> f2 = 10*exp(-2*x)
f2 =
10/exp(2*x)
EDU>> df2 = diff(f2)
df2 =
-20/exp(2*x)
EDU>> f3 = 10*cos(40*x)
f3 =
10*cos(40*x)
EDU>> df3 = diff(f3)
df3 =
(-400)*sin(40*x)
```

Performing integration in MATLAB

int(S) – finds the ***indefinite integral*** of a symbolic expression S with respect to its symbolic variable (or variable closest to x)

int(S, z) – finds the ***indefinite integral*** of a symbolic expression S with respect to z

int(S, a, b) – finds the ***definite integral*** of a symbolic expression S from a to b with respect to its symbolic variable (or variable closest to x)

int(S, z, a, b) – finds the ***definite integral*** of a symbolic expression S from a to b with respect to z

double(int(S, z, a, b)) – finds a numeric result for the ***definite integral*** of a symbolic expression S from a to b with respect to z. This is useful in cases where MATLAB can't find a symbolic solution.

MATLAB Examples:

```
% Filename:  integrals1.m
% Example 1:  Integrating with respect to (wrt)
%             a certain variable
syms a c e
int1 = int(a^2*c)    % wrt c since c is "closer" to x
int2 = int(e^2*c)    % wrt e since e is "closer" to x
int3 = int(a^2*c,a) % specify that integral is wrt a
int4 = int(a^2*c,c) % specify that integral is wrt c
% Example 2:  Earlier example done by hand
syms x
int5 = int(4*x+20,x,0,20)
% Example 3:  Integrals of some simple functions
int6 = int(40*cos(4*x),x)
int7 = int(10*exp(2*x),x)
int8 = int(12*sqrt(x),x)
```

```
EDU>> integrals1
int1 =
(a^2*c^2)/2
int2 =
(c*e^3)/3
int3 =
(a^3*c)/3
int4 =
(a^2*c^2)/2
int5 =
1200
int6 =
10*sin(4*x)
int7 =
5*exp(2*x)
int8 =
8*x^(3/2)
```

Classroom Demonstration: Try the following examples on your own or the instructor may demonstrate them in class.

1) Command Window: Try some simple calculations using the command window.
   - Calculate V and P if R = 1.2 kΩ and I = 3.75 mA
   - Try using ***clc***, ***format compact***, and ***clear***
   - Try using ***format long, format rat,*** and ***format short***
   - Try editing expressions already executed using the Command History window or using the up and down arrows.

2) Writing a program requiring inputs and formatted output:

Recall that:

$$R_1 \| R_2 = \frac{1}{\frac{1}{R_1} + \frac{1}{R_2}}$$

Write MATLAB program to do the following:

- Prompt the user to enter the value of R_1 in ohms
- Prompt the user to enter the value of R_2 in ohms
- Calculate $R_1 \| R_2$ and display the result in ohms (with 1 digit after the decimal point).
- If you enter R1 = 20 and R2 = 30, try to format the output to look as follows:
  a) R1 = 20.0 ohms, R2 = 30.0 ohms, Result: 12.0 ohms
  b) R1 = 20.0 ohms
     R2 = 30.0 ohms
     R1 || R2 = 12.0 ohms

Classroom Demonstration: Try the following examples on your own or the instructor may demonstrate them in class.

3) Symbolic manipulation: Solve for IX in the following symbolic expression (KVL in a simple series circuit): $\mathbf{-V + I_X{*}R_1 + I_X{*}R_2 + I_X{*}R_3 = 0}$

(Note that Ix was used instead of I. Avoid using variables i, j, or I in MATLAB.)

4) Calculus:

a) If $Q_1 = 10e^{-4t}$ mC, use MATLAB to find a symbolic expression for I_1

b) If $V_2 = 10e^{-1000t}$ V and $I_2 = 20e^{-2000t}$ mA:

- Find a symbolic expression for $P_2 = V_2{*}I_2$
- Evaluate P_2 at time t = 1ms
- Find a symbolic expression for W_2
- Find the energy from 0 to 1ms