Tables and Graphs in MATLAB

An efficient way to work with tables or graphs in MATLAB is to enter and/or calculate values and place them in matrices or vectors. However, it is first necessary to explain dot operations in MATLAB.

Dot Operations in MATLAB

Dot operations are used in MATLAB to perform operations on an element-by-element basis. Note that the matrices must have the same dimensions. MATLAB's dot operation symbols are shown below.

Operation	Arithmetic Operation	MATLAB Dot Operation
Addition	+	+
Subtraction	-	-
Multiplication	*	.*
Division	/	./
Exponentiation	^	.^

Dot Operations in MATLAB

If matrices A and B have the same dimensions, dot operations work as follows:

C = A + B implies that C(i,j) = A(i,j) + B(i,j)

D = A - B implies that D(i,j) = A(i,j) - B(i,j)

E = A .* B implies that E(i,j) = A(i,j) * B(i,j)

F = A ./ B implies that F(i,j) = A(i,j) / B(i,j)

G = A .^ B implies that D(i,j) = A(i,j) ^ B(i,j)

Example MATLAB program:

```
% Filename:  DotOperators.m
format compact
A=[4 9 8;4 3 2]
B=[2 3 4;1 -1 2]
C=A+B
D=A-B
E=A.*B
F=A./B
G=A.^B
```

Results

```
>> DotOperators
A =
     4     9     8
     4     3     2
B =
     2     3     4
     1    -1     2
C =
     6    12    12
     5     2     4
D =
     2     6     4
     3     4     0
E =
     8    27    32
     4    -3     4
F =
     2     3     2
     4    -3     1
G =
   1.0e+003  *
    0.0160    0.7290    4.0960
    0.0040    0.0003    0.0040
```

Generating a Range of Values in MATLAB

A range of values can be assigned to a variable (a vector) easily in MATLAB.

There are three convenient ways to do this:

1) Use the **colon operator “:”** to vary points from a to b with ***increment*** delta

Syntax: **VariableName = a:delta:b**

Examples:
x = 0:25:100; % What values are assigned?
y = [-5:1:5]; % Note that brackets are optional.
%What values are assigned?

2) Use **linspace( )** to use n points spaced ***linearly*** from a to b

Syntax: **VariableName = linspace(a, b, n)**

Examples:
x = linspace(0,15,25); % 25 points linearly spaced from 0 to 15
y = linspace(-12,12); % 100 points (default) from -12 to 12

3) Use **logspace( )** to use n points spaced ***logarithmically*** from 10^a to 10^b

Syntax: **VariableName = logspace(a, b, n)**

Examples:
x = logspace(1, 3, 75); % 75 points spaced logarithmically from 10 to 1000
y = logspace(-2, 2); % 50 points spaced logarithmically from 0.01 to 100

MATLAB examples using linspace(), logspace() and the colon operator

```
>> x = [0:2:12]
x =
     0     2     4     6     8    10    12
>> y = -6:2:8
y =
    -6    -4    -2     0     2     4     6     8
>> x = linspace(0,2,10)
x =
         0    0.2222    0.4444    0.6667    0.8889    1.1111    1.3333    1.5556    1.7778    2.0000
>> x = linspace(0,2,11)
x =
  Columns 1 through 10
         0    0.2000    0.4000    0.6000    0.8000    1.0000    1.2000    1.4000    1.6000    1.8000
  Column 11
    2.0000
>> y = logspace(1,2,10)
y =
   10.0000   12.9155   16.6810   21.5443   27.8256   35.9381   46.4159   59.9484   77.4264  100.0000
>> x = logspace(1,4,31)
x =
  1.0e+004 *
  Columns 1 through 10
    0.0010    0.0013    0.0016    0.0020    0.0025    0.0032    0.0040    0.0050    0.0063    0.0079
  Columns 11 through 20
    0.0100    0.0126    0.0158    0.0200    0.0251    0.0316    0.0398    0.0501    0.0631    0.0794
  Columns 21 through 30
    0.1000    0.1259    0.1585    0.1995    0.2512    0.3162    0.3981    0.5012    0.6310    0.7943
  Column 31
    1.0000
```

Calculating values for a function over a range

Suppose that we wanted to calculate $v(t) = 50te^{-3t}$ for t = 0 to 1.4 in increments of 0.1

We can easily specify the values for t as follows:

t = 0:0.1:1.4

As a first guess we might try to calculate V using

V = 50*t*exp(-3*t)

but we would get an error in MATLAB because t is a (1x15) vector and exp(-2*t) is also a (1x15) vector, so the dimensions are incorrect for matrix multiplication. So we must use the ***dot operation for multiplication***, or

V = 50*t.*exp(-3*t)

Let's try this out in MATLAB:

```
>> t=0:0.1:1.4;
>> V=50*t*exp(-3*t);
??? Error using ==> mtimes
Inner matrix dimensions must agree.

>> V=50*t.*exp(-3*t);
>> t,V
t =
  Columns 1 through 9
         0    0.1000    0.2000    0.3000    0.4000    0.5000    0.6000    0.7000    0.8000
  Columns 10 through 15
    0.9000    1.0000    1.1000    1.2000    1.3000    1.4000
V =
  Columns 1 through 9
         0    3.7041    5.4881    6.0985    6.0239    5.5783    4.9590    4.2860    3.6287
  Columns 10 through 15
    3.0242    2.4894    2.0286    1.6394    1.3157    1.0497
```

Printing Tables in MATLAB

Notice in the last example that the results for t and V were printed in rows rather than columns. We more typically would like to see a table of results displayed in columns. We can easily correct this by printing the **transpose** of each matrix.

Note that the command

t',V'

did result in the values being printed in columns, but they are not side by side as desired. This will be corrected on the following page.

```
>> t=0:0.1:1.4;
>> V=50*t.*exp(-3*t);
>> t',V'      %Display the transpose of each vector
ans =
         0
    0.1000
    0.2000
    0.3000
    0.4000
    0.5000
    0.6000
    0.7000
    0.8000
    0.9000
    1.0000
    1.1000
    1.2000
    1.3000
    1.4000
ans =
         0
    3.7041
    5.4881
    6.0985
    6.0239
    5.5783
    4.9590
    4.2860
    3.6287
    3.0242
    2.4894
```

A key step to displaying t and V in column format is to create a new (15x2) matrix containing the transpose of matrices t and V. This is illustrated using MATLAB:

```
>> t=0:0.1:1.4;
>> V=50*t.*exp(-3*t);
>> new1=[t',V']
new1 =
         0         0
    0.1000    3.7041
    0.2000    5.4881
    0.3000    6.0985
    0.4000    6.0239
    0.5000    5.5783
    0.6000    4.9590
    0.7000    4.2860
    0.8000    3.6287
    0.9000    3.0242
    1.0000    2.4894
    1.1000    2.0286
    1.2000    1.6394
    1.3000    1.3157
    1.4000    1.0497
```

The final version of the MATLAB instructions, including a table heading, is stored in a MATLAB program called Table and is shown on the next page.

```
% Creating a table of values using MATLAB
% Filename:  Table.m
format compact                  % Suppress extra lines in the output
t=0:0.1:1.4;                    % Set up a range of values for t
V=50*t.*exp(-3*t);              % Use dot multiplication for form V
fprintf('\n\n');                % Print two blank lines
disp('  Time, t(s)  Voltage, V(V)')    % Print table heading
disp([t',V'])                   % Print t and V to column format
```

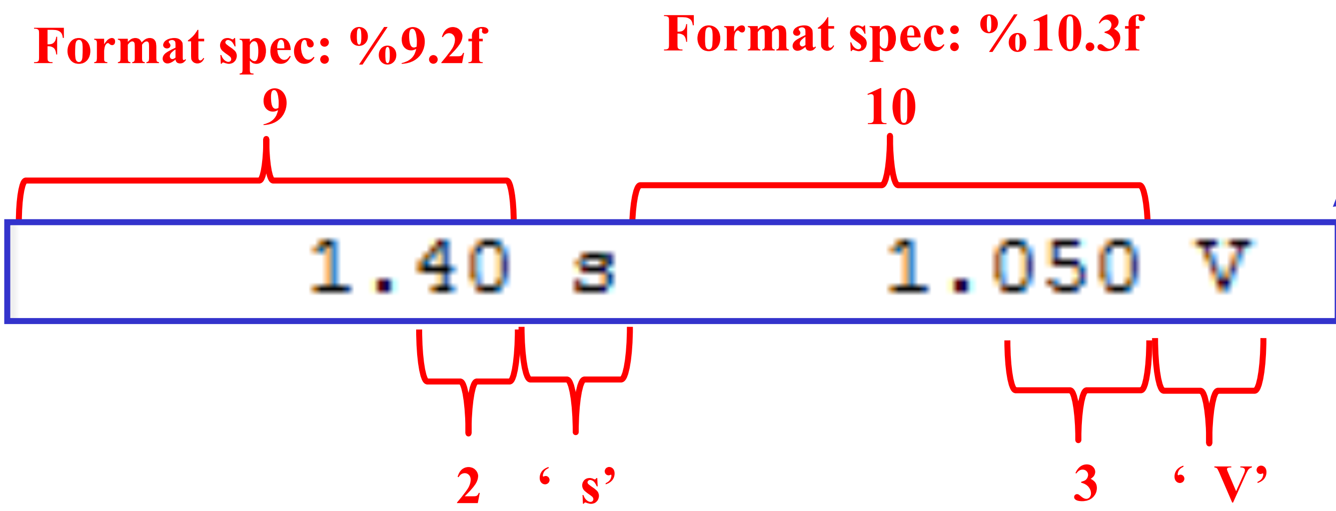
```
>> Table


  Time, t(s)  Voltage, V(V)
         0         0
    0.1000    3.7041
    0.2000    5.4881
    0.3000    6.0985
    0.4000    6.0239
    0.5000    5.5783
    0.6000    4.9590
    0.7000    4.2860
    0.8000    3.6287
    0.9000    3.0242
    1.0000    2.4894
    1.1000    2.0286
    1.2000    1.6394
    1.3000    1.3157
    1.4000    1.0497
```

Formatting Tables – The previous example could be expanded to also include the power dissipated by a 10 ohm resistor and precise formatting.

```
% Create a formatted table of values using MATLAB
% Filename:  TableF.m
format compact
t = 0:0.1:1.4;     % t varies from 0 to 1.4 in increments of 0.1 (15 values)
V = 50*t.*exp(-3*t);     % use dot multiplication to form V
Table = [t',V'];         % form a new matrix named Table
disp('      Time          Voltage')
fprintf('%9.2f s%10.3f V\n', Table')    % print two values per line
% Note that Table' was printed rather than Table as fprintf prints by
% columns and not by rows
```

```
EDU>> TableF
      Time          Voltage
      0.00 s       0.000 V
      0.10 s       3.704 V
      0.20 s       5.488 V
      0.30 s       6.099 V
      0.40 s       6.024 V
      0.50 s       5.578 V
      0.60 s       4.959 V
      0.70 s       4.286 V
      0.80 s       3.629 V
      0.90 s       3.024 V
      1.00 s       2.489 V
      1.10 s       2.029 V
      1.20 s       1.639 V
      1.30 s       1.316 V
      1.40 s       1.050 V
```



Class Examples - Try one or more of the following examples in class:

Example 1:

Create a formatted table of values for the parabolic function $v(t) = 10t^2$.

Let t vary linearly from -2 to +2 using 41 points.

Use ***linspace***( ), but discuss how this could be done with the colon(:) operator.

Example 2:

Create a table of values for the function $$LM(w) = 20\log\left[\frac{100}{\sqrt{100^2 + w^2}}\right]$$

Let w vary logarithmically from 10 to 1000 using 10 points per decade.

Use ***logspace***( ).

Graphing in MATLAB

Several types of graphs can be created in MATLAB, including:

- x-y charts
- column charts (or histograms)
- contour plots
- surface plots

x-y charts are most commonly used in engineering courses, so we will focus on these.

x-y Charts in MATLAB

The table on the next page shows several commands that are available in MATLAB for plotting x-y graphs.

Table of MATLAB Plotting Commands

MATLAB Command	**Description**
plot(x,y)	**Plots y values versus x values (x and y values stored in vectors).**
plot(x,y,S)	**String S can be used to specify various line types, symbols, and colors as shown in the table on the next page.**
loglog(x,y)	**Like plot(x,y) but uses log-log axes**
semilogx(x,y)	**Like plot(x,y) but uses log scale for x-axis**
semilogy(x,y)	**Like plot(x,y) but uses log scale for y-axis**
plot(x1,y1,S1,x2,y2,S2,…)	**Use for plotting multiple curves on the same graph (or use hold)**
hold	**Plot the next set of data on the same chart**
figure	**Open a new window (figure) for the graph – otherwise a second plot command will overwrite the current figure.**
title('text')	**Adds a title to the top of the graph.**
xlabel('text')	**Adds a label to the x-axis.**
ylabel('text')	**Adds a label to the y-axis.**
grid on (or grid)	**Turns on the grid lines.**
grid off	**Turns off the grid lines.**
grid minor	**Turns on the minor grid lines.**

Table of S Options for the MATLAB plot(x,y,S) Command

S	Color	S	Data Symbol	S	Line Type
b	Blue	.	Point	-	Solid
g	Green	O	Circle	:	Dotted
r	Red	x	x-mark	-.	Dash-dot
c	Cyan	+	Plus	--	Dashed
m	Magenta	*	Star		
y	Yellow	s	Square		
k	Black	d	Diamond		
		v	Triangle (down)		
		^	Triangle (up)		
		<	Triangle (left)		
		>	Triangle (right)		

Note: Many of these options can be set using the Graph Property Editor (illustrated in the following pages)

Examples of using S options in the plot command:

plot(x, y, ‘k*-’) -Plot y versus x using a solid black line with an * marker shown for each point

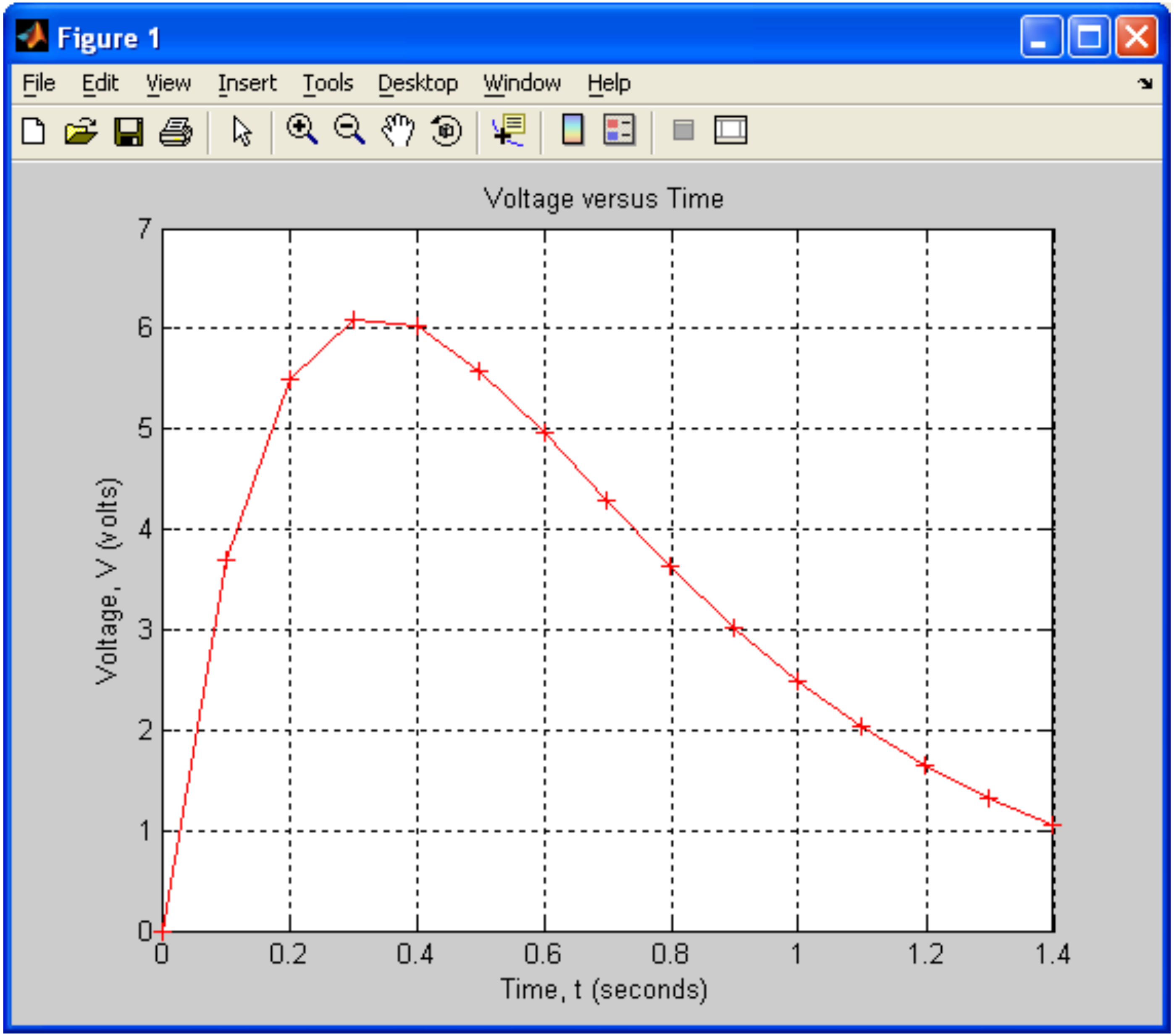
plot(x, y, ‘gd:’) -Plot y versus x using a dotted green line with a diamond marker shown for each point

plot(x, y, ‘r+--’) -Plot y versus x using a dashed red line with a + marker shown for each point

Example – MATLAB program to plot an x-y graph:

```
% Creating an x-y graph using MATLAB
% Filename:  Graph.m
format compact           % Suppress extra lines in the output
t=0:0.1:1.4;             % Set up a range of values for t
V=50*t.*exp(-3*t);       % Use dot multiplication for form V
plot(t,V,'r+-')          % Plot V vs t using a solid red line with + markers
title('Voltage versus Time')        % Add title to graph
xlabel('Time, t (seconds)')         % Add x-axis label to graph
ylabel('Voltage, V (volts)')        % Add y-axis label to graph
grid                                % Add gridlines to graph
```

Enter the name of the program (Graph) in the MATLAB Command Window and the graph (Figure 1) on the following page appears.



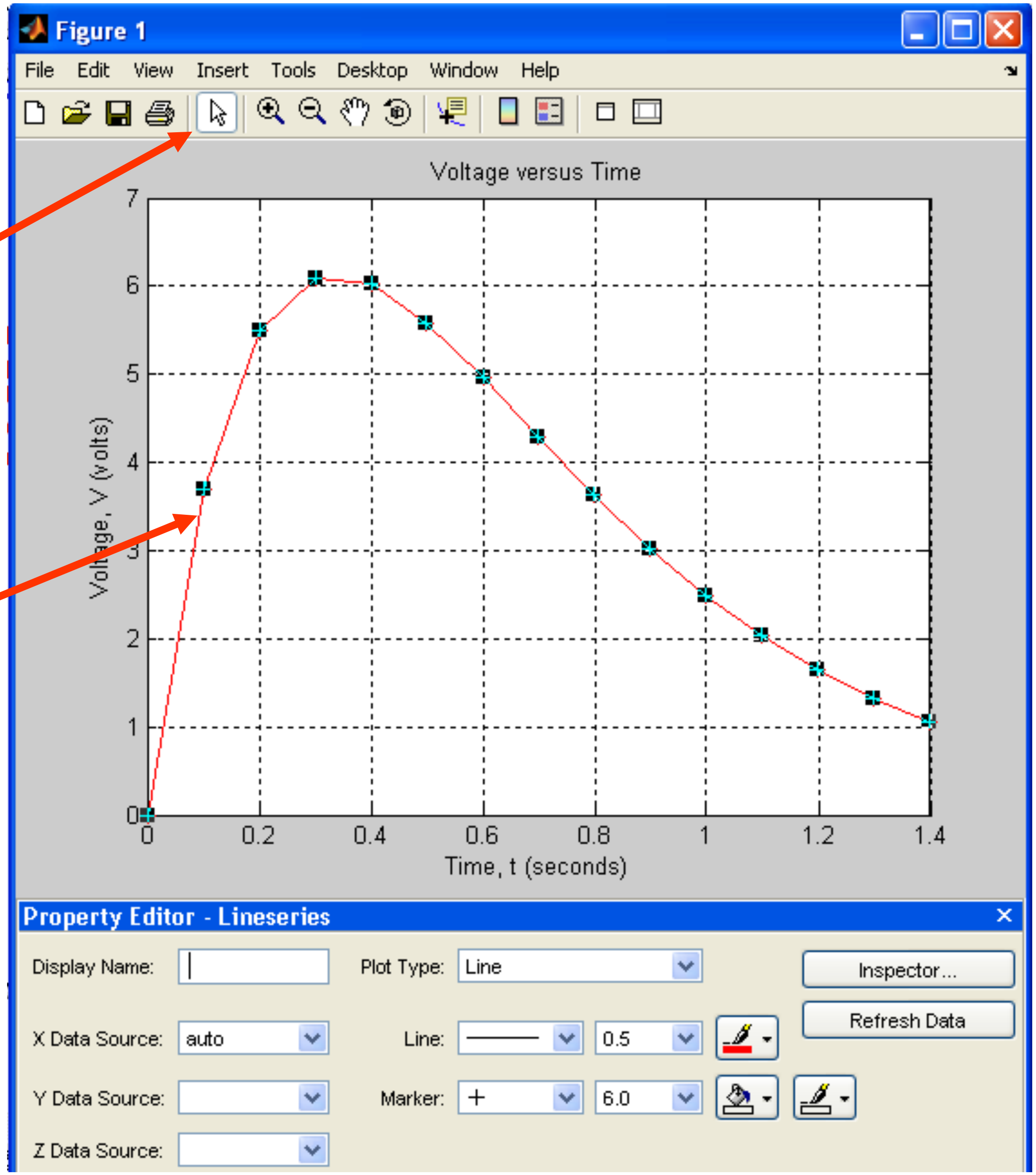
Note that certain features of the graph may be edited from this window (called the Graph Property Editor).

See next page.

Editing a graph - Changing Line Series Properties

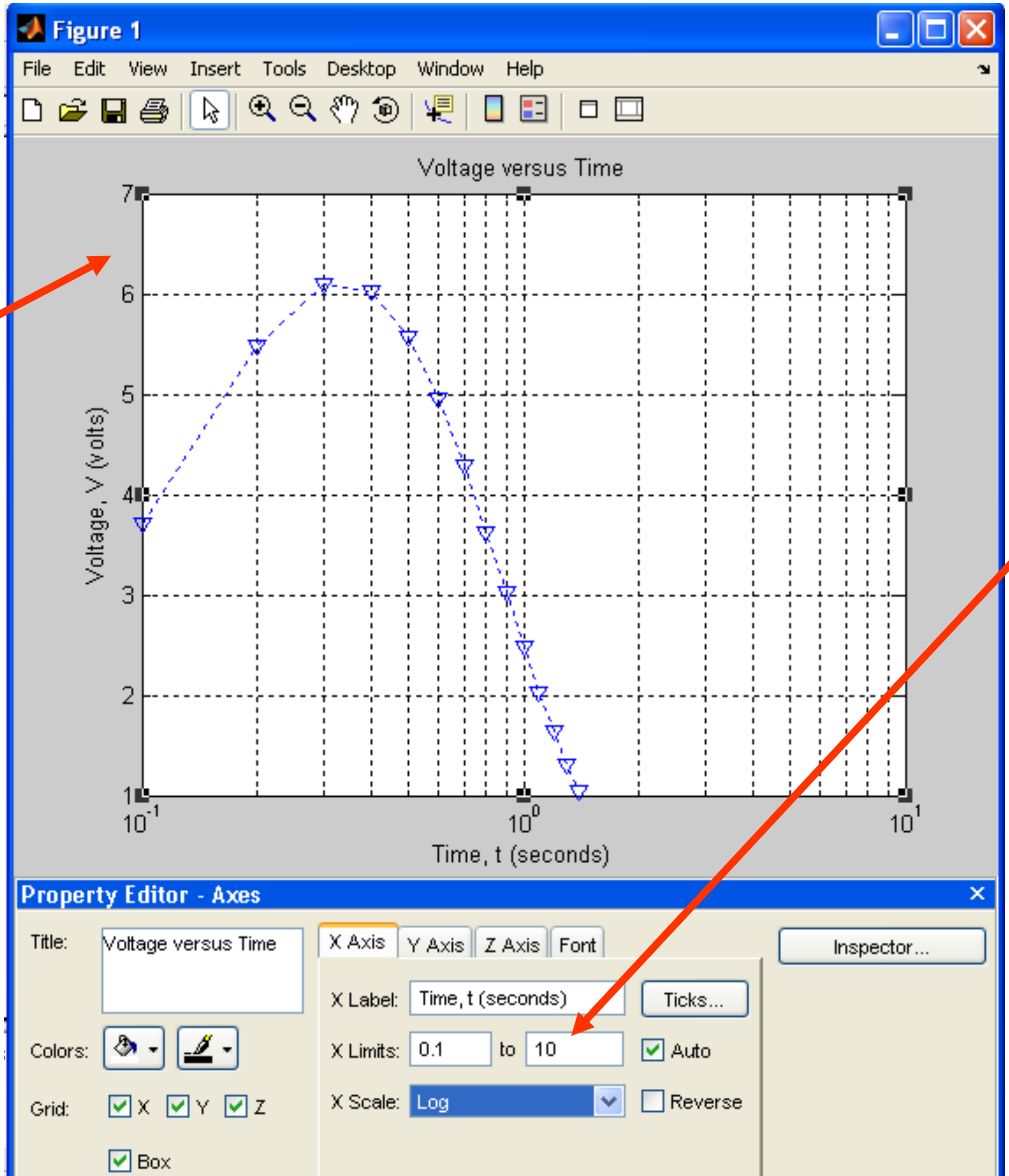
Begin by pressing this button to enter “picking mode”.

Double-click on a graph line or point to open the Property Editor – Lineseries shown below.



You can change various line series properties here.

Editing a graph - Changing Axes Properties



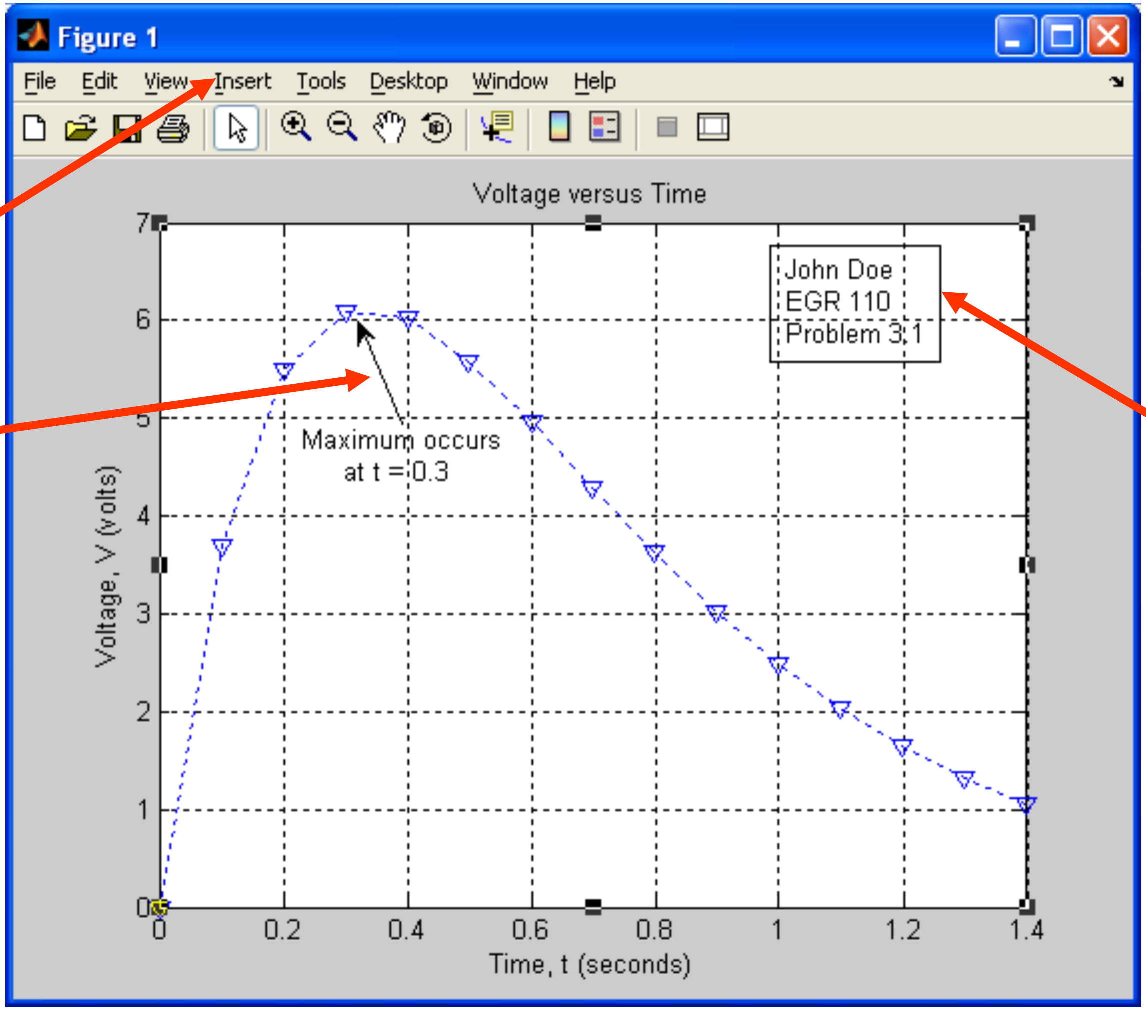
Double-click on an axis (or a point on an axis) to open the Property Editor – Axes shown below.

Note that the x-axis was changed to a log scale.

You can change various axes properties here.

Editing a graph - Using features on the Insert menu.

Use Insert – Text Arrow to add the arrow and text shown on the graph.



Use Insert – Text Box to add the text box shown on the graph.

Note that various other features of graphs can be changed using the Graph Properties Editor above.

Character Sequence	Symbol	Character Sequence	Symbol	Character Sequence	Symbol
\alpha	α	\upsilon	υ	\sim	~
\angle	∠	\phi	Φ	\leq	≤
\ast	*	\chi	χ	\infty	∞
\beta	β	\psi	ψ	\clubsuit	♣
\gamma	γ	\omega	ω	\diamondsuit	♦
\delta	δ	\Gamma	Γ	\heartsuit	♥
\epsilon	ε	\Delta	Δ	\spadesuit	♠
\zeta	ζ	\Theta	Θ	\leftrightarrow	↔
\eta	η	\Lambda	Λ	\leftarrow	←
\theta	Θ	\Xi	Ξ	\Leftarrow	⇐
\vartheta	ϑ	\Pi	Π	\uparrow	↑
\iota	ι	\Sigma	Σ	\rightarrow	→
\kappa	κ	\Upsilon	ϒ	\Rightarrow	⇒
\lambda	λ	\Phi	Φ	\downarrow	↓
\mu	µ	\Psi	Ψ	\circ	°
\nu	ν	\Omega	Ω	\pm	±
\xi	ξ	\forall	∀	\geq	≥
\pi	π	\exists	∃	\propto	∝
\rho	ρ	\ni	∍	\partial	∂
\sigma	σ	\cong	≅	\bullet	•
\varsigma	ς	\approx	≈	\div	÷
\tau	τ	\Re	ℜ	\neq	≠
\equiv	≡	\oplus	⊕	\aleph	ℵ
\Im	ℑ	\cup	∪	\wp	℘
\otimes	⊗	\subseteq	⊆	\oslash	∅
\cap	∩	\in	∈	\supseteq	⊇
\supset	⊃	\lceil	⌈	\subset	⊂
\int	∫	\cdot	·	\o	o
\rfloor	⌋	\neg	¬	\nabla	∇
\lfloor	⌊	\times	x	\ldots	...
\perp	⊥	\surd	√	\prime	′
\wedge	∧	\varpi	ϖ	\0	∅
\rceil	⌉	\rangle	〉	\mid	$\mid$
\vee	∨			\copyright	©
\langle	〈				

TeX Character Sequence Table

Character sequences can be used in text strings for titles and axis labels.

Additional sequences can be used to change text properties (bold, italics, font, color, etc), but they are not listed here.

Unfortunately, these character sequences do not seem to work well in **fprintf**.

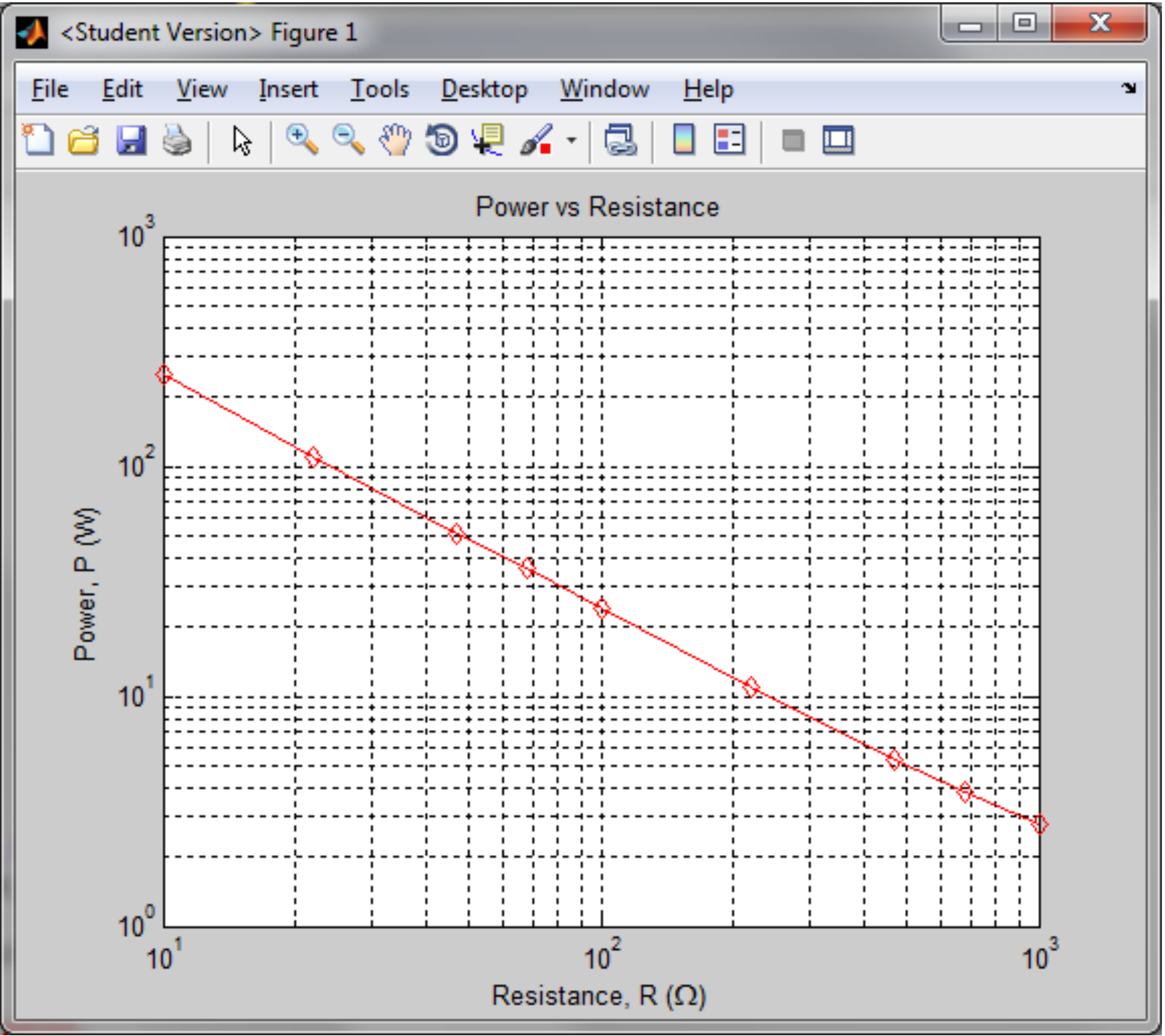
Graphing data points

Graphing data points is similar to graphing functions. The data points simply need to be stored in vectors.

Example: Suppose that the power, P, dissipated by a resistor was measured in lab for a variety of resistance, R, values using a constant voltage of 50 V. The measured results are shown below. Graph P versus R using log scales. Include the Greek letter Ω in the x-axis label.

Resistance R (Ω)	Power P (W)
10	248.3
22	110.2
47	50.4
68	36.1
100	24.2
220	11.0
470	5.25
680	3.85
1000	2.75

```
% Graph power dissipated by a resistor using values of P measured in lab
% for various values of resistance.  Use log scales.
% Filename:  Graph_Power_vs_R_using_data.m
format compact
R = [10,22,47,68,100,220,470,680,1000];    % Data for R
P = [248.3 110.2 50.4 36.1 24.2 11.0 5.25 3.85 2.75]    % Data for P
loglog(R,P,'rd-')
title('Power vs Resistance')
xlabel('Resistance, R (\Omega)')    % use character sequence for ohms
ylabel('Power, P (W)')
grid
```



Note that Greek letter Ω appears in the x-axis label.

Class Examples - Try one or more of the following examples in class:

Example 1:

- Calculate the max voltage that can be applied to a ¼ W resistor (i.e., $P_{max} = 0.25$ W) for various values of resistors.

$$V_{max} = \sqrt{P_{max} \cdot R}$$

- Let R vary linearly from 500 to 10,000 ohms using 20 points.
- Display the results in a table.
- Graph area versus radius.

Example 2:

- Let t vary linearly from 0 to 2 µs in steps of 0.2 µs
- Define $V_{theoretical}$ as $100e^{-2.5t}$ V
- Define $V_{measured}$ = [100, 70.5, 41.2, 25.1, 15.9, 10.4, 7.05, 4.95, 3.85, 2.50, 1.90] in volts.
- Graph $V_{measured}$ versus t and $V_{theoretical}$ versus t on the same graph. By convention show a line only for the theoretical curve and both points and line for the measured. Include a legend.