15-Annotating Plots

text: Chapter 5.1-5.4

ECEGR 101

Engineering Problem Solving with Matlab

Professor Henry Louie

Overview

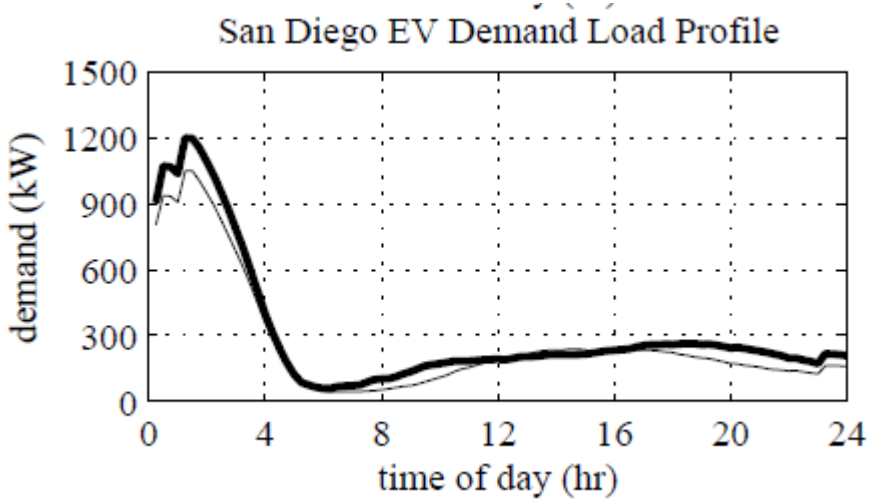
- Multiple Plots
- Axis and Title Labeling
- Legends
- Axis Limits and Labels
- Advanced Editing

Annotating Plots

- Effective plots often contain
  - x- and y-axis labels
  - Title
  - Legend
  - Arrows or other text
  - Multiple traces
- We will discuss how to format and annotate plots in this lecture

Plotting Multiple Graphs

- There are several ways of plotting multiple graphs on the same figure



Plotting Multiple Graphs

1. Use the plot command.

```
x = 0:0.1:10;
y1 = log(x);
y2 = log10(x);
y3 = log2(x);

plot(x, y1, 'r-', x, y2, 'g--', x, y3, 'b:')
```

This can get cumbersome quickly, especially if you want to specify line properties

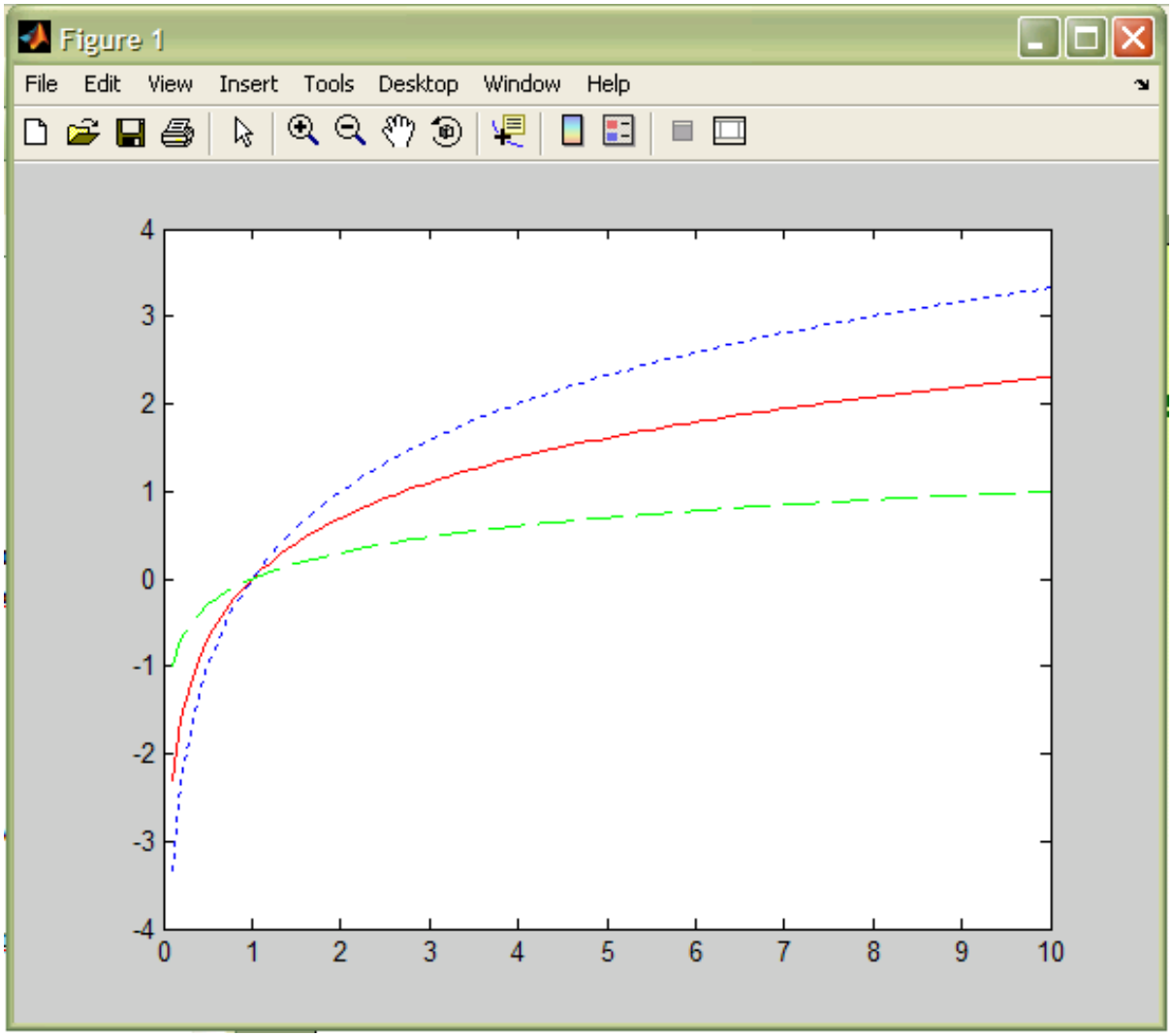


Figure 1
File Edit View Insert Tools Desktop Window Help
4
3
2
1
0
-1
-2
-3
-4
0 1 2 3 4 5 6 7 8 9 10

Plotting Multiple Graphs

2. Use the hold on command.

```
plot(x, y1, 'r-')
hold on
plot(x, y2, 'g--')
plot(x, y3, 'b:')
```

Use the "hold off" command to end holding on a graph

Exercise

Create a new script file and do the following:

a) Use one **plot** command to plot 2sin(x) and cos(4x) versus x in one figure for the values of x between 0 and 3π in increments of $\pi/20$.

b) Comment out the plot command you used in part a (add a % in front of it). Now use the **plot** and **hold** commands to plot 2sin(x) and cos(4x) versus x in one figure for the values of x between 0 and 3π in increments of $\pi/20$.

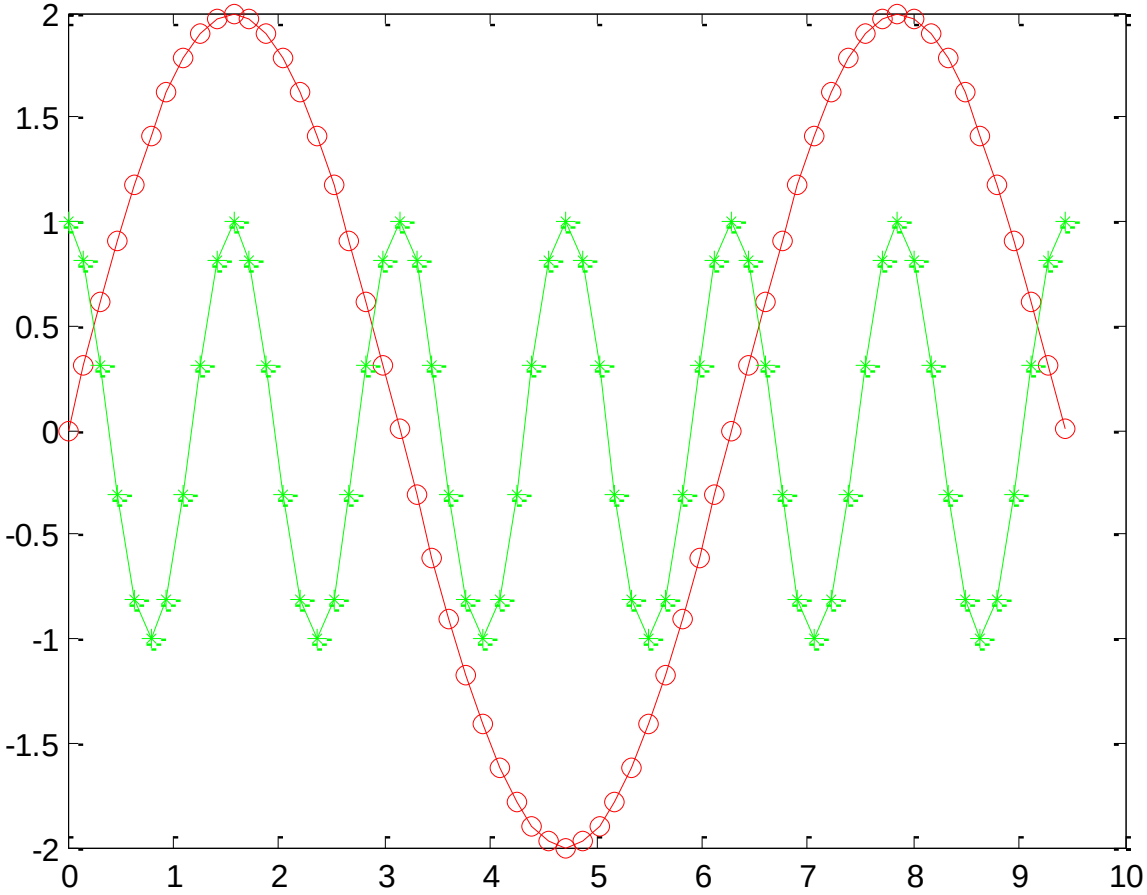
Exercise

```
% Exercise

x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);

plot(x, y1, 'o-r', x, y2, '*-g');
```

Exercise

```
% Exercise 5

x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);

%plot(x, y1, 'o-r', x, y2, '*-g');

plot(x, y1, 'o-r');
hold on
plot(x, y2, '*-g');
```

Exercise



Plot Formatting (Annotating)

xlabel('text string')
ylabel('text string')

Adds text next to the x or y axis.

title('text string')

Adds text on the top of the graph.

text(x, y, 'text string')

Adds text at position (x,y).

Example

```
x = 0:0.01:10;
y = x.^2 - 10*x + 15;
figure(1)
plot(x, y)
xlabel('x')
ylabel('y')
title('Plot of y = x.^2 - 10*x + 15');
text(5, 4, 'Tuesday, 10/17/2006')
```

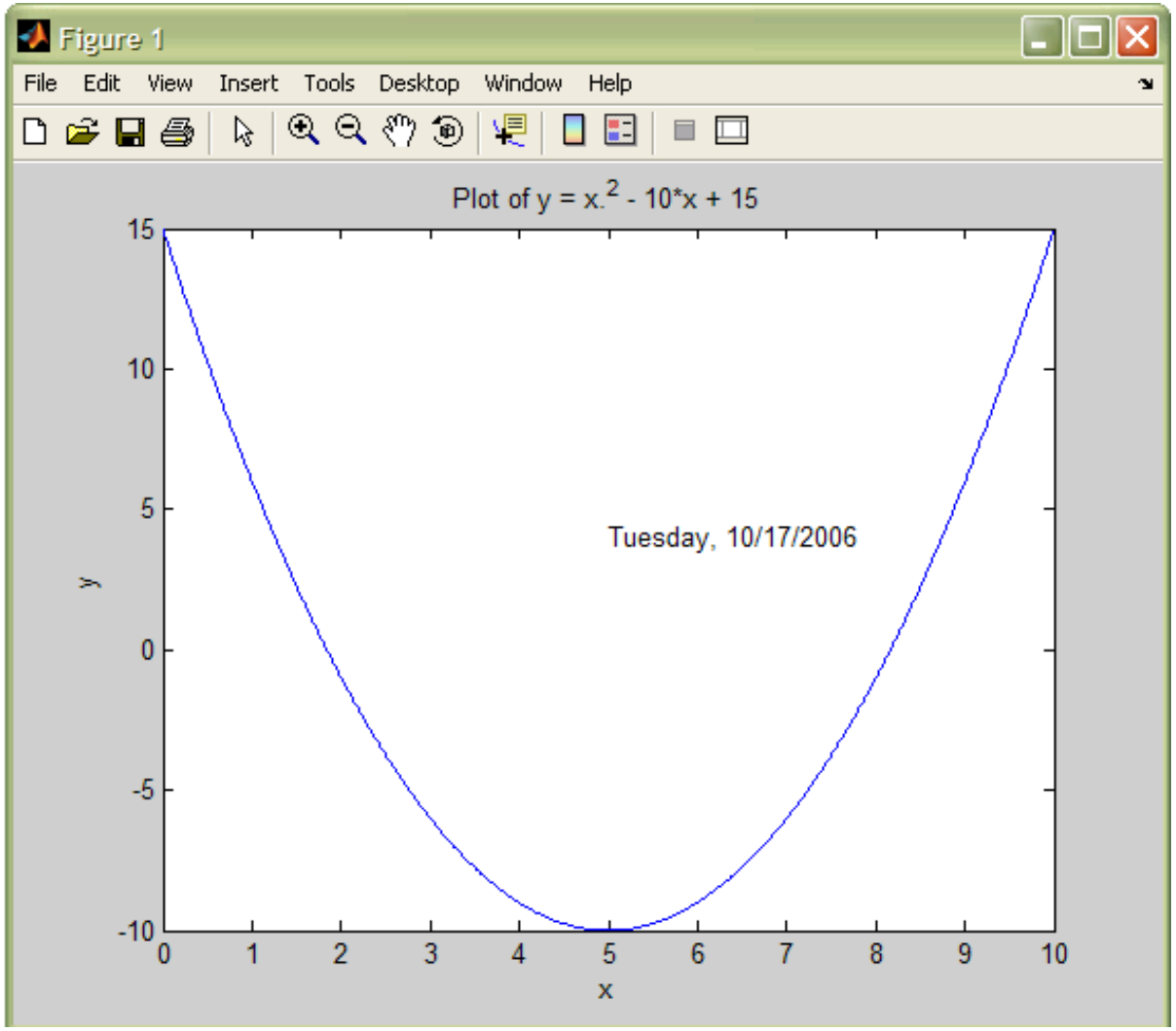


Figure 1
File Edit View Insert Tools Desktop Window Help
Plot of y = x.^2 - 10*x + 15
Tuesday, 10/17/2006
y
x
15
10
5
0
-5
-10
0 1 2 3 4 5 6 7 8 9 10

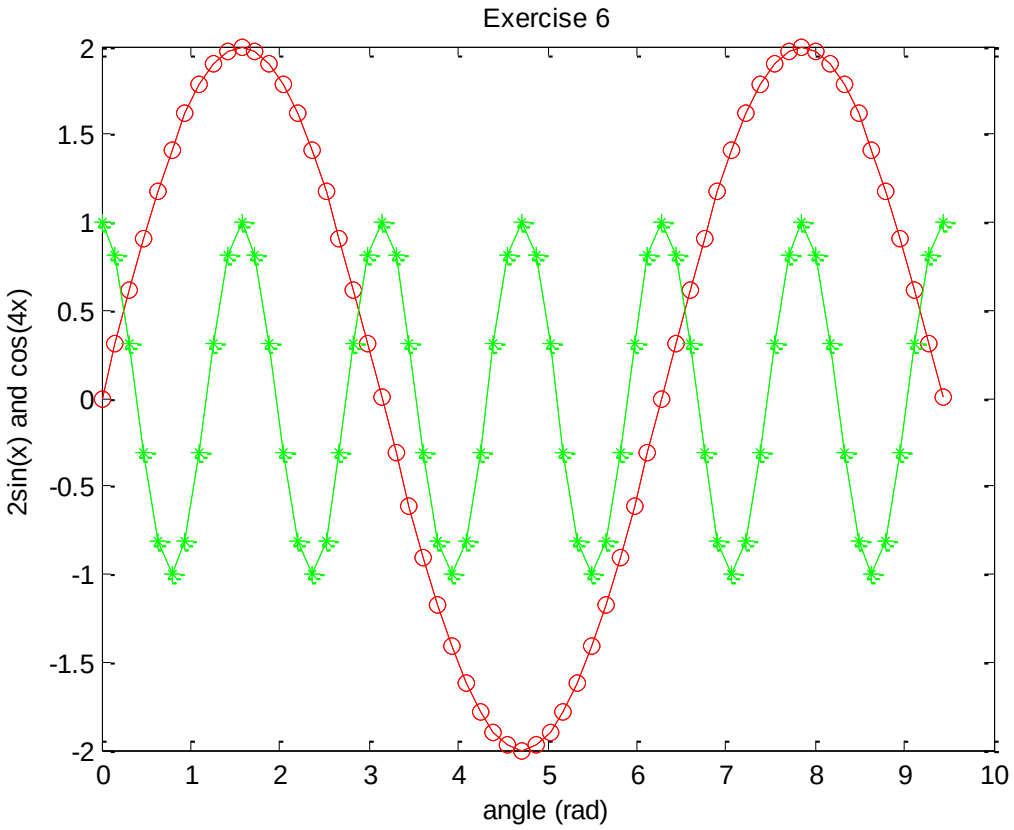
Exercise

Add labels for the x and y axis of your plot from the last exercise. Also add a title.

Exercise

```
% Exercise

x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);

plot(x, y1, 'o-r');
hold on
plot(x, y2, '*-g');

xlabel('angle (rad)');
ylabel('2sin(x) and cos(4x)');
title('Exercise 6')
```

Exercise


Exercise 6
2sin(x) and cos(4x)
angle (rad)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6
7
8
9
10

Legends

legend('string1', 'string2', …, 'Location', pos)

Labels (same order as the order of the plotting commands).

Plot Formatting (Annotating)

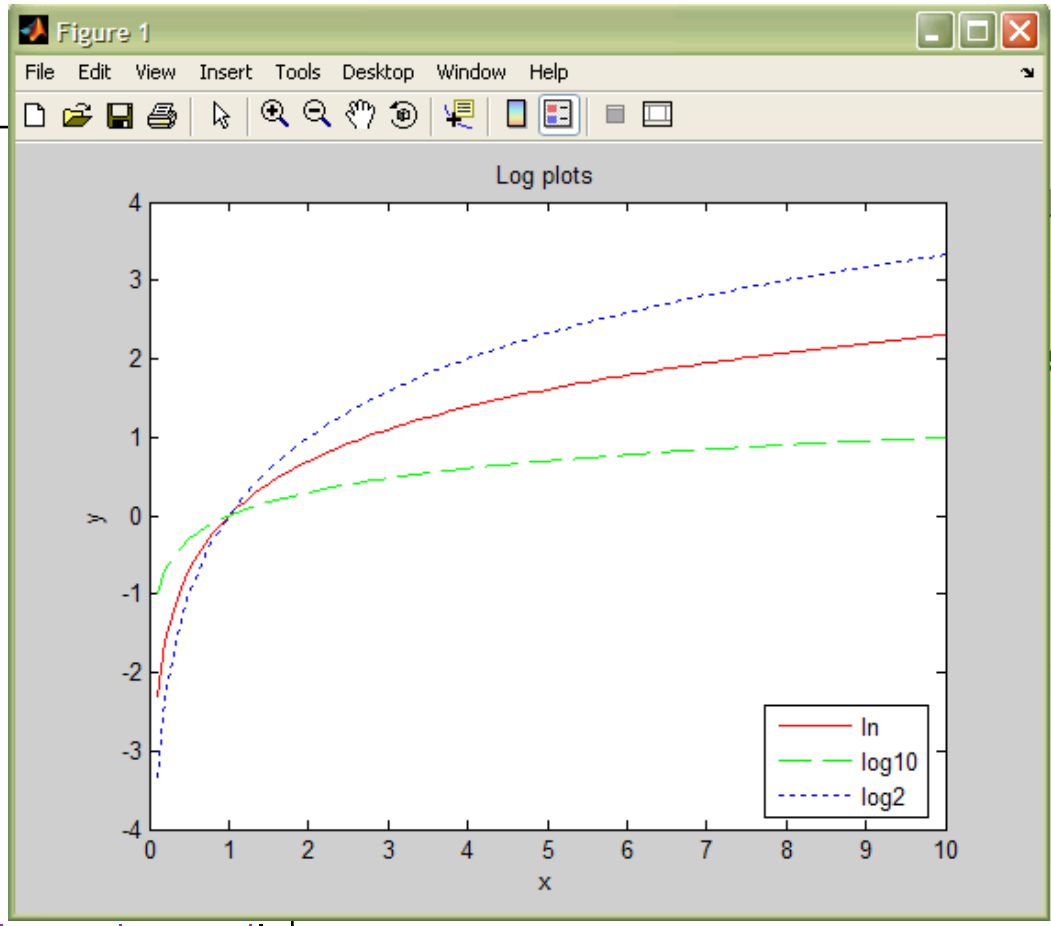
legend('string1', 'string2', …, 'Location', pos)

Specifier	Location in Axes
North	inside plot box near top
South	inside bottom
East	inside right
West	inside left
NorthEast	inside top right (default)
NorthWest	inside top left
SouthEast	inside bottom right
SouthWest	inside bottom left

Specifies where the legend is positioned (optional).

Example

```
x = 0:0.1:10;
y1 = log(x);
y2 = log10(x);
y3 = log2(x);

plot(x, y1, 'r-')
hold on
plot(x, y2, 'g--')
plot(x, y3, 'b:')

xlabel('x');
ylabel('y');
title('Log plots')
legend('ln', 'log10', 'log2', 'Location', 'SouthEast')
```



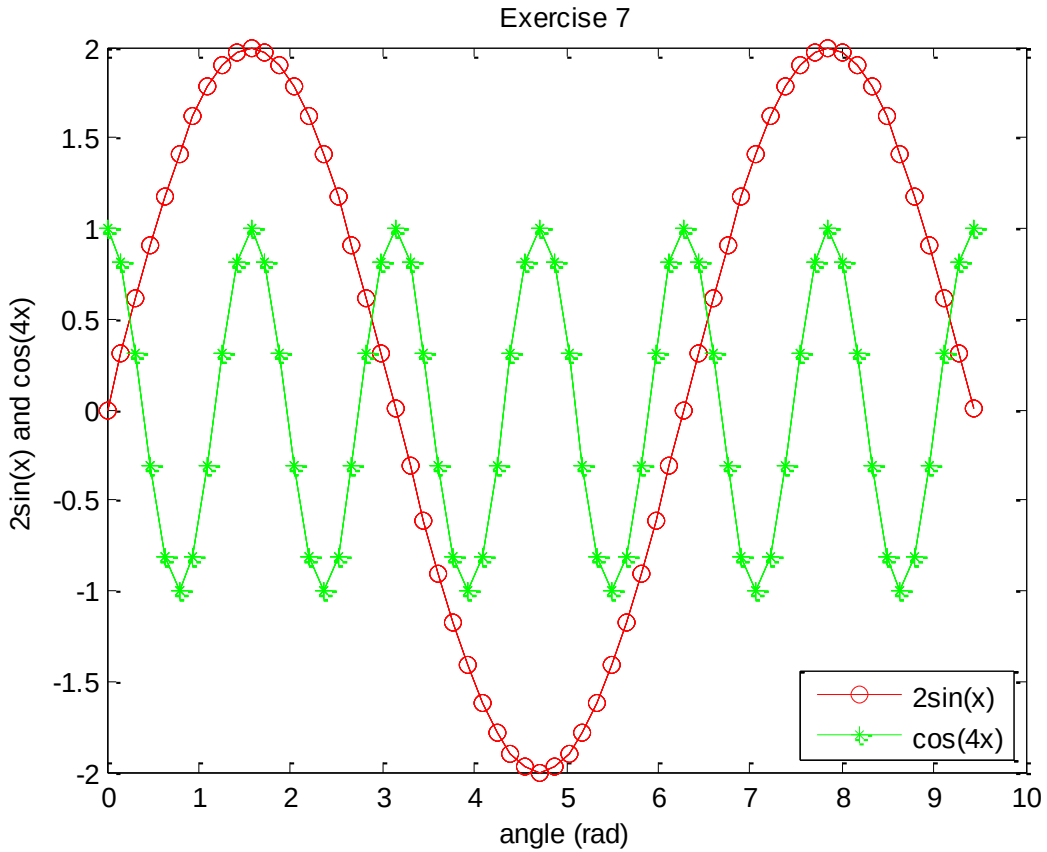
Exercise

Add a legend to your plot from the previous exercise in the bottom, right corner.

Exercise

```
% Exercise

x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);

plot(x, y1, 'o-r');
hold on
plot(x, y2, '*-g');

xlabel('angle (rad)');
ylabel('2sin(x) and cos(4x)');
title('Exercise 7')

legend('2sin(x)', 'cos(4x)', 'Location', 'SouthEast');
```

Exercise


Exercise 7
2sin(x) and cos(4x)
angle (rad)
2sin(x)
cos(4x)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6
7
8
9
10

Formatting Text

- How can we format the text in xlabel, ylabel, title, text, and legend commands?

1. Add modifiers inside the string:

   \bf bold

   \it italic

   \fontname{name}

   \fontsize{size}

Example

```
xlabel('\fontname{Verdana} x');
ylabel('\bf y');
title('\fontsize{14}Log plots')
legend('\itln', '\bflog10', '\itlog2', 'Location', 'SouthEast')
```

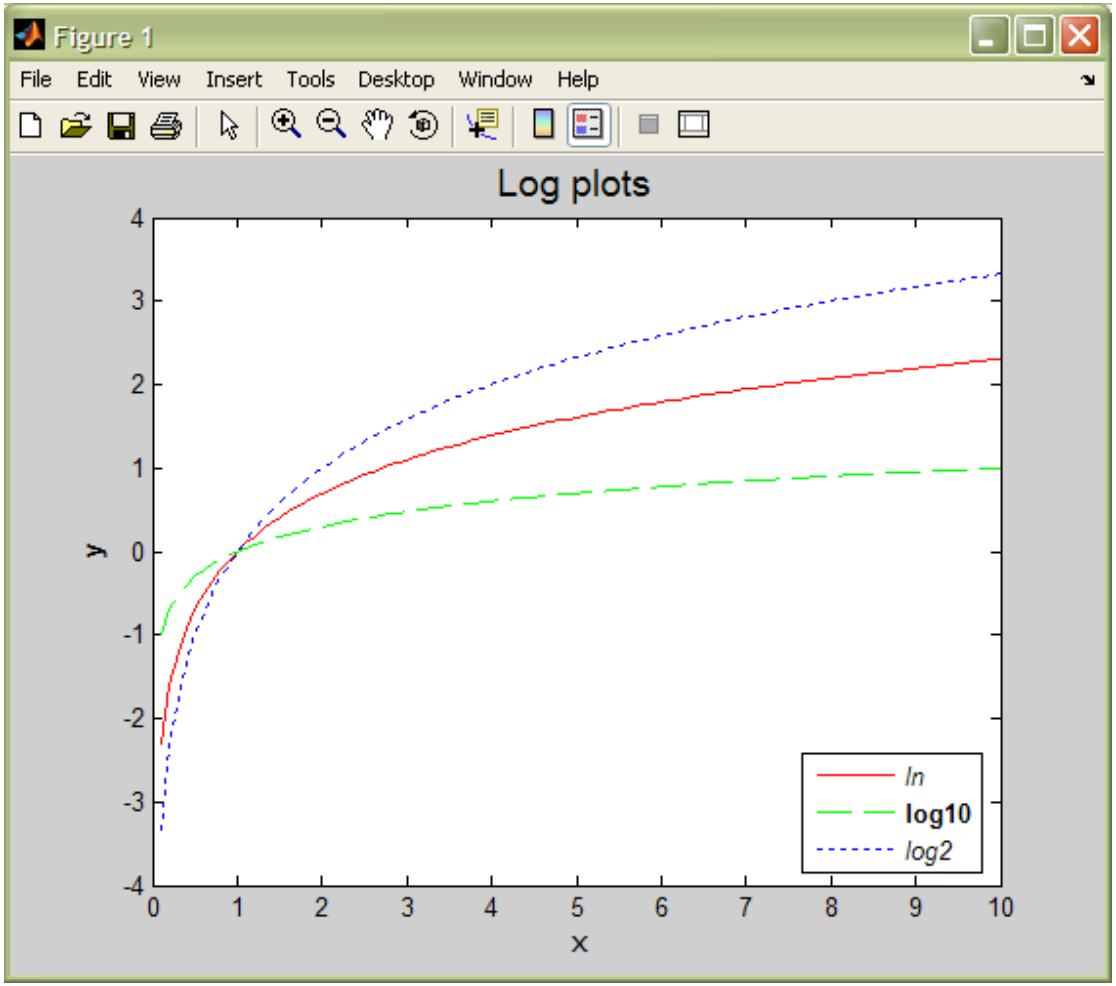


Figure 1
File Edit View Insert Tools Desktop Window Help
Log plots
y
x
ln
log10
log2

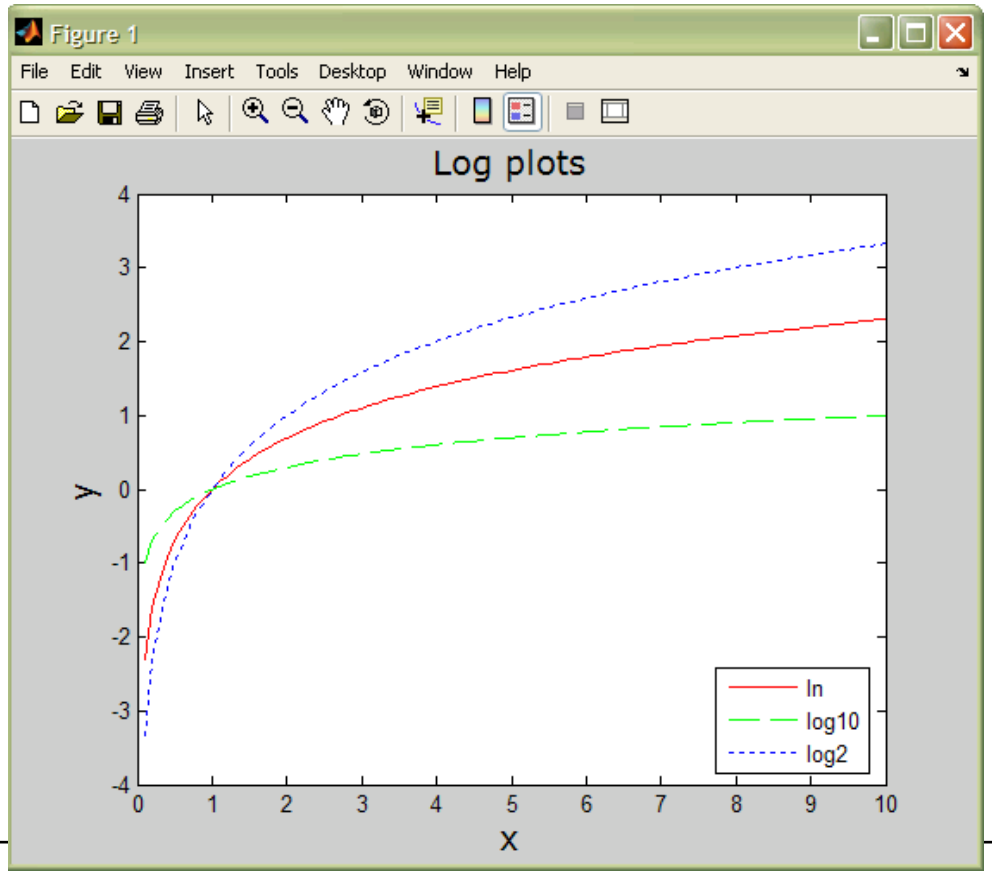
Formatting Text

2. Use PropertyName and PropertyValue:

xlabel('string', PropertyName, PropertyValue)

PropertyName: Rotation (degrees), FontAngle (normal/italic), FontName, FontSize, FontWeight (light/normal/bold),
Color, BackgroundColor, EdgeColor, LineWidth.

Example




```
xlabel('x', 'FontName', 'Verdana', 'FontSize', 14);
ylabel('y', 'FontName', 'Verdana', 'FontSize', 14);
title('Log plots', 'FontName', 'Verdana', 'FontSize', 14)
```

Exercise

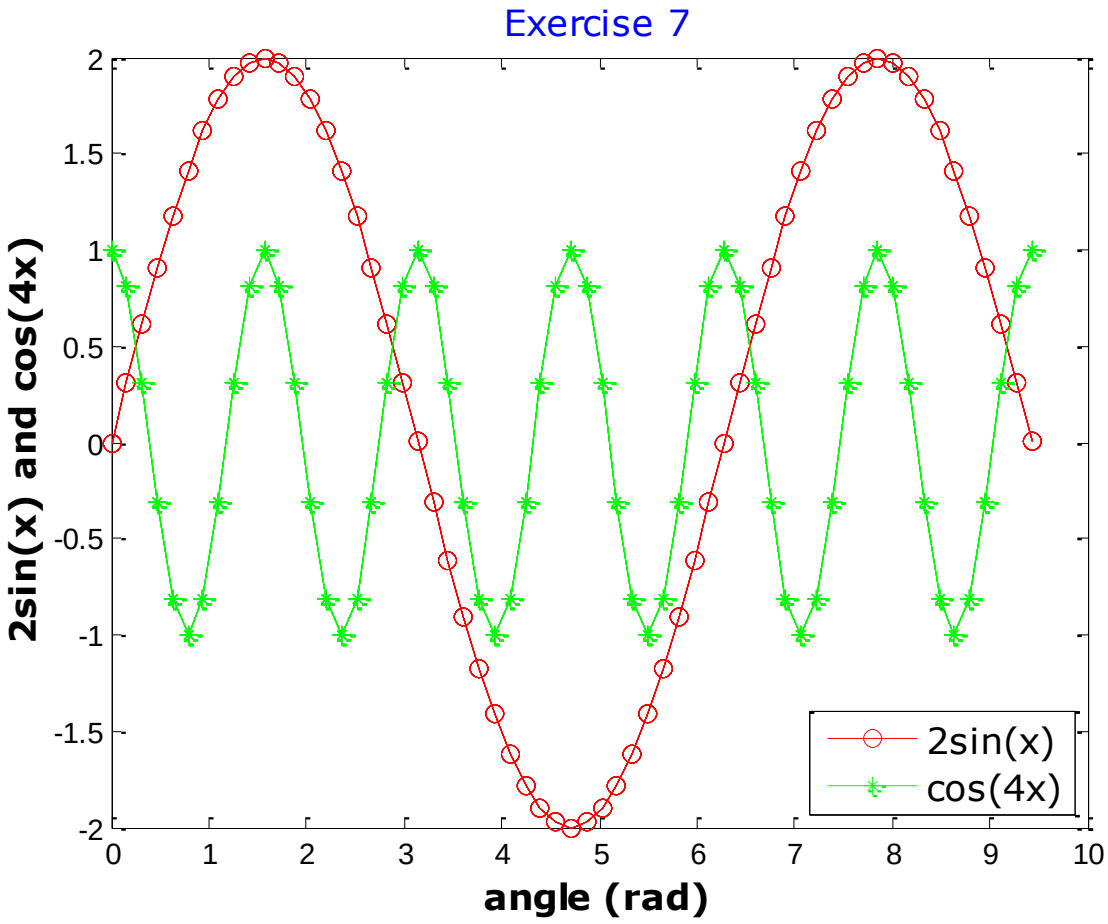
Add any modifiers you would like to the xlabel, ylabel, title, and legend commands. Try both methods:

- use modifiers inside the string,
- use propertyName and propertyValue.

Exercise

```
% Exercise
x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);
plot(x, y1, 'o-r');
hold on
plot(x, y2, '*-g');

xlabel('\bf\fontname{Verdana}\fontsize{12}angle (rad)');
ylabel('\bf\fontname{Verdana}\fontsize{12}2sin(x) and
    cos(4x)');
title('Exercise 7', 'FontName', 'Verdana', 'Color', 'b',
    'FontSize', 12)
legend('\fontname{Verdana}\fontsize{12}2sin(x)',
    '\fontname{Verdana}\fontsize{12}cos(4x)', 'Location',
    'SouthEast');
```

Exercise


Exercise 7
2sin(x) and cos(4x)
angle (rad)
2sin(x)
cos(4x)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6
7
8
9
10

Controlling the Axis

axis([xmin xmax ymin ymax])

Example:

```
axis([1 10 0 4])
```

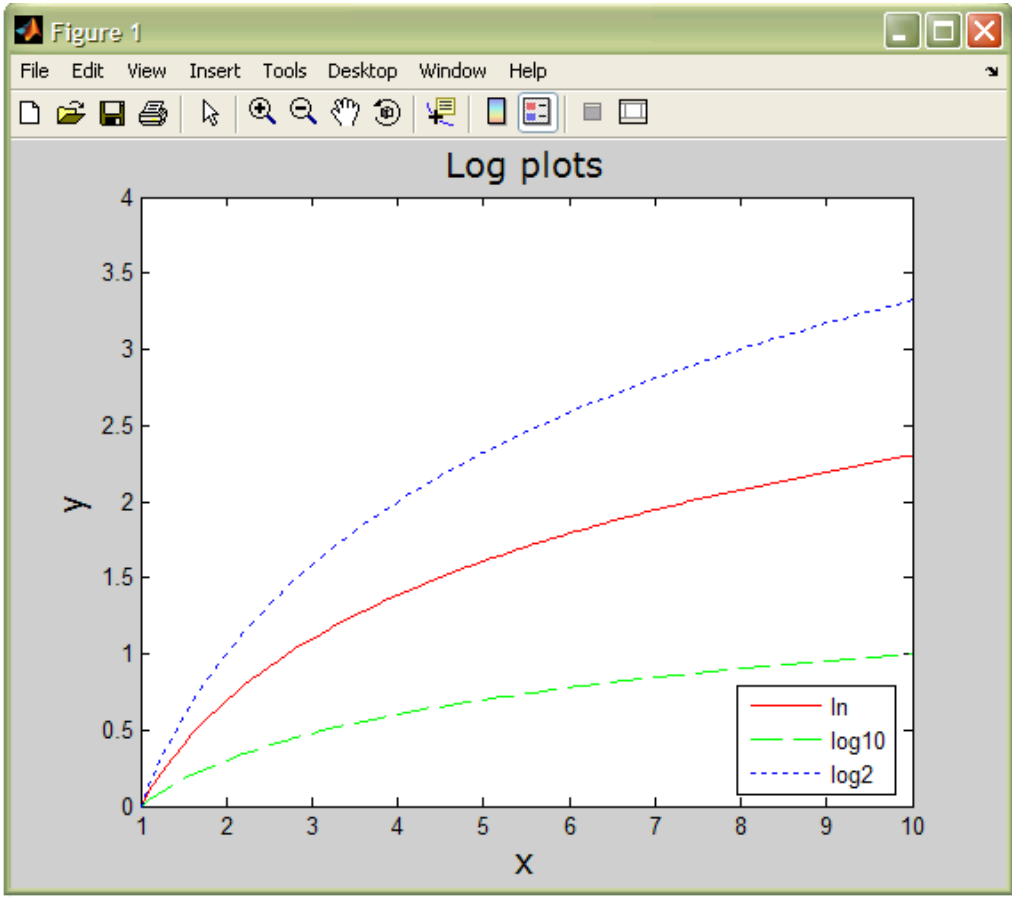


Figure 1
File Edit View Insert Tools Desktop Window Help
Log plots
y
x
ln
log10
log2

Adding/Removing Grid

grid on – adds grid lines to the plot

grid off – removes grid lines from the plot

```
grid on
```

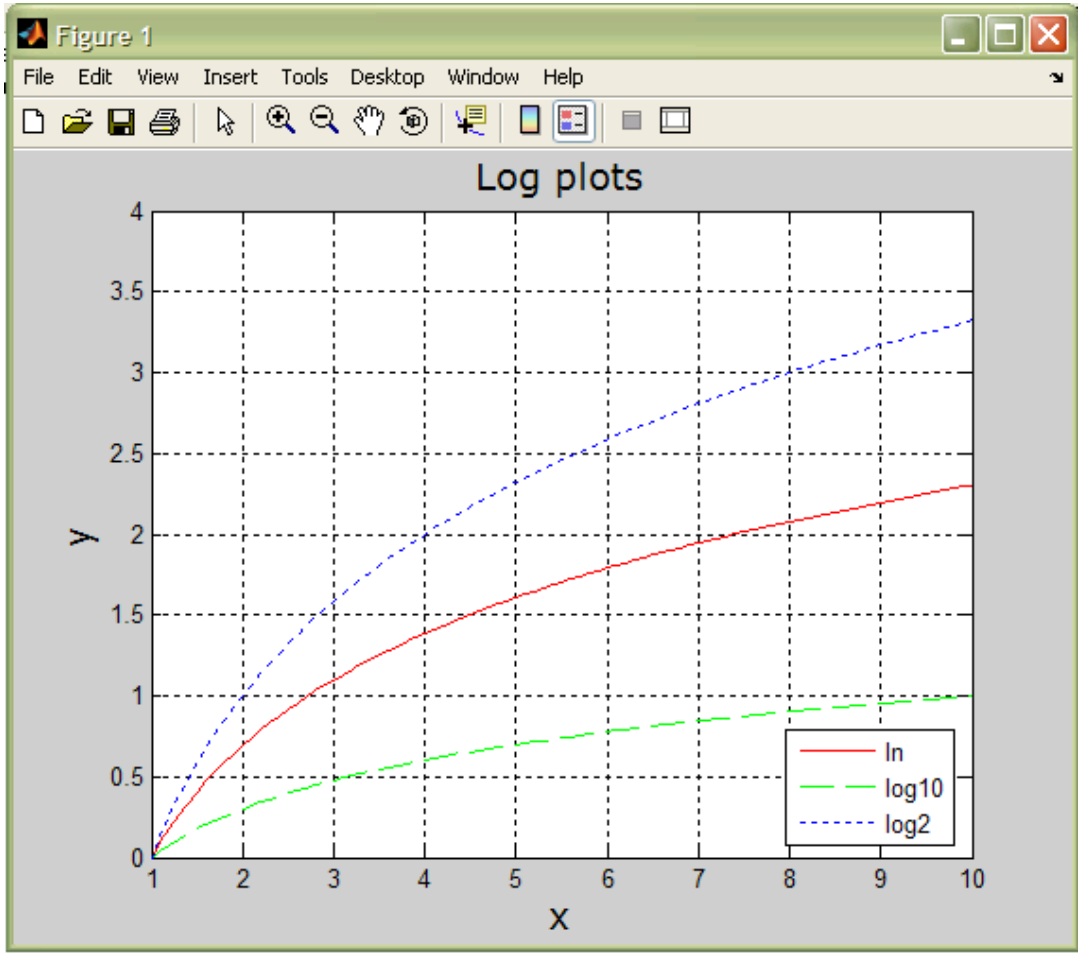


Figure 1
File Edit View Insert Tools Desktop Window Help
Log plots
y
x
ln
log10
log2

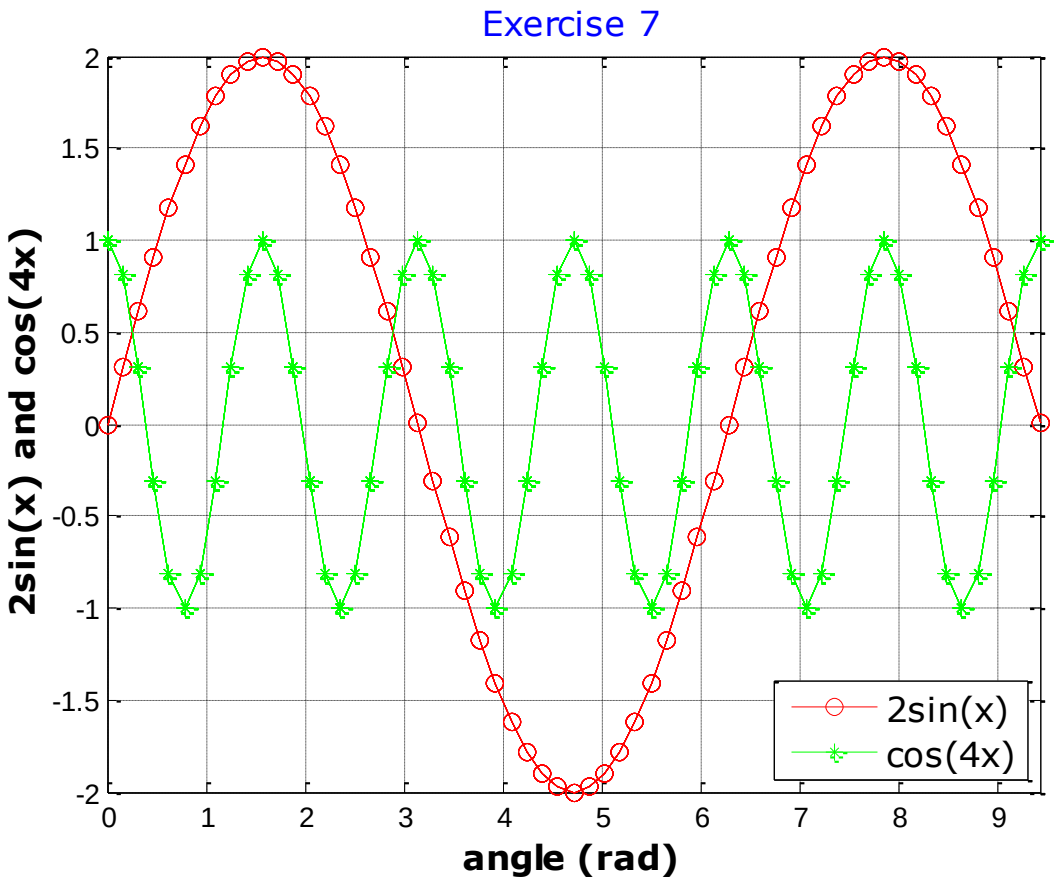
Exercise

In your plot from the previous exercise, use the axis command to plot only from π to 3π. Add grid to your plot.

Exercise

```
% Exercise
x = 0 : pi/20 : 3*pi;
y1 = 2*sin(x);
y2 = cos(4*x);
plot(x, y1, 'o-r');
hold on
plot(x, y2, '*-g');
axis([0 3*pi -2 2]);
grid on
xlabel('\bf\fontname{Verdana}\fontsize{12}angle (rad)');
ylabel('\bf\fontname{Verdana}\fontsize{12}2sin(x) and
    cos(4x)');
title('Exercise 7', 'FontName', 'Verdana', 'Color', 'b', 'FontSize', 12)
legend('\fontname{Verdana}\fontsize{12}2sin(x)',
    '\fontname{Verdana}\fontsize{12}cos(4x)', 'Location',
    'SouthEast');
```

Exercise


Exercise 7
2sin(x) and cos(4x)
angle (rad)
2sin(x)
cos(4x)
2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
0
1
2
3
4
5
6
7
8
9

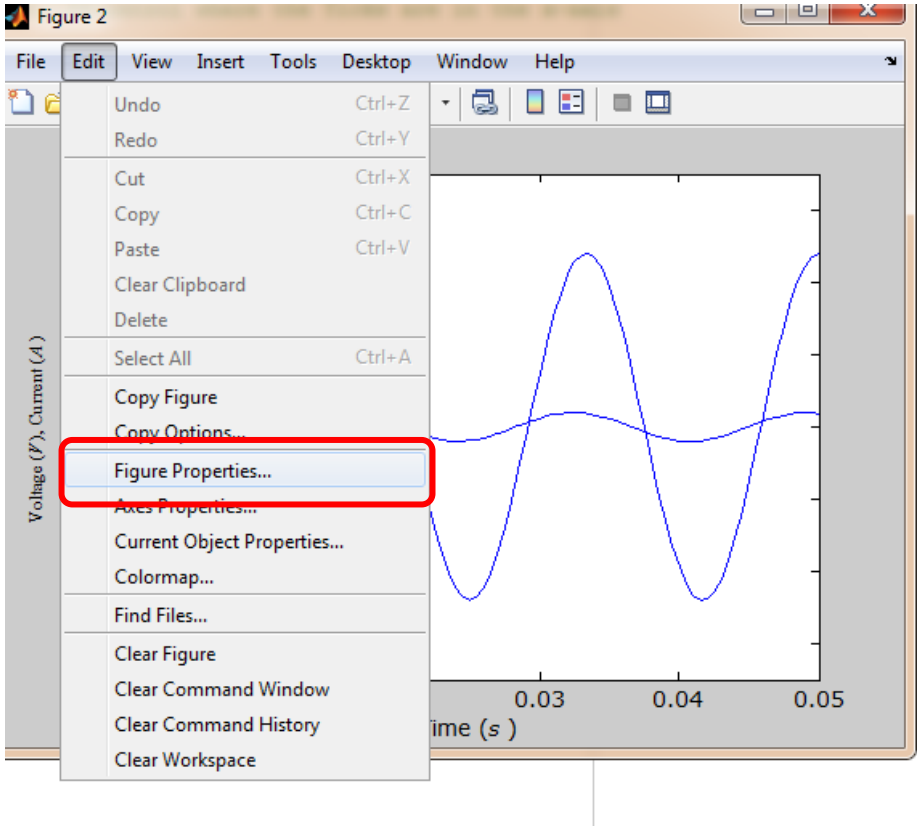
Advanced Editing

- Matlab provides more control than the options discussed
- Two options
  - GUI
  - set and get commands

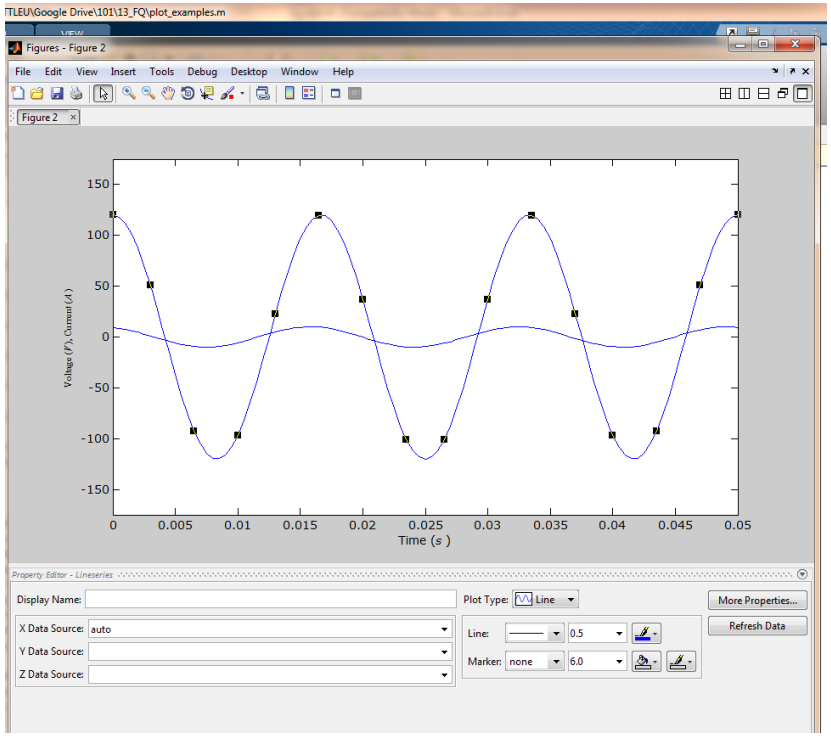
GUI

- The lazy person's way of editing plots (not recommended)

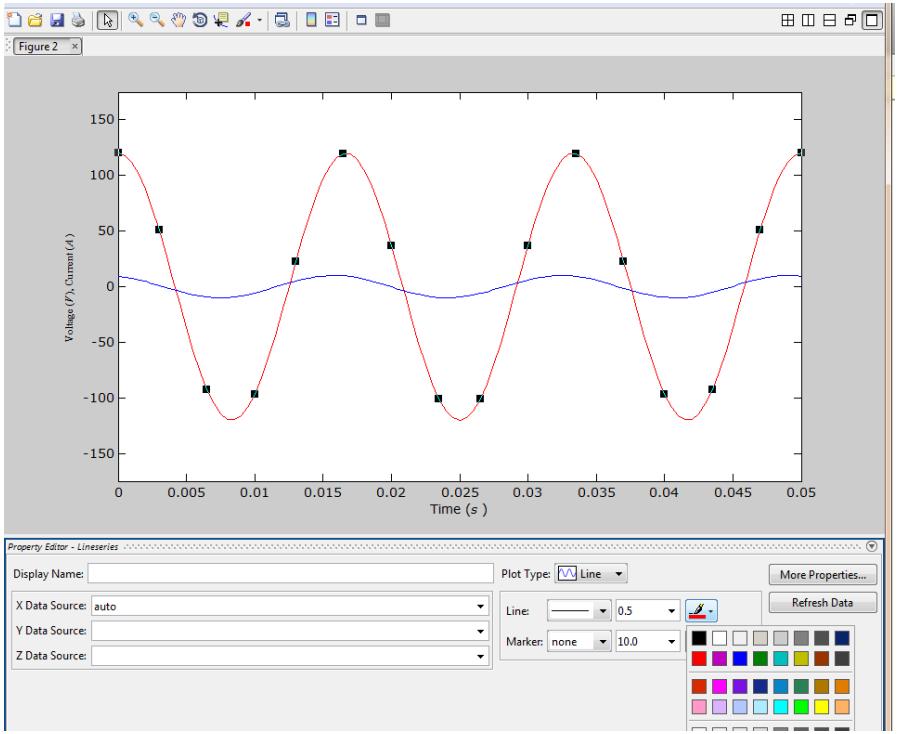
Create a plot.
Edit "Figure Properties"



GUI



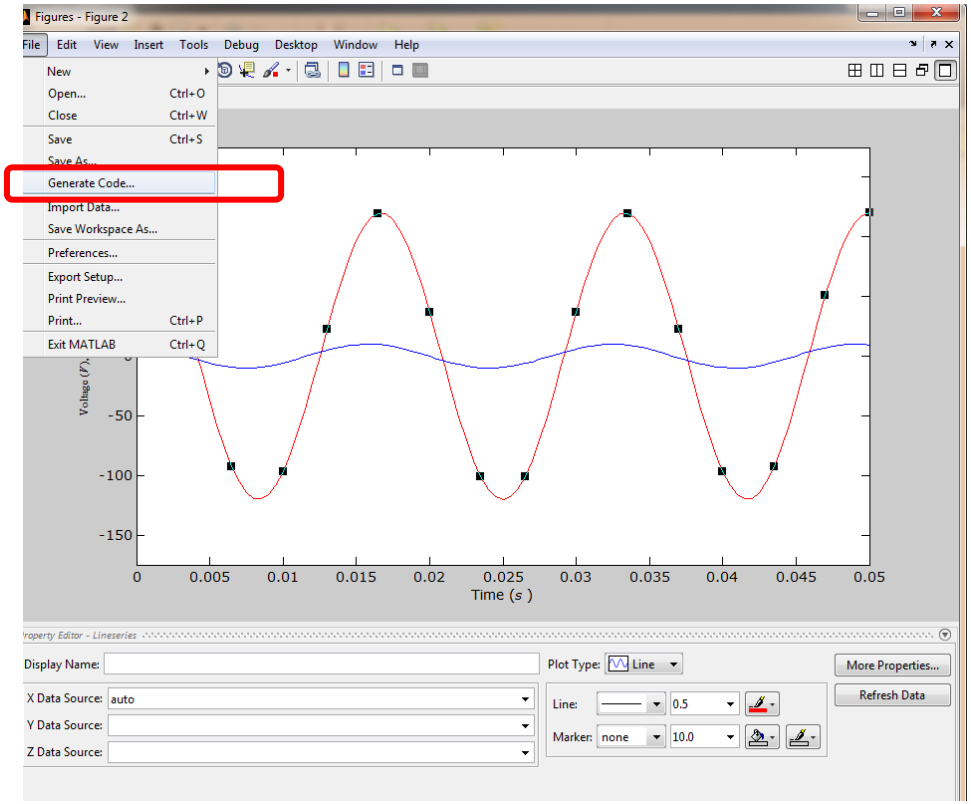
Click on plot, label, axis, etc. to edit



Example: change the color to red

GUI

- To see how to re-create your plot using code



Click on "Generate Code"

```
function createfigure(X1, YMatrix1)
%CREATEFIGURE(X1,YMATRIX1)
%  X1:  vector of x data
%  YMATRIX1:  matrix of y data

%  Auto-generated by MATLAB on 28-Oct-2013 12:10:16

% Create figure
figure1 = figure;

% Create axes
axes1 = axes('Parent',figure1,'YTick',[-200 -150 -100 -50 0 50 100 150 200],...
    'FontName','verdana');
%% Uncomment the following line to preserve the Y-limits of the axes
% ylim(axes1,[-175 175]);
box(axes1,'on');
hold(axes1,'all');

% Create multiple lines using matrix input to plot
plot1 = plot(X1,YMatrix1);
set(plot1(1),'MarkerSize',10,'Color',[1 0 0]);

% Create xlabel
xlabel('Time (\it{s} \rm{)}','FontName','verdana');

% Create ylabel
ylabel('Voltage (\it{V} \rm{)}, Current (\it{A} \rm{)}','FontSize',8,...
    'FontName','times');
```

Set and Get

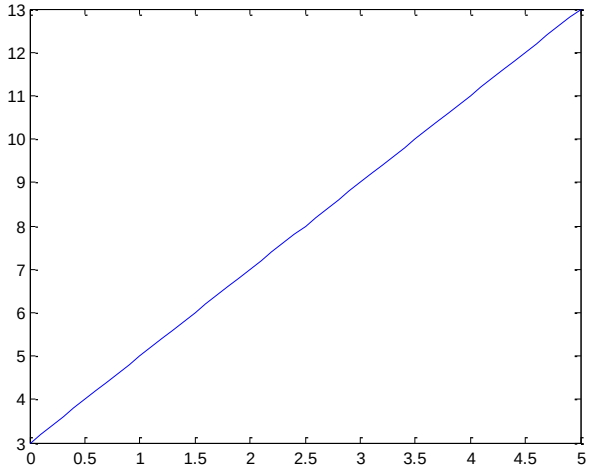
- GUI is easy, but requires human intervention
- Better approach is to specify your plots
  - Use commands previously discussed
  - Use set and get

Set and Get

- Set: lets you set figure, plot, axis and label properties
- Get: returns value of specified object
- Objects include: figure, plot, axis and label, etc.
- Objects have unique handles
  - gca: get current axes handle
  - gcf: get current figure handle
  - gco: get current object handle

Example

```
x=0:.1:5;
y=2*x+3;%%a simple affine function
figure1 = figure%%figure1 holds the figure's handle
plot(x,y)
get(figure1)%%display the current properties
```




```
Alphamap = [ (1 by 64) double array]
CloseRequestFcn = closereq
Color = [0.8 0.8 0.8]
Colormap = [ (64 by 3) double array]
CurrentAxes = [175.026]
CurrentCharacter =
CurrentObject = []
CurrentPoint = [0 0]
DockControls = on
FileName =
IntegerHandle = on
InvertHardcopy = on
KeyPressFcn =
KeyReleaseFcn =
MenuBar = figure
Name =
NextPlot = add
NumberTitle = on
PaperUnits = inches
PaperOrientation = portrait
PaperPosition = [0.25 2.5 8 6]
PaperPositionMode = manual
```

Truncated list of what Matlab outputs

Example

truncated list of properties

```
Alphamap = [ (1 by 64) double array]
CloseRequestFcn = closereq
Color = [0.8 0.8 0.8]
Colormap = [ (64 by 3) double array]
CurrentAxes = [175.026]
CurrentCharacter =
CurrentObject = []
CurrentPoint = [0 0]
DockControls = on
FileName =
IntegerHandle = on
InvertHardcopy = on
KeyPressFcn =
KeyReleaseFcn =
MenuBar = figure
Name =
NextPlot = add
NumberTitle = on
PaperUnits = inches
PaperOrientation = portrait
PaperPosition = [0.25 2.5 8 6]
PaperPositionMode = manual
```

We can change any of these properties.
Let's see what happens if we change 'color' to [.8 .1 .1]

Search help for "figure properties" for what each property controls, and what the defaults are

Example

```
x=0:.1:5;
y=2*x+3;%%a simple affine function
figure1 = figure%%figure1 holds the
figure's handle
plot(x,y)
get(figure1)%%display the current
properties
set(figure1,'color',[.8 .1 .1])
```

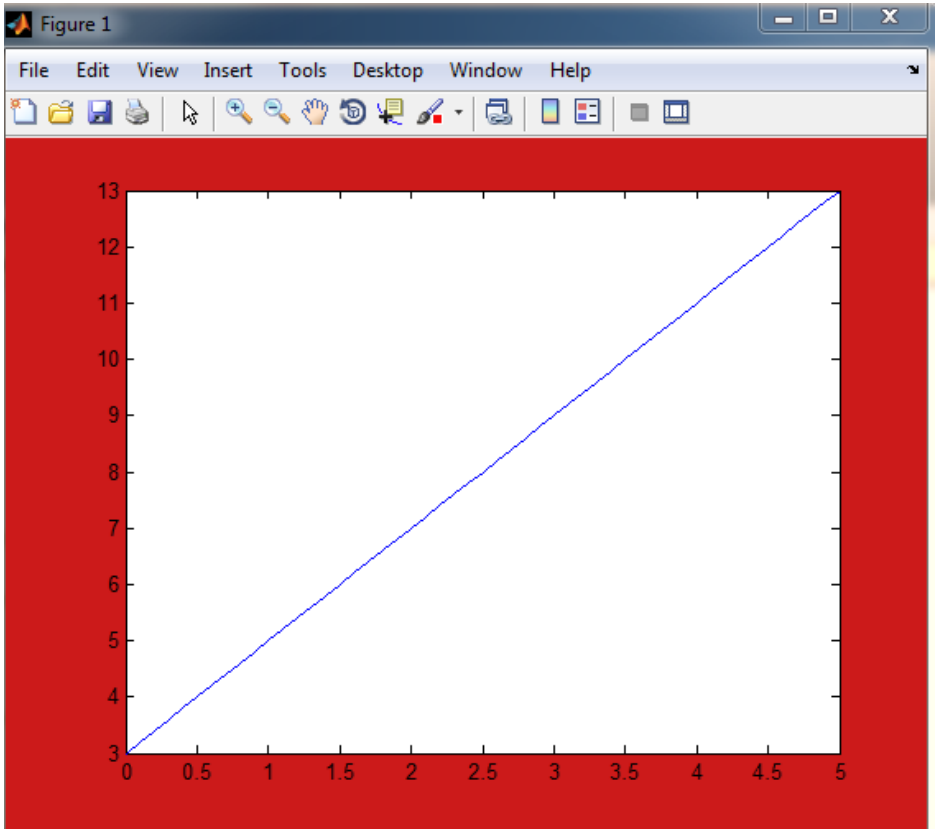
Use the "set" command

property name

property values

Example

The figure color changed!



GCF

- The same result can be obtained using gcf

```
x=0:.1:5;
y=2*x+3;%%a simple affine function
figure
plot(x,y)
set(gcf,'color',[.8 .1 .1])
```

gcf gets the current figure's handle

Advanced Editing

- A similar approach can be used for lines, axes and other objects

```
x=0:.1:5;
y=2*x+3;%%a simple affine function
figure1 = figure%%figure1 holds the figure's handle
axes1= axes('parent',figure1)%%axes1 holds the axes' handle
trace1 = plot(x,y)%%trace1 holds the plot's handle
get(axes1)%%displays the current axes settings
set(axes1,'color',[0 1 0])%%changes axes color
get(trace1)%%displays the current plot settings
set(trace1,'color',[1 .5 .5],'linestyle',':','marker','+','markersize',10)%%changes plot properties
```

Advanced Editing

truncated axes properties

```
ActivePositionProperty = outerposition
ALim = [0 1]
ALimMode = auto
AmbientLightColor = [1 1 1]
Box = on
CameraPosition = [2.5 8 17.3205]
CameraPositionMode = auto
CameraTarget = [2.5 8 0]
CameraTargetMode = auto
CameraUpVector = [0 1 0]
CameraUpVectorMode = auto
CameraViewAngle = [6.60861]
CameraViewAngleMode = auto
CLim = [0 1]
CLimMode = auto
Color = [1 1 1]
CurrentPoint = [ (2 by 3) double array]
ColorOrder = [ (7 by 3) double array]
DataAspectRatio = [2.5 5 1]
DataAspectRatioMode = auto
DrawMode = normal
FontAngle = normal
FontName = Helvetica
FontSize = [10]
FontUnits = points
FontWeight = normal
GridLineStyle = :
Layer = bottom
LineStyleOrder = -
```

truncated line properties

```
          DisplayName: ''
           Annotation: [1x1 hg.Annotation]
                Color: [0 0 1]
            LineStyle: '-'
            LineWidth: 0.5000
               Marker: 'none'
           MarkerSize: 6
      MarkerEdgeColor: 'auto'
      MarkerFaceColor: 'none'
                XData: [1x51 double]
                YData: [1x51 double]
                ZData: [1x0 double]
       BeingDeleted: 'off'
     ButtonDownFcn: []
           Children: [0x1 double]
           Clipping: 'on'
          CreateFcn: []
          DeleteFcn: []
          BusyAction: 'queue'
    HandleVisibility: 'on'
             HitTest: 'on'
       Interruptible: 'on'
            Selected: 'off'
  SelectionHighlight: 'on'
```

Advanced Editing