Plotting Data

Key Terms:

plot
LineStyle
Color
MarkerType
PropertyNames
hold on, hold off
line
xlabel, ylabel, zlabel
title
text
legend
axis
xlim, ylim, zlim
grid
fplot
subplot
ceil
floor
fix
round

Basic Plots

In MATLAB, a plot refers to a two dimensional graph of points. Each point must be specified with an x and y coordinate, and neighboring points are connected with straight lines. Plots are often more useful as output for conveying meaning and trends to a person than showing large lists of numbers. Sometimes, plots can be used to estimate solutions to problems. Let's start with a basic plot, and the rest of the lecture will show how to add more features and make them more presentable.

The syntax of the plot function is

plot(array of x coordinates, array of y coordinates);

Enter the following commands into the command window, noting what the Figure window displays after each line.

```
>>plot([1 2 3 4 5], [2 4 9 4 6]);

>>plot([2 4 9 4 6]);

>>plot([1:1:10],linspace(1,27,10).^3);
```

Things to note:

- Obviously, the lengths of both arrays must be the same
- If only one array is supplied, it is interpreted as the y array, and the x coordinates are integers beginning at 1.
- The default is a plain, blue line on a white background
- The x and y limits are automatically selected so that all the data can be displayed.
- Nothing is labeled.

Line Specifiers

Plot lines can be modified in many ways. The easiest way is with a line specifier, which can alter the color, type of line, and the marker type, although MATLAB has only a few options to select from. The line specifier is an additional, optional input to the plot function, and the option for each of the three parameters is shown below.

plot(array of x coordinates, array of y coordinates, line specifier);

LineStyle:	LineColor:
- solid	r red
-- dashed	g green
: dotted	b blue
-. dash-dot	c cyan
	m magenta
MarkerType	y yellow
+ o * . x ^ v < > s d p h	k black
	w white

Enter the following commands into the command window, noting what the Figure window displays after each line.

>>plot([1 2 3 4 5], [2 4 9 4 6], '—rd');
>>plot([1 2 3 4 5], [2 4 9 4 6], 's:m');
>>plot([1 2 3 4 5], [2 4 9 4 6], 'g-.');

Notes:

- The order of the three parameters does not matter.

Additional Line Properties

The three line properties already discussed as well as several others can be specified more explicitly. Instead of the abbreviated symbols on the previous page, any property can be modified by including the property name and the desired value in the plot function. The format for this plot function is

plot(array of x coordinates, array of y coordinates, ‘property’,’value’, ‘property’, ‘value’,…);

The list of valid properties is

- LineStyle
- Color
- Marker
- LineWidth
- MarkerSize
- MarkerEdgeColor
- MarkerFaceColor

Notes:

- Valid values for each property can be found by accessing MATLAB’s help.
- Each property and its corresponding value must be consecutive, and both are in single quotes.
- You can specify any number of parameters and in any order.
- Watch capitalization. The properties and values must be capitalized as above or as in the help section.

Examples

The two lines below show the same plot in line specifier form and in line property form.

>>plot([1 2 3 4 5], [2 4 9 4 6], '—rd'); % line specifiers

>>plot([1 2 3 4 5], [2 4 9 4 6],'LineStyle','-','Color','Red', …

'Marker','Diamond'); % line properties

What are the line property forms for the other two examples on page 3?

Adjusting Limits

MATLAB automatically scales the figure window to the data. You may want to control this manually, since having data points at the very edge of a plot is sometimes hard to read. Use the xlim or ylim commands to set the x range or y range. These commands do not return output. The format is shown below. You may also want to look at help on the axis command.

xlim([low_limit high_limit]);

ylim([low_limit high_limit]);

Multiple Lines per Plot

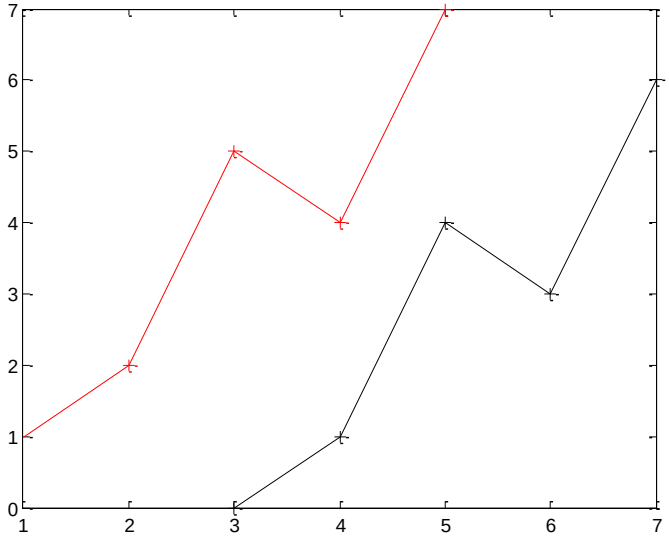
There are several ways to show multiple lines on the same plot. If you only want to distinguish the lines using the three parameters in line specifiers, you can add multiple triples in the plot command. Each triple is an array of x coordinates, an array of y coordinates, and a line specifier. Note that you MUST include the x coordinates when supplying multiple lines.

Example:

```
>>plot([1:1:5],[1 2 5 4 7], '-r+',[3:1:7],[0 1 4 3 6], '-k+');
```

The second method is with the hold command. The plot function normally erases any existing figure and plots a new one. If you enter the “hold on” command, following plot commands will add to the current figure, and all line parameters can be used. Use a “hold off” command to stop adding lines to the figure.

```
>>plot([1:1:5],[1 2 5 4 7], '-r+');
>>hold on;
>>plot([3:1:7],[ 0 1 4 3 6], '-k+');
>>hold off;
```



Adding Details to a Plot

Anyone submitting a plot that doesn't have each axis labeled with units, a useful title, a legend if appropriate, etc. should be…well, something that's inappropriate to be spoken of in polite company. Therefore, you need to know how to add all these labels to a figure. The list of text objects below is fairly self-explanatory.

xlabel('text', 'property', 'value',…)
ylabel('text', 'property', 'value',…)
zlabel('text', 'property', 'value',…)
title('text', 'property', 'value',…)
text(x,y,'text', 'property', 'value',…)
legend('name', 'name',…)

Notes:

Text – x and y are coordinates on the plot that determines where the text goes.

Legend – the 'Location' parameter is NSEW

Each text object can be specified with property-pairs just like the lines. The valid properties are shown below, and you can access MATLAB help for lists of valid values

Rotation
FontAngle
FontName
FontSize
FontWeight
Color
BackgroundColor
EdgeColor
LineWidth

What are the valid property values for FontAngle?

What are the valid property values for FontWeight?

Example of a Labeled Plot

Let's create a well labeled graph. We'll graph a simple sine wave so we can focus on the labeling part (sense a trick coming?) First, we need to generate the data for the graph.

```
1: theta = linspace(0,6*pi,7);   % pi is a built-in constant

2: plot(theta, 2*sin(theta), 'LineStyle', ':', 'Color', 'red', 'LineWidth', 2);

3: ylim([-2.5 2.5])
```

It seems we have a problem with the plot. Here's the issue.

Sampling rate, increase linspace to 13, then 25.

Now on to the labels.

```
4: title('Sine Function','FontSize',18,'FontName','Times New Roman')

5: ylabel('Magnitude', 'FontSize', 12)

6: xlabel('Angle (radians)', 'FontSize', 12)
```

Plotting a Function

When you have sets of x and y points, you are limited to the plot command. If you're trying to graph a continuous function, MATLAB has a better command. The fplot command takes a function of one variable and plots it over a specified range. Unlike the plot command, you do not need to supply x coordinates. MATLAB uses the function to calculate the y coordinate for each pixel on the screen with an x coordinate, so the graph will always be smooth and have the "best" number of points. Like the plot command, you can add line properties to the command. The format is shown below.

fplot('function', [low_limit high_limit], 'property', 'value',…)

Add to the previous example by adding a cosine and a legend.

7: hold on

8: fplot('2*cos(x)', [0 6*pi],'Color', 'blue', 'LineWidth, 2);

9: legend('sine', 'cosine', 'Location','NorthEast');

10: hold off

Question: How does MATLAB know that the red line should have the sine label and cosine should have the blue label?

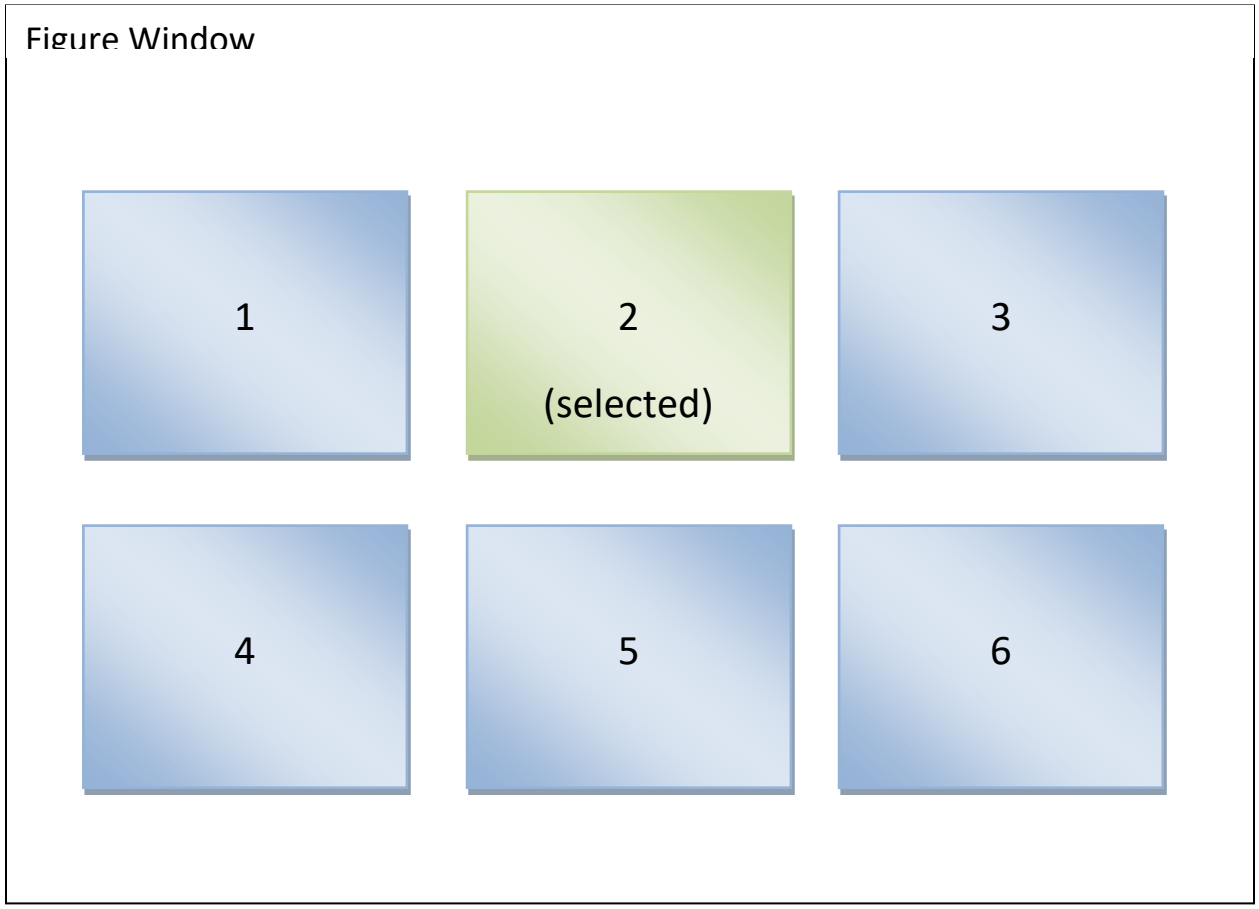
MATLAB assigns legend entries to plots in the order that plots were sent to the figure window. You first sent the sine with the plot command then sent the cosine with the fplot command.

Multiple Plots per Figure

Instead of writing multiple plots into the same graph, you can have separate graphs. The subplot command tells MATLAB to divide the figure window into an array of graphs, and it tells it which graph to select. After the subplot command, all plotting commands will be routed to the selected graph. You switch to a different graph with another subplot command. Graphs are numbered across and then down.

subplot(rows_of_graphs, columns_of_graphs, selected_graph)

>> subplot(2,3,2); % 2 rows, 3 graphs per row, select graph 2



Example subplots_1.m

The script below creates the sine and cosine graphs on two separate plots in the same figure window.

```
subplot(2,1,1);
fplot('2*sin(x)', [0 6*pi],'Color', 'red');
title('Sine Function - Lecture 5'…
    ,'FontSize',18,'Fontname','Verdana');
xlabel('Angle (radians)','FontSize',12)
ylabel('Magnitude','FontSize',12)
ylim([-2.5 2.5])

subplot(2,1,2);
fplot('2*cos(x)', [0 6*pi],'Color', 'blue');
title('Cosine Function - Lecture 5'…
    ,'FontSize',18,'Fontname','Verdana')
xlabel('Angle (radians)','FontSize',12)
ylabel('Magnitude','FontSize',12)
ylim([-2.5 2.5])
```