Markers

Sometimes you want to plot individual points as well as curves. For example, suppose you have collected the data points (1,2), (3,3), (4,3), (5,5), (7,8) and you want to plot both the data points and the line that best fits the data in the least squares sense. Try the following commands.

```matlab
xdata=[1 3 4 5 7];
ydata=[2 3 3 5 8];
p1=polyfit(xdata, ydata,1); %Finds the polynomial of degree 1 that best fits the data
xline=linspace(0,8,100);
plot(xdata,ydata, '*', xline, polyval(p1, xline), '-b') %polyval(p1, xline) evaluates the polynomial p1 a the points in the array xline
```

Try changing the * to + or x in the plot command to see what happens. You can see the complete list of marker types by issuing the command `>>help plot`.

Formatting

As you have seen, you can control the line type and color of a graph using the `plot` command. You can also control other details of a figure. The easiest way to make changes to a figure is to use the figure editor. Here is an example. First generate a graph to work with using

```matlab
x=linspace(0, 2*pi);
y=sin(x);
z=cos(x);
plot(x,y,x,z);
```

Suppose you decide you want the graph of the sine function to be green, dashed curve with line width 2 points instead of the default 0.5 point. In the figure window, click `edit`, then click on `figure properties`. Right click on the graph of the sine function, then do the following.

1. Click `color`, click on the color you want the graph to have, then click `OK`. 
2. Right click on the graph again, click on *line style*, then click on *dash*.

3. Right click on the graph again, click on *line width*, then click on *2.0*.

### 3 Error bars

Here is an example showing how to add error bars to data points. This example uses the data from the section of Markers.

```matlab
xddata=[1 3 4 5 7];
ydata=[2 3 5 8];
yerr=[0.5 0.8 0.2 0.4 0.6];
p1=polyfit(xdata, ydata,1); %Finds the polynomial of degree 1 that best fits the data
xline=linspace(0,8,100);
plot(xdata,ydata, '*', xline, polyval(p1, xline), '-b')
hold % Keeps the current figure open
errorbar(xdata, ydata, yerr)
```

The `errorbar` command generate line segments joining the data points. If you do not want those line segments, try

```matlab
errorbar(xdata, ydata, yerr, 'LineStyle', 'none')
```

Note that the values in the vector `yerr` give the half-width of the error bars. For example, the first error bar runs from 0.5 units below the point (1,2) to 0.5 units above the point. If your error bars are not symmetric, you need to specify the lower and upper range of each error bar, something like this.

```matlab
earlo=[0.2 0.4 0.3 0.1 0.2];
errhi=[0.1 0.2 0.5 0.3 0.4];
plot(xdata, ydata, '*', xline, polyval(p1, xline), '-b')
hold
errorbar(xdata, ydata, errlo, errhi, 'LineStyle', 'none')
```

### 4 Bar graphs, Pie charts, and Histograms

Here are some examples illustrating how to create bar graphs, pie charts, and histograms.

#### 4.1 Bar graphs

Try these commands
year=[2010:2014];
sales=randi([5, 15], [1,5]); % This command generates a 1 by 5 array of randomly
% chosen integers between 5 and 15
bar(year, sales)
xlabel('Year')
ylabel('Millions of Dollars')
title('ACME Corporation Sales')
barh(year, sales) % Generates a horizontal bar graph

4.2 Pie Charts

Suppose the following table gives the grade distribution in Calculus I.

<table>
<thead>
<tr>
<th>Grade</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Students</td>
<td>11</td>
<td>18</td>
<td>26</td>
<td>9</td>
<td>5</td>
</tr>
</tbody>
</table>

Try these commands

count=[11 18 26 9 5];
pie(count);

Use the figure editor to add the letters A, B, C, D, and F in the appropriate regions of the chart. In the figure window, click Edit, then click on Figure Properties. Click on Insert on the toolbar, then click on TextBox. Click in the largest piece of the pie, then type the letter C. Add text boxes for other letters.
If you don’t want to see the box around the letter, right click on the text box, click on Line Style, then click on none.

5 Histrograms

A histogram takes a set of numbers, breaks the set into a number of intervals (or bins), and shows how many elements from the data set fall into each bin.
Try these commands.

x=randi([0, 100], [1, 50]);
histogram(x);

You can specify the number of bins

histogram(x, 10)
If you want to see the histogram properties (bin edges, bin width, number of points in each bin, etc.), try

\[ h = \text{histogram}(x, 10) \]

### 6 The fplot command

The fplot command offers an alternative to the plot command for graphing functions. Try these commands

\[
\text{fplot}(\sin(x), [-\pi, \pi]) \quad \text{Plots the sine over the interval } [-\pi, \pi] \\
\text{fplot}(\sin(x), [-\pi, \pi], '--k')
\]

Note that you do not need to calculate any x and y values when using fplot. You can also plot curves defined parametrically. The following command plots the curve given by

\[
x = \cos(t), y = \sin(t), 0 \leq t \leq 2\pi
\]

\[
\text{fplot}(\cos(t), \sin(t), [0, 2\pi])
\]
Practice Problems

1. Plot the function

\[ f(x) = x^2, \quad -3 \leq x \leq 3 \]

and mark the points (-2,4), (1,1), and (2,4)

2. Add symmetric error bars to the graph you just generated using error bars of half-width 0.5, 0.4, and 0.8 for three points.

3. Generate a vector of 150 randomly chosen integers between 0 and 50, then generate a histogram with 10 bins using this vector.

4. Use the \texttt{fplot} command to graph the function

\[ f(x) = e^{\sin(x)}, \quad -2\pi \leq x \leq 2\pi. \]