A. 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')
```

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

```matlab
>> help plot
```

B. 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 issue these commands to generate a graph to work with.

```matlab
>> x = linspace(0, 2*pi, 100);
>> y = sin(x);
>> z = cos(x);
>> plot(x, y, x, z)
```

Suppose you decide you want the graph of the sine function to be a green, dashed curve with line width 2 points instead of the default 0.5 point. In the Figure window, click on *Edit*, then click on *Figure Properties*. Right click on the graph of the sine function, then do the following.

1. Click on *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 one more time, click on *Line Width*, then click on *2.0*.

You can also implement these changes using commands in the command window. To get more information, issue the command

```matlab
>> help plot
```
C. Error Bars
Here is an example showing how to add error bars to data points. This example uses the data from the section on Markers.

```matlab
>> xdata = [1 3 4 5 7];
>> ydata = [2 3 3 5 8];
>> yerr = [0.5 0.8 0.2 0.4 0.6];
>> p1 = polyfit(xdata, ydata, 1);
>> 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 generates line segments joining the data points. If you do not want those line segments, try this command:

```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
>> errlo = [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')
```

D. Polar Plots
To graph the polar equation $r = f(\theta)$ you can use the `polar` command. For example, to graph the equation $r = \cos(2\theta)$ for $0 \leq \theta \leq 2\pi$ try these commands.

```matlab
>> theta = linspace(0, 2*pi, 100);
>> r = cos(2*theta);
>> polar(theta, r)
```

E. Bar Graphs, Pie Charts, and Histograms
Here are some examples illustrating how to create bar graphs, pie charts, and histograms.

i. Bar Graphs
Try these commands.

```matlab
>> year = [2010:2014];
>> sales = randi([5,15],[1,5]);
  %This command generates a 1x5 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
ii. Pie Charts
Suppose the following table gives the grade distribution in Methods of Meteorology 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.

```matlab
>> 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 on 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 the 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.

iii. Histograms
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.

```matlab
>> x = randi([0, 100], [1, 50]);
% This command generates a 1x50 array of randomly chosen integers between 0 and 100
>> histogram(x)
```

You can specify the number of bins. For example, try

```matlab
>> histogram(x, 10)
```

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

```matlab
>> h = histogram(x, 10)
```

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

```matlab
>> fplot('sin(x)', [-pi, pi])
>> fplot('sin(x)', [-pi, pi, -2, 2])
>> fplot('sin(x)', [-pi, pi, -2, 2], ':r')
```

What effect did changing `[-pi, pi]` to `[-pi, pi, -2, 2]` have on the plot?
Practice Problems.

1. Plot the function $f(x) = x^2$ for $-3 \leq x \leq 3$ and mark the points $(-2, 4)$, $(1,1)$, and $(2,4)$ with circles.

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 the 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. Draw a polar plot of $f = 1 + 2 \sin(\theta)$ for $0 \leq \theta \leq 2\pi$.

5. Use the `fplot` command to graph the function $f(x) = e^{\sin(x)}$ for $-2\pi \leq x \leq 2\pi$. 