Learning Objectives

1. Create scatter plots in MATLAB with good graphing conventions (e.g., legend, linestyles, title, multiple plots on same graph)

Topics

Students will read Chapter 5 of the MATLAB book before coming to class. This preparation material is provided to supplement this reading.

Students will learn a basic understanding of creating and formatting two-dimensional (2D) plots in MATLAB. This material contains the following:

1. Introduction to 2D Plots
2. The plot() Command
3. Multiple Graphs in the Same Plot
4. Formatting Plots

1. Introduction to 2D Plots

MATLAB has many functions and commands that can be used to create various types of plots. In this class, we will only create 2D plots. This particular lesson only focuses on x-y plots.
Example of a 2D Plot in Engineering:

![Light Intensity as a Function of Distance](image)

2. The plot() Command

The basic 2-D plot command is:

```
plot(x, y)
```

- where \( x \) and \( y \) are **any vector names**. Both vectors must have the same number of elements.

The plot() command creates a single curve with the \( x \) values on the abscissa (horizontal axis) and the \( y \) values on the ordinate (vertical axis).

The curve is made from segments of lines that connect the points that are defined by the \( x \) and \( y \) coordinates of the elements in the two vectors.
**Example of a Plot from Given Data**

Using the given data in two vectors, x and y:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>5</th>
<th>7</th>
<th>7.5</th>
<th>8</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>y</td>
<td>2</td>
<td>6.5</td>
<td>7</td>
<td>7</td>
<td>5.5</td>
<td>4</td>
<td>6</td>
<td>8</td>
</tr>
</tbody>
</table>

Vectors of equal length can be created, and then plotted, using:

```matlab
>> x = [1 2 3 5 7 7.5 8 10];
>> y = [2 6.5 7 7 5.5 4 6 8];
>> plot(x,y)
```

Once the `plot()` command is executed, the Figure Window opens with the following plot.
Line Specifiers in the plot() Command
Line specifiers are used in the plot() command to specify:

- The style of the line
- The color of the line
- The type of the markers (if any)

The general format of the plot() command is the following:

\[
\text{plot}(x, y, \text{'line specifiers'})
\]

General reminders about specifiers:

- The specifiers are entered as a string
- Within the string, the specifiers can be in any order
- The specifiers are optional, which means that 0, 1, 2 or all 3 can be included in a command

<table>
<thead>
<tr>
<th>Line Style</th>
<th>Specifier</th>
<th>Line Color</th>
<th>Specifier</th>
<th>Marker Type</th>
<th>Specifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Solid</td>
<td>-</td>
<td>Red</td>
<td>r</td>
<td>X-mark</td>
<td>x</td>
</tr>
<tr>
<td>Dotted</td>
<td>:</td>
<td>Green</td>
<td>g</td>
<td>Circle</td>
<td>o</td>
</tr>
<tr>
<td>Dashed</td>
<td>--</td>
<td>Blue</td>
<td>b</td>
<td>Asterisk</td>
<td>*</td>
</tr>
<tr>
<td>Dash-dot</td>
<td>-.</td>
<td>Black</td>
<td>k</td>
<td>Point</td>
<td>.</td>
</tr>
</tbody>
</table>

Information about these options (and others) can be accessed in the command window by typing:

\[
>> \text{help plot}
\]
Example of a Plot using Specifiers
Given the data in the table, the equally sized Year and Sales vectors can be created. These vectors can then be plotted with the proper specifiers to make it dashed and red.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Sales ($M)</td>
<td>127</td>
<td>130</td>
<td>136</td>
<td>145</td>
<td>158</td>
<td>178</td>
<td>211</td>
</tr>
</tbody>
</table>

```matlab
>> year = [1988: 1: 1994];
>> sales = [127, 130, 136, 145, 158, 178, 211];
>> plot(year, sales, '--r*')
```

Line Specifiers: dashed red line and asterisk markers.
Creating a Plot of a Function
Consider the following function $y$ as defined by $x$:

$$ y = 3.5^{-0.5x} \cos(6x) \quad \text{for} \quad -2 \leq x \leq 4 $$

A script file for plotting the function, and the created plot, is:

```matlab
% A script file that creates a plot of
% the function: 3.5^(-0.5x)*cos(6x)
x = [-2: 0.01: 4];
y = 3.5 .^ (-0.5*x) .* cos(6*x);
plot(x,y)
```

Likewise, using large (too large) spacing, such as $x = [-2: 0.3: 4]$; creates an un-smooth plot:
3. Multiple Graphs in the Same Plot
There are two options for plotting multiple graphs in the same plot:

- Use the `plot()` command
- Use the `hold on` and `hold off` commands

**Using the plot() Command for Multiple Graphs**
Using the `plot()` command requires only to list additional plots within the same command. The basic command could look like this:

```
plot(x, y, u, v, t, h)
```

This creates three graphs in the same plot:

- `y` versus `x`, `v` versus `u`, and `h` versus `t`.

- By default, MATLAB makes the curves in different colors.
- The curves can have a specific style by adding specifiers after each pair, for example:

```
plot(x, y, '-b', u, v, '--r', t, h, 'g:')
```
For example, plot the function:

$$y = 3x^3 - 26x + 10$$

and its first and second derivatives, for \(-2 \leq x \leq 4\) all in the same plot.

```matlab
x = [-2: 0.01: 4];
y = 3 * x .^ 3 - 26 * x + 6;
yd = 9 * x .^ 2 - 26;
ydd = 18 * x;
plot(x, y, '-b', x, yd, '--r', x, ydd, ':k')
```

Create three graphs, \(y\) vs \(x\) (solid blue line), \(yd\) vs \(x\) (dashed red line), \(ydd\) vs \(x\) (dotted black line) in the same figure.
Using the hold on and hold off Commands

Using the ‘hold on’ and ‘hold off’ commands are an alternative to using the plot command for placing multiple graphs on one plot. This method is useful when all of the information (all vectors) used for the plotting is not available at the same time.

- **hold on** - Holds the current plot and all axis properties so that subsequent plot commands add to the existing plot.

- **hold off** - Returns to the default mode where plot commands erase the previous plots and reset all axis properties before drawing new plots.

For light intensity vs. distance, the following equation and the corresponding collected data can be graphed in the same plot.

\[ y = \frac{95000}{x^2} \]

```matlab
x = [10: 0.1: 22];
y = 95000 ./ x .^ 2;
x_data = [10: 2: 22];
y_data = [950 640 460 340 250 180 140];
plot(x, y, 'r-'); % The first graph is created
hold on
plot(x_data, y_data, 'ro--'); % A second graph is added to the first
hold off
```

![Graph Image](image_url)
3. Formatting Plots
A plot can be formatted to have a required appearance. With formatting, you can:

- Add a title to the plot
- Add labels to the axes
- Change the range of the axes (as appropriate)
- Add a legend (when multiple curves)
- Add a grid (if needed for clarity)

Some common formatting commands are shown below:

- `title('string')` Adds the string as a title at the top of the plot
- `xlabel('string')` Adds the string as a label to the x-axis
- `ylabel('string')` Adds the string as a label to the y-axis
- `axis([xmin xmax ymin ymax])` Sets the minimum and maximum limits of the x-axis and y-axis
- `legend('string1', 'string2', 'string3')` Creates a legend for the various plots – must be in same order as plots were generated.

More formatting options are discussed in the textbook.
Example of Formatting a Plot

```matlab
x = [10: 0.1: 22];
y = 95000 ./ x .^ 2;
x_data = [10: 2: 22];
y_data = [950 640 460 340 250 180 140];
plot(x, y, '-', x_data, y_data, 'ro--')
xlabel('DISTANCE (cm)')
ylabel('INTENSITY (lux)')
title('Light Intensity as a Function of Distance')
axis([8 24 0 1200])
legend('Theory', 'Experiment')
```

Axes labels
Plot title
Axes limits
Legend