Chapter Five

Plotting and Graphics

Plotting is one of the most useful applications of a math package used on the computer, and MATLAB is no exception to this rule. Often, we need to visualize functions that are too hard to graph "by hand" or to plot experimental or generated data. In this chapter we will introduce the commands and techniques used in MATLAB to accomplish these kinds of tasks.

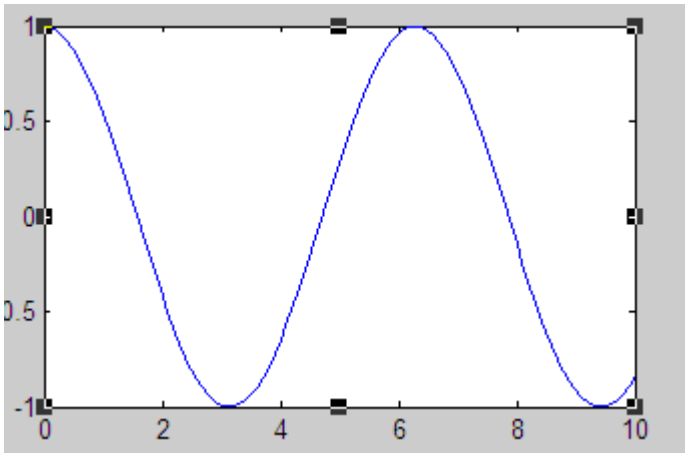
Basic 2D Plotting

Let's start with the most basic type of plot we can create, the graph of a function of one variable. Plotting a function in MATLAB involves the following three steps:

1. Define the function
2. Specify the range of values over which to plot the function
3. Call the MATLAB $plot(x, y)$ function

Example:-

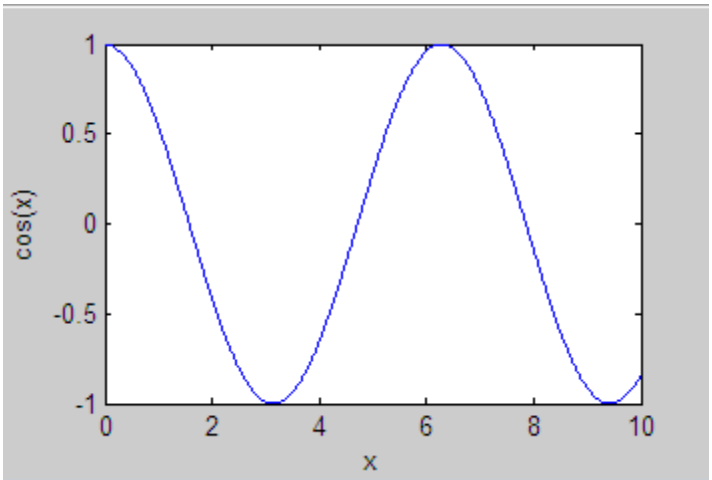
```
>> x=[0:0.1:10];
>> y=cos(x);
>> plot(x,y)
```



Using the xlabel and ylabel functions.

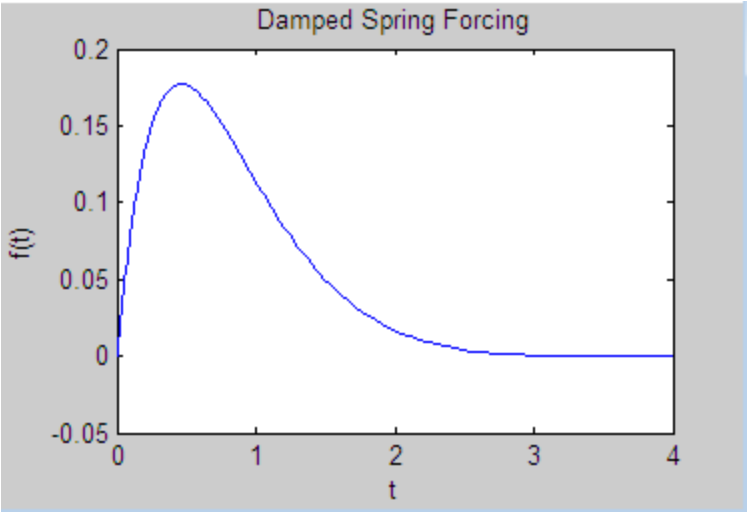
Place the *xlabel* and *ylabel* functions separated by commas on the same line as your plot command. For example,

```
>> x=[0:0.01:10];
y=cos(x);
plot(x,y),xlabel('x'),ylabel('cos(x)')
```



If we want to add labels and a title to the plot, we can follow the same procedure used with plot(x,y).

```
>> fplot('exp(-2*t)*sin(t)',[0,4]), xlabel('t'), ylabel('f(t)'),title('Damped Spring Forcing')
```



Adding the phrase grid on to your plot statement

We will plot $y = tanh(x)$ over the range $-6 \leq x \leq 6$ with a grid display. First we define our interval:

$>> \ x \ = \ [-6{:}\,0.01{:}\,6];$

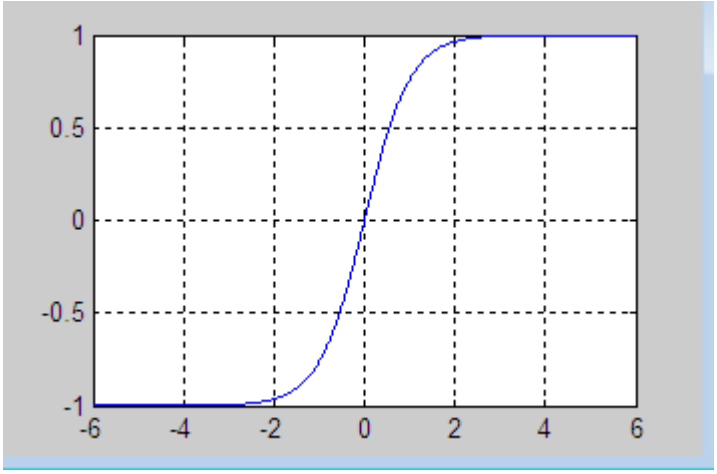
Next, we define the function:

$>> \ y \ = \ tanh(x);$

The plot command looks like this, and produces the plot shown in following Figure

$>> \ plot(x, y), grid \ on$

```
Command Window
>> x = [-6:0.01:6];
>> y = tanh(x);
>> plot(x,y),grid on
```



Showing Multiple Functions on One Plot

In many cases, it is necessary to plot more than one curve on a single graph. Let's start by showing two functions on the same graph. In this case let's plot the following two functions over $0 \leq t \leq 5$:

$f(t) = e - t$

$g(t) = e - 2t$

We will differentiate between the two curves by plotting g with a dashed line. Following the usual procedure, we first define our interval:

$>> t = [0: 0.01: 5];$

Next, we define the two functions:

$>> f = exp(-t);$

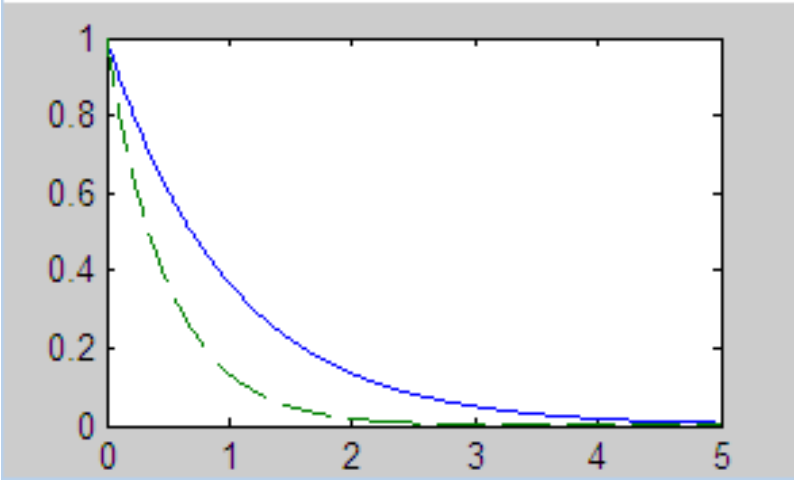
$>> g = exp(-2 * t);$

This is followed by a character string enclosed in single quotes to tell us what kind of line to use to generate the second curve. In this case we have:

$>> plot(t, f, t, g, '--')$

Command Window

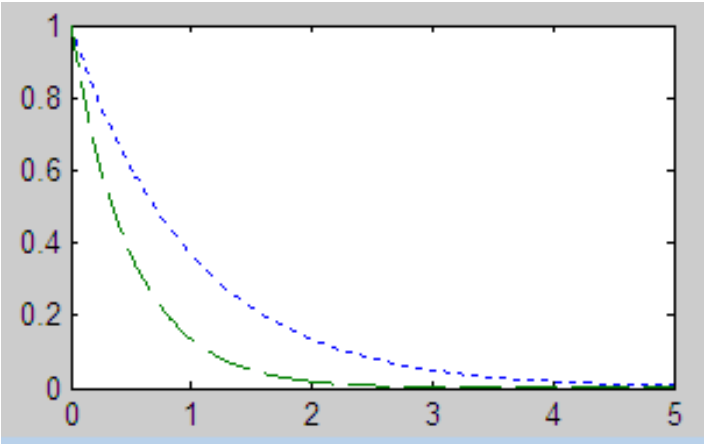
```
>> t = [0:0.01:5];
>> f = exp(-t);
g = exp(-2*t);
>> plot(t,f,t,g,'--')
```



MATLAB has four basic line types that can be defined in a plot. These are, along with the character strings, used to define them in the plot command:

- Solid line $'-'$
- Dashed line $'--'$
- Dash-dot line $'-.'$
- Dotted line $':'$

$>>\ plot(t, f, ':', t, g, '-')$



Adding Legends

A professionally done plot often has a legend that lets the reader know which curve is which. In the next example, let's suppose that we are going to plot two potential energy functions that are defined in terms of the hyperbolic trig functions $sinh(x)$ and $cosh(x)$ for $0 \leq x \leq 2$.

First we define *x*:

$>> \ x \ = \ [0{:}\,0.01{:}\,2];$

Now we define our two functions. There is nothing magical about calling a function *y* or anything else in MATLAB, so let's call the second function *z*. So we have

$>> \ y \ = \ sinh(x);$

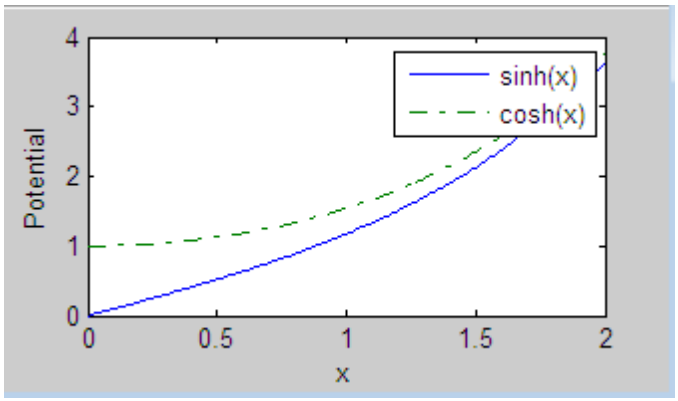
$>> \ z \ = \ cosh(x);$

The *legend* command is simple to use. Just add it to the line used for the $plot(x, y)$ command and add a text string enclosed in single quotes for each curve you want to label. In our case we have:

$legend('sinh(x)', 'cosh(x)')$

We just add this to the plot command. For this example, we include *x* and *y* labels as well, and plot the curves using a solid line for the first curve and a dot-dash for the second curve:

```
Command Window
>> x = [0:0.01:2];
>> y = sinh(x);
>> z = cosh(x);
>> plot(x,y,x,z,'-.'),xlabel('x'),ylabel('Potential'),legend('sinh(x)','cosh(x)')
```



\

Setting Colors

The color of each curve can be set automatically by MATLAB or we can manually select which color we want. This is done by enclosing the appropriate letter assigned to each color used by MATLAB in single quotes immediately after the function to be plotted is specified. Let's illustrate with an example.

Let's plot the hyperbolic sine and cosine functions again. This time we'll use a different interval for our plot, we will take $-5 \leq x \leq 5$. So we define our data array as

```
>> x = [-5:0.01:5];
```

Now we redefine the functions. Remember if we don't do this and we're in the same session of MATLAB, the program is going to think that the functions are defined in terms of the previous x we had used. So now we type:

```
>> y = sinh(x);
>> z = cosh(x);
```

Now we will generate the plot representing *y* with a red curve and *z* with a blue curve. We do this by following our entries for *y* and *z* in the *plot* function by the character strings 'r' and 'b' respectively. The command looks like this:

```
>> plot(x,y,'r',x,z,'b')
```

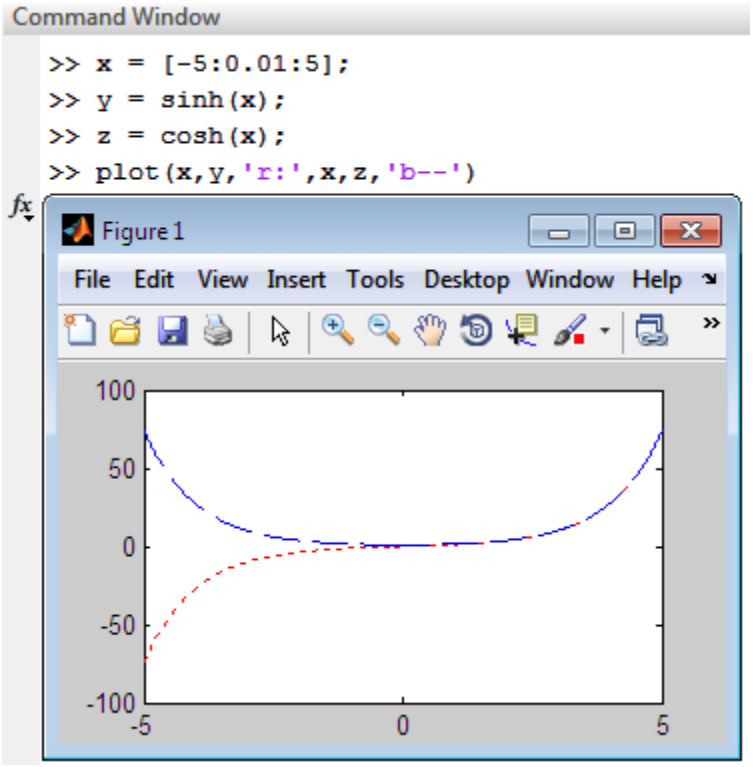


Table 3-1 MATLAB specifiers for selecting plot colors.

Color	Specifier
White	w
Black	k
Blue	b
Red	r
Cyan	c
Green	g
Magenta	m
Yellow	y

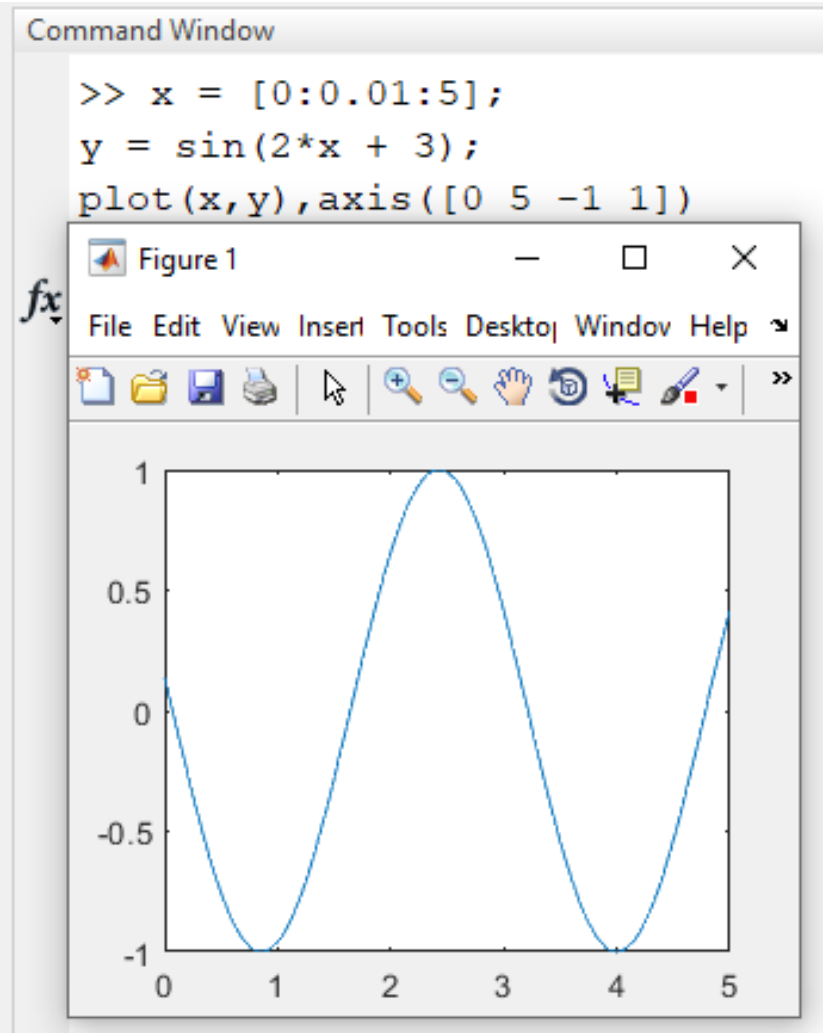
Setting Axis Scales

Let's take another look at the axis command and see how to set the plot range. This is done by calling axis in the following way:

$axis\ ([xmin\ xmax\ ymin\ ymax])$

Suppose that we want to generate a plot of $y = sin(2x + 3)$ for $0 \leq x \leq 5$. We might consider that the function ranges over $-1 \leq y \leq 1$. We can set the y axis to only show these values by using the following sequence of commands:

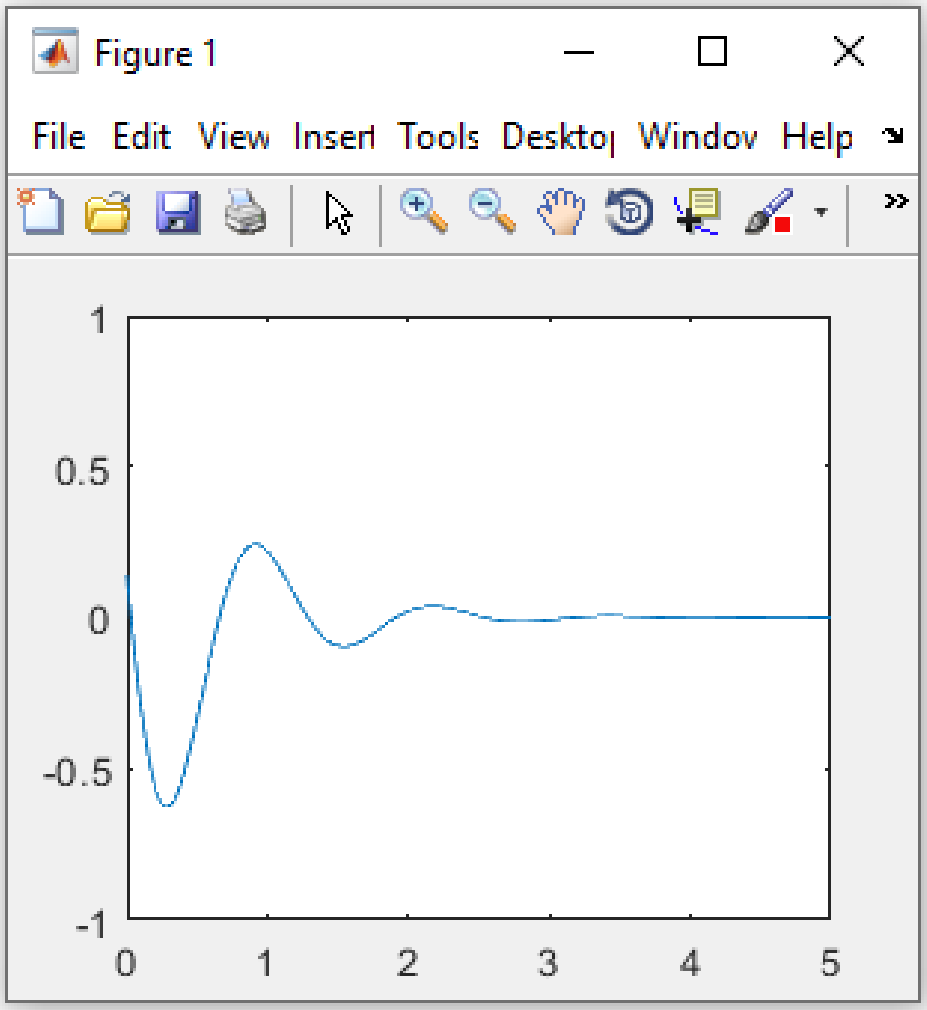
Example 1:-



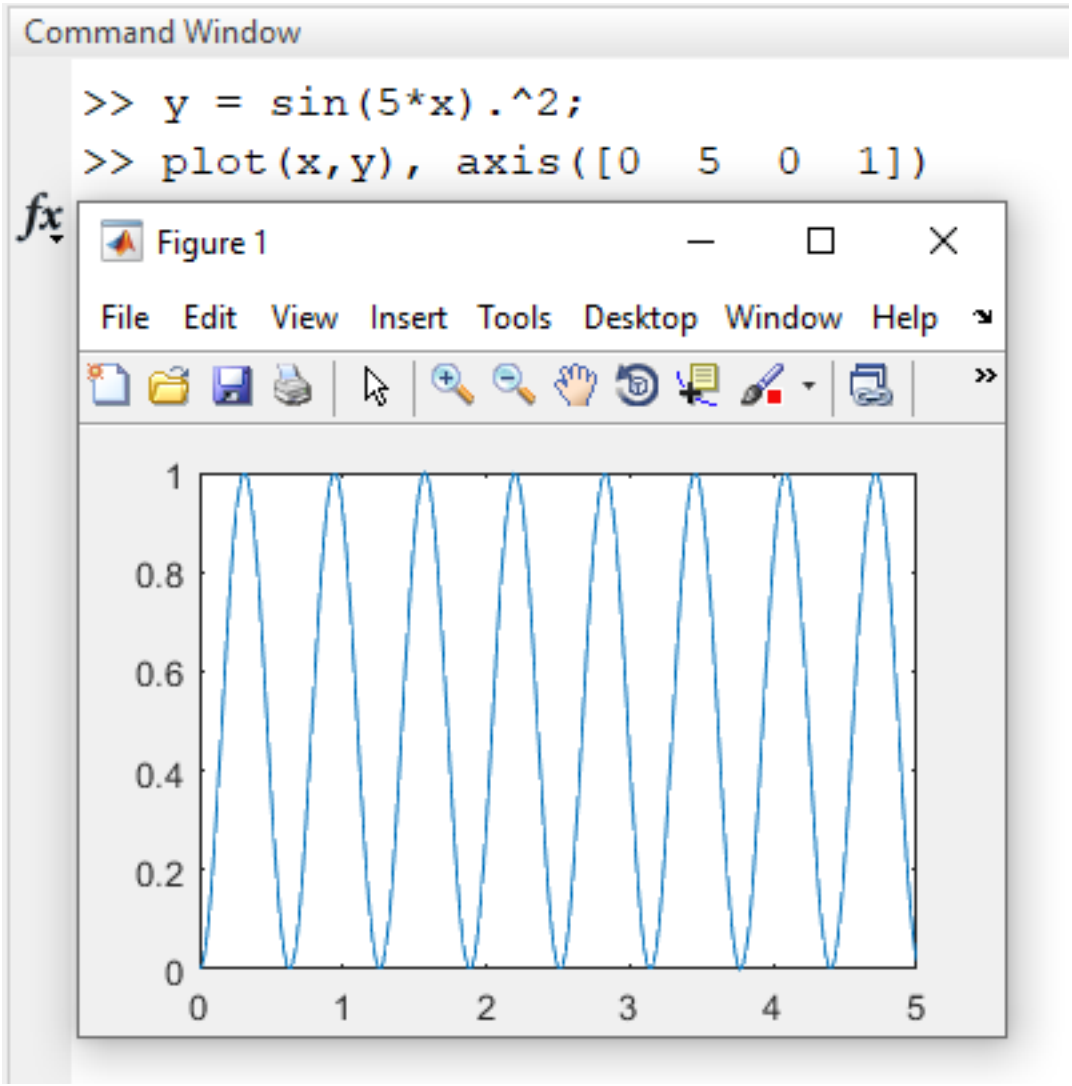
Example 2 :-

Now let's make a plot of $y = e - 3/2xsin(5x + 3)$. First we try $0 \leq x \leq 5, -1 \leq y \leq 1$.

```
>> y = exp(-1.5*x).*sin(5*x+3);
plot(x,y),axis([0 5 -1 1])
```



Example 3 :-


Command Window
>> y = sin(5*x).^2;
>> plot(x,y), axis([0 5 0 1])
Figure 1
File Edit View Insert Tools Desktop Window Help
1
0.8
0.6
0.4
0.2
0
0 1 2 3 4 5

Subplots

A *subplot* is one member of an array of plots that appears in the same figure. The subplot command is called using the syntax $subplot(m,n,p)$. Here m and n tell MATLAB to generate a plot array with m rows and n columns. Then we use p to tell MATLAB where to put the particular plot we have generated. As always, these ideas are best illustrated with an example.

Each plot created with the subplot command can have its own characteristics. For our first example, we will show $y = e^{-1.2x}sin(20x)$ and $y = e^{-2x}sin(20x)$ side by side.

In both cases, we will set $0 \leq x \leq 5$ and $-1 \leq y \leq 1$.

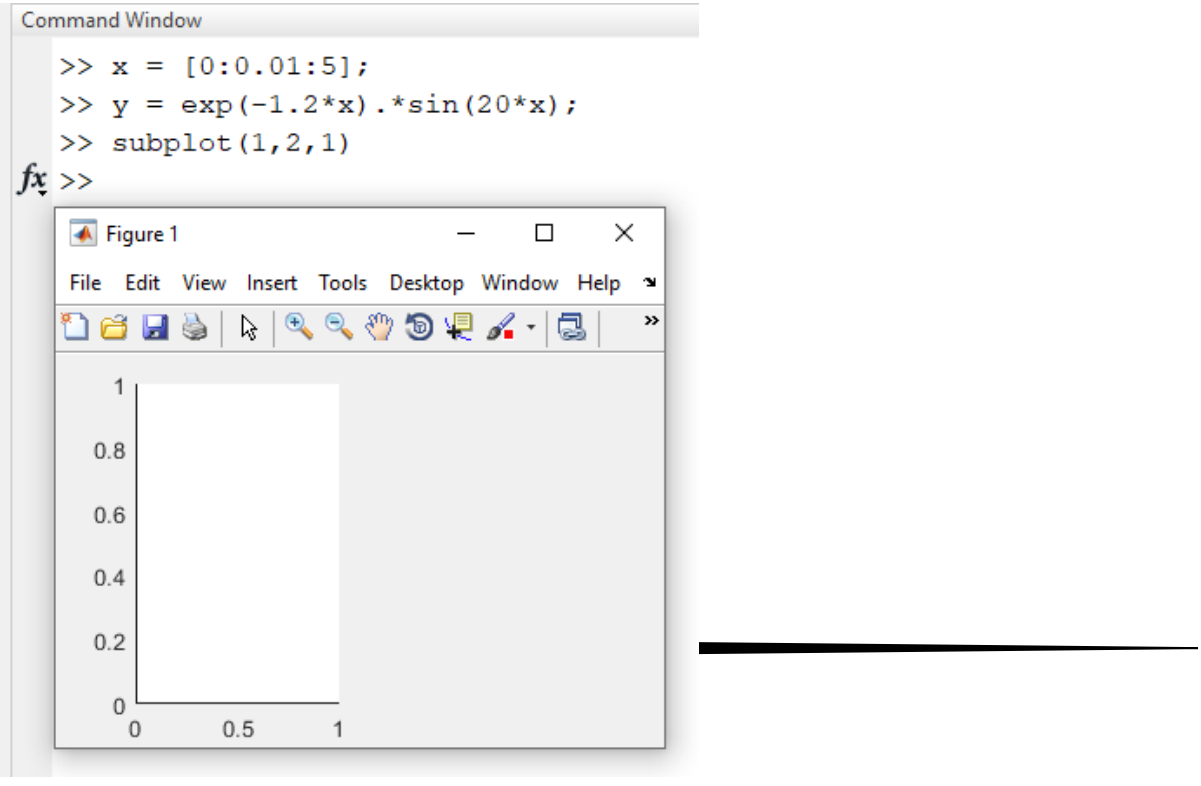
First we define the values used in our domain, define the first function, and then make a call to subplot:

Example 1 :-

$>> x = [0:0.01:5];$

$>> y = exp(-1.2*x).*sin(20*x);$

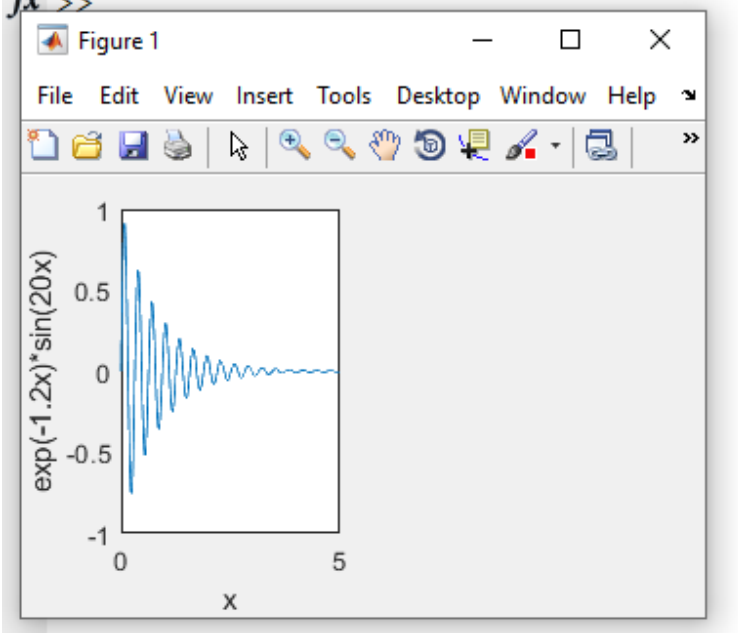
$>> subplot(1,2,1)$



Example 2:-

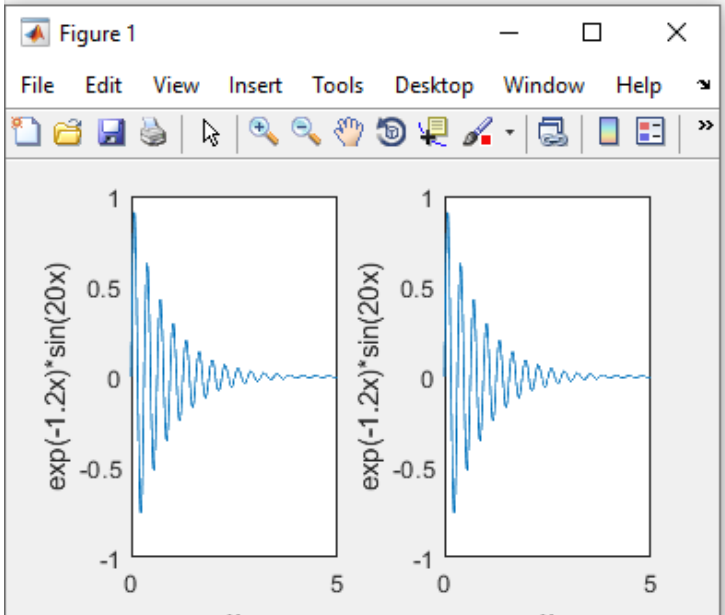
Command Window

```
>> x = [0:0.01:5];
>> y = exp(-1.2*x).*sin(20*x);
>> subplot(1,2,1)
>> plot(x,y),xlabel('x'),ylabel('exp(-1.2x)*sin(20x)'),axis([0 5 -1 1])
fx >>
```



Example 3:-

```
>> x = [0:0.01:5];
y = exp(-1.2*x).*sin(20*x);
subplot(1,2,2)
plot(x,y),xlabel('x'),ylabel('exp(-1.2x)*sin(20x)'),axis([0 5 -1 1])
>>
```



Overlaying Plots and linspace

Let's suppose that we plot a function, and then decide that we want the plot of a second function to appear on the same graph. We can do this with two calls to the plot command by telling MATLAB to *hold on*.

In the following example we will generate plots of cos(x) and sin(x) and place them on the same graphic. First, let's learn a new command that can be used to generate a set of x data. This can be done using the linspace command. It can be called in one of two ways. If we write:

x = linspace(a,b)

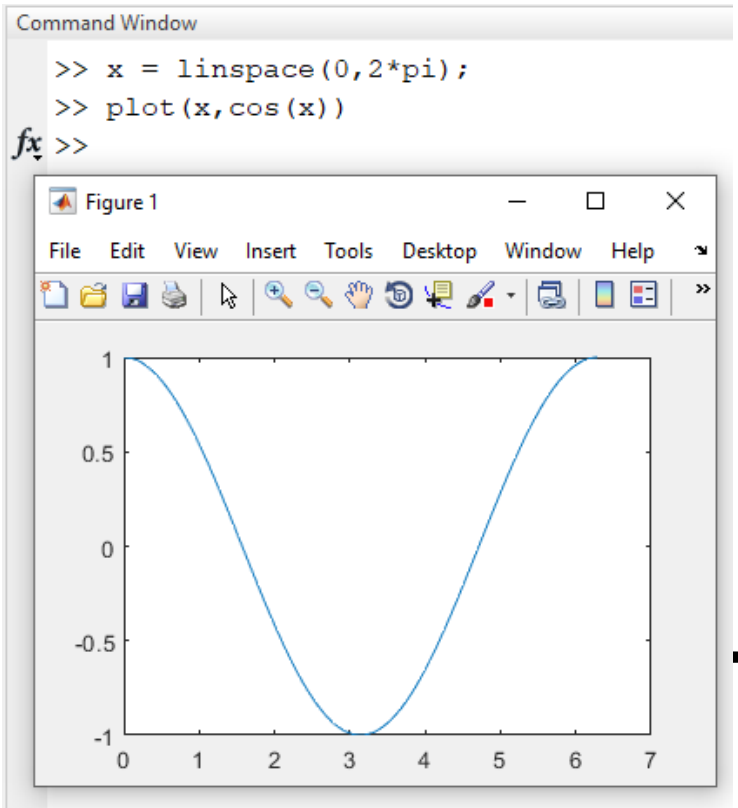
Then MATLAB will create a line of n uniformly spaced points from a to b. Now let's use this tool to plot cos(*x*) and sin(*x*). We define a set of 100 linearly spaced points from 0 to 2π by entering the following command:

Example 1:-

```
>> x = linspace(0,2*pi);
```

Now let's plot cos(x);

```
>> plot(x,cos(x))
```

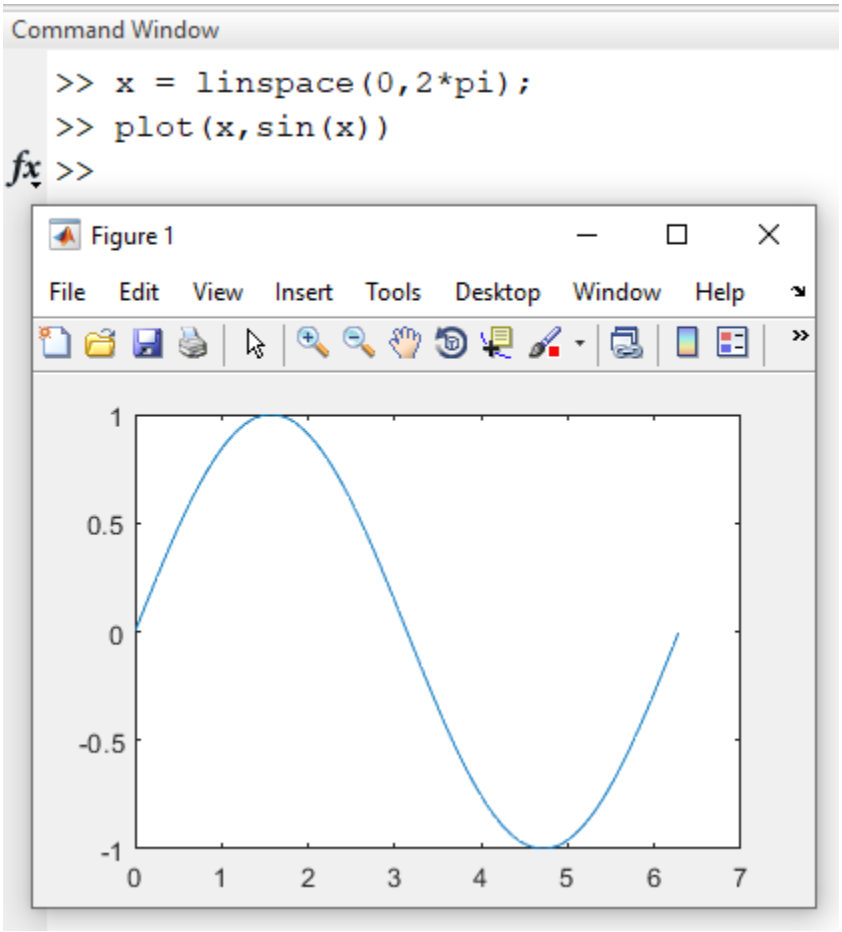


Example 2:-

We get the graphic shown in following Figure

If now type:

```
>> plot(x,sin(x))
```

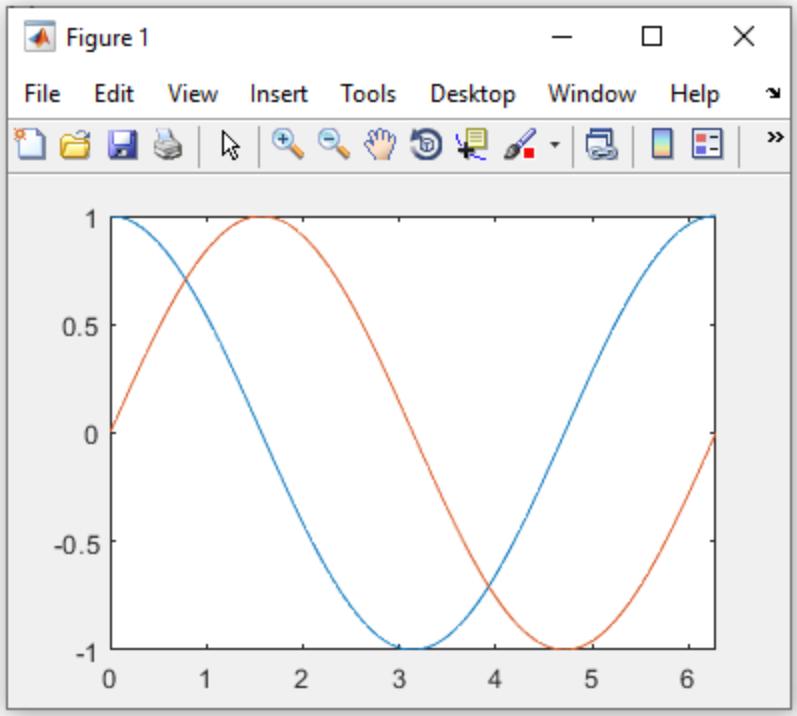


Example 3:-

```
>> x = linspace(0,2*pi);
>> plot(x,cos(x)),axis([0 2*pi -1 1])
>> hold on
>> plot(x, sin(x)), axis ([0 2*pi -1 1])
```

Command Window

```
>> x = linspace(0,2*pi);
plot(x,cos(x)),axis([0 2*pi -1 1])
hold on
plot(x, sin(x)), axis ([0 2*pi -1 1])
>>
```



Problems

1. Plot the function $f(x) = \dfrac{x^2 - 3x + 7}{\sqrt{2x+5}}$ for $-1 \le x \le 5$.

2. Plot the function $f(x) = (3\cos x - \sin x)e^{-0.2x}$ for $-4 \le x \le 9$.

3. Plot the function $f(x) = \dfrac{x^2}{2 + \sin x + x^4}$ for $-4 \le x \le 4$.

4. Plot the function $f(x) = x^3 - 2x^2 - 10\sin^2 x - e^{0.9x}$ and its derivative for $-2 \le x \le 4$ in one figure. Plot the function with a solid line, and the derivative with a dashed line. Add a legend and label the axes.

5. Make two separate plots of the function $f(x) = -3x^4 + 10x^2 - 3$, one plot for $-4 \le x \le 3$ and one for $-4 \le x \le 4$.

6. Use the `fplot` command to plot the function $f(x) = (\sin 2x + \cos^2 5x)e^{-0.2x}$ in the domain $-6 \le x \le 6$.

7. Plot the function $f(x) = \sin^2(x)\cos(2x)$ and its derivative, both on the same plot, for $0 \le x \le 2\pi$. Plot the function with a solid line, and the derivative with a dashed line. Add a legend and label the axes.

1) Solution :-

Script file:

```
clear, clc
%.1 is usually a good interval to start with - then adjust if necessary
x=-1:.1:5;
f=(x.^2-3*x+7)./sqrt(2*x+5);
plot(x,f)
%note all plot annotation functions will accept some basic tex syntax
title('f(x)=(x^2-3x+7)/sqrt(2x+5)')
%and latex commands for fancier
%title('$$f(x)=\frac{x^2-3x+7}{\sqrt{2x+5}}$$','Interpreter','latex')
xlabel('x-->')
ylabel('f(x)-->')
```

Figure Window: