Introduction to MATLAB

1. Mathematical functions

MATLAB offers many predefined mathematical functions for technical computing which contains a large set of mathematical functions.

Typing `help elfun` and `help specfun` calls up full lists of *elementary* and *special* functions respectively.

There is a long list of mathematical functions that are *built* into MATLAB. These functions are called *built-ins*. Many standard mathematical functions, such as $\sin(x)$, $\cos(x)$, $\tan(x)$, e^x, $ln(x)$, are evaluated by the functions `sin`, `cos`, `tan`, `exp`, and `log` respectively in MATLAB.

Table 2.1: Elementary functions

`cos(x)`	Cosine	`abs(x)`	Absolute value
`sin(x)`	Sine	`sign(x)`	Signum function
`tan(x)`	Tangent	`max(x)`	Maximum value
`acos(x)`	Arc cosine	`min(x)`	Minimum value
`asin(x)`	Arc sine	`ceil(x)`	Round towards $+\infty$
`atan(x)`	Arc tangent	`floor(x)`	Round towards $-\infty$
`exp(x)`	Exponential	`round(x)`	Round to nearest integer
`sqrt(x)`	Square root	`rem(x)`	Remainder after division
`log(x)`	Natural logarithm	`angle(x)`	Phase angle
`log10(x)`	Common logarithm	`conj(x)`	Complex conjugate

In addition to the elementary functions, MATLAB includes a number of predefined

Table 2.2: Predefined constant values

`pi`	The π number, $\pi = 3.14159\ldots$
`i,j`	The imaginary unit i, $\sqrt{-1}$
`Inf`	The infinity, ∞
`NaN`	Not a number

Example

We illustrate here some typical examples which related to the elementary functions previously defined.

As a first example, the value of the expression $y = e^{-a}\sin(x) + 10\sqrt{y}$, for $a = 5$, $x = 2$, and $y = 8$ is computed by

```
>> a = 5; x = 2; y = 8;
>> y = exp(-a)*sin(x)+10*sqrt(y)
y  =
   28.2904
```

2. Basic plotting

MATLAB has an excellent set of graphic tools. Plotting a given data set or the results of computation is possible with very few commands. You are highly encouraged to plot mathematical functions and results of analysis as often as possible. Trying to understand mathematical equations with graphics is an enjoyable and very efficient way of learning mathematics. Being able to plot mathematical functions and data freely is the most important step, and this section is written to assist you to do just that.

The basic MATLAB graphing procedure, for example in 2D, is to take a vector of x-coordinates, $\mathbf{x} = (x_1, \ldots, x_N)$, and a vector of y-coordinates, $\mathbf{y} = (y_1, \ldots, y_N)$, locate the points (x_i, y_i), with $i = 1, 2, \ldots, n$ and then join them by straight lines. You need to prepare x and y in an identical array form; namely, x and y are both row arrays or column arrays of the *same* length.

The MATLAB command to plot a graph is `plot(x,y)`. The vectors $\mathbf{x} = (1, 2, 3, 4, 5, 6)$ and $\mathbf{y} = (3, -1, 2, 4, 5, 1)$ produce the picture shown in Figure 2.1.

```
>> x = [1 2 3 4 5 6];
>> y = [3 -1 2 4 5 1];
>> plot(x,y)
```

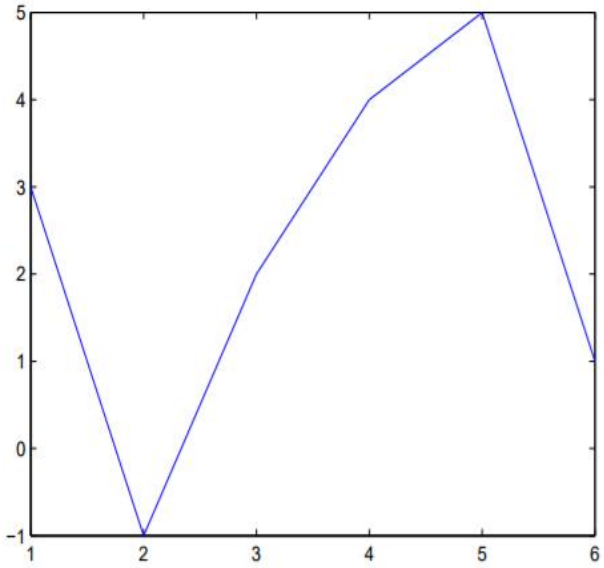

Figure 2.1: Plot for the vectors x and y

```
>> x = 0:pi/100:2*pi;
>> y = sin(x);
>> plot(x,y)
```

NOTES:

- `0:pi/100:2*pi` yields a vector that
  - starts at 0,
  - takes steps (or increments) of $\pi/100$,
  - stops when 2π is reached.
- If you omit the increment, MATLAB automatically increments by 1.

3. Adding titles, axis labels, and annotations

MATLAB enables you to add axis labels and titles. For example, using the graph from the previous example, add an x- and y-axis labels.

Now label the axes and add a title. The character `\pi` creates the symbol π. An example of 2D plot is shown in Figure 2.2.

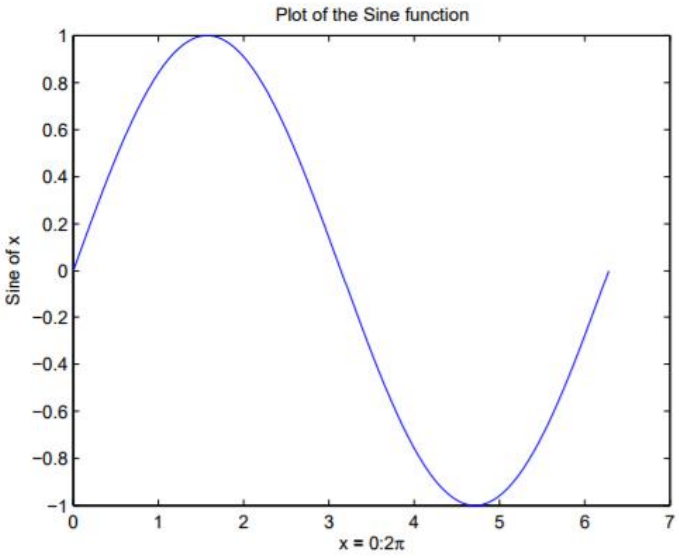

Figure 2.2: Plot of the Sine function

```
>> xlabel('x = 0:2\pi')
>> ylabel('Sine of x')
>> title('Plot of the Sine function')
```

The color of a single curve is, by default, blue, but other colors are possible. The desired color is indicated by a third argument. For example, red is selected by `plot(x,y,'r')`. Note the single quotes, ' ', around `r`.

4. Multiple data sets in one plot

Multiple (x, y) *pairs* arguments create *multiple* graphs with a single call to *plot*. For example, these statements plot three related functions of x: $y_1 = 2\cos(x)$, $y_2 = \cos(x)$, and $y_3 = 0.5 * \cos(x)$, in the interval $0 \leq x \leq 2\pi$.

```
>> x = 0:pi/100:2*pi;
>> y1 = 2*cos(x);
>> y2 = cos(x);
>> y3 = 0.5*cos(x);
>> plot(x,y1,'--',x,y2,'-',x,y3,':')
>> xlabel('0 \leq x \leq 2\pi')
>> ylabel('Cosine functions')
>> legend('2*cos(x)','cos(x)','0.5*cos(x)')
```

```
>> title('Typical example of multiple plots')
>> axis([0 2*pi -3 3])
```

The result of multiple data sets in one graph plot is shown in Figure 2.3.

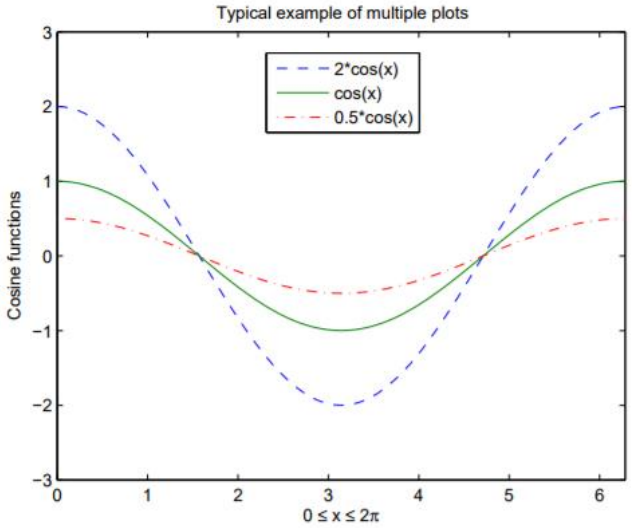

Figure 2.3: Typical example of multiple plots