Creating Your Own Functions and Scripts

You can create your own functions within MATLAB using M-files. An M-file is an ASCII text file that has a filename ending with .m, such as stars.m. You can create and edit M-files with any text editor (not just the built-in one). There are two classes of M-files, functions and scripts.

Function definitions are stored in files with the name function-name.m. The first line of a function definition must start with the word **function** and can be of the form:

```
function  function-name(argument1, argument2,...)
```

This specifies the name of the function and its input arguments. For instance, the first line of a function definition for **stars** from the M-file named stars.m would be **function stars(t)**.

Functions Returning No Values

A common example of a function that doesn't return any value is one that draws a graph.

```
function stars(t)
%STARS(T)       draws stars with parameter t
n = t * 50;
plot(rand(1,n), rand(1,n),'.');
%that line plots n random points
title('Oh My, Look at all the Stars!');
%label the graph
```

When **stars** is executed, it will plot a random sprinkling of points in your graphics window on a set of axes. The first line in the M-file defines this as a function named **stars** that takes one argument named **t**, and doesn't return anything. The next line is the help comment. To include help info for your own functions, just start the text line in the corresponding M-file with a **%**. Any comments preceded by a **%** and coming immediately after the first line in the M-file are returned by the **help** command from within MATLAB. For example:

```
>>  help stars
STARS(T)       draws stars with parameter t
```

A line of comments is also indicated by the **%** character. You can have as many of these as you want, just remember that the **help** command only prints the ones immediately following the function definition at the beginning of the M-file. Next, the function defines an internal variable named **n** to be fifty times **t**. This **n** is totally unrelated to any variable already defined in the main MATLAB workspace. Assigning this value will not alter any **n** you had already defined before calling the **stars** function. You can also see how we put a comment in the middle of the function indicating which command actually drew the stars.

Functions Returning One Value

Functions may return scalar or matrix values. The first line of such functions is of the form:

```
function variable = function-name(argument1,
argument2,...)
```

where the variable will be set equal to the output value somewhere in the function definition. Here is an example where **y** is set to equal the function **fliplr**.

```
function y = fliplr(x)
%FLIPLR(X) returns X with row preserved and columns flipped
%in the left/right direction.
```

```
%  X = 1 2 3     becomes  3 2 1
%      4 5 6              6 5 4

[m,n] = size(x);
y = x(:,n:-1:1);
```

Functions Returning More Than One Value

If you want to return more than one argument, you can do it by having the function return a vector of values. For example, the following function returns two vectors. The first vector is the prime factors less than 10 of the argument. The second indicates how many times each of these factors is used.

```
function [factors, times] = primefact(n)
%[FACTORS TIMES] = PRIMEFACT(N) find prime factors of n

primes = [2 3 5 7];

for i = 1:4
   temp = n;
     if ( rem(temp,primes(i)) == 0)
        factors = [factors primes(i)];
        times = [times 0];
            while (rem(temp,primes(i)) == 0)
             temp = temp/primes(i);
             times(length(times)) = times(length(times))+1;
            end
     end
end
```

If you call this function with just one argument (for example, **a = primefact(10)**) the function returns a vector of prime factors, but not the vector indicating how many times each factor appears in the number. To get both vectors, you would call the function as follows:

```
>> [a b] = primefact(180)
a =
     2    3    5
b =
     2    2    1
```

This way, both vectors are returned: the primes in **a**, and the number of times each prime was used in **b**. From these results, you can see that 180=2*2*3*3*5.

Script M-files

A script M-file is simply a file of MATLAB commands as you would type them at the >> prompt. A script does not begin with a function line. Its contents are executed in sequence whenever you type their name. This also differs from a function M-file in that there are no arguments or output values. Also, the variables inside a script file are the same ones as in the main MATLAB workspace. If you type **n = 1000**, then execute a script that ends with the line **n = 2**, **n** would then be 2, and not 1000.

To create a script file, just create a text file that contains the commands you want executed, and save it in the working directory. A script file does not need comments for the **help** command. The filename should still end in .m.