Programming in Matlab

While Matlab is a interactive programming environment, the Matlab programming language is a mid-level language similar to fortran. Matlab provides many data types, but the default data type is a double precision complex matrix. The Matlab command window is a command-line computational environment, where instructions from the Matlab programming language can be given, one line at a time, with the results saved as variables and accessible to other instructions. This is Matlab in *interpretive mode.*

This can be a fruitful way to explore data or simulate a dynamical system, etc., but often we would like to save a sequence of instructions so that they can be used again later, or applied to many problems. Matlab provides two similar, but distinct ways to do this: functions and scripts. Both function files and script files are text files which contain Matlab instructions. They contain source code for Matlab programs. Both function files and script files have filenames of the form   filename.m   and must be in a directory (folder) that Matlab will look in (try `help path`). Now for the differences.

A Matlab *script routine* is simply a collection of instructions saved in a file with a .m extension. For example, if we type the instructions below into a file called `urandab.m`, then we can execute them simply by typing `urandab` at the Matlab prompt.

```
% x is an m x n matrix of pseudo-random numbers
% uniformly distributed in the interval [a,b].
u = rand(m,n);
x = a + (b-a).*u;
```

For example, if I have variables `m=1, n=2, a=100, b=200, u=3` then typing `urandab` gives `x=[178.96 132.55], u=[.78962 .32549]`. Notice that the Matlab environment must have values assigned to m, n, a and b, before I call `urandab`. Notice also, that the values of x and u got changed by `urandab`, just as if I had entered the lines at the Matlab prompt.

A Matlab *function routine* is a collection of instructions that has its own environment. It cannot see any variables except those passed in, and it cannot change any variables except those passed out. Its first line should be   `function` $[o_1, o_2, \ldots, o_s]$=`myname`$(i_1, i_2, \ldots, i_t)$ When the function is invoked, variables on the rhs are passed into the function and named within the function environment *by order.* When the function is finished executing, the output is written to the variables on the lhs by order. Here is a function version of urandab in the file `urandab2`

```
function x = urandab2(m,n,a,b)
% x is an m x n matrix of pseudo-random numbers
% uniformly distributed in the interval [a,b].
u = rand(m,n);
x = a + (b-a).*u;
```

Now if I have variables `r=1, b=200, u=3` then typing   `z = urandab2(r,2,100,b)` gives `z=[178.96 132.55], u=3`. Notice that was u unaffected by urandab2 and that the output is called `z`, not `x`. The fact that the function environment is sequestered is tremendously useful, and when you write a function routine, you have extended the capabilities of the Matlab environment.