Tutorial 3: Creating Functions

Getting Started

1. In this tutorial, we will develop two very simple, but very useful functions.

2. Start Matlab. Change directories to some neutral location (for example, the directory you created for Tutorial 2).

Mapping

3. One task I seem to do over and over is to transform a set of data defined on one interval (say 10 to 20) to another (say 0 to 1). Fortunately, this is fairly easy. To map a set of data to [0, 1], subtract the minimum of the data set and then divide by the difference between the data's max and min. For example, if the data are stored in a vector x, then

   ```
   >>newx=x-min(x);
   >>newx=newx/(max(x)-min(x));
   ```

   Prove to yourself that this procedure works. Create a vector x (say x= -10:5:20), create newx as above, and then check that min(newx) is 0 and max(newx) is 1.

4. Now, assume we want to map x to the interval [a b]. One way of doing this is to first map x to [0 1] as above and then multiply by b-a and add a, e.g.

   ```
   >>x2=newx*(b-a);
   >>x2=x2+a;
   ```

   Pick an a and b and prove to yourself that this procedure works.

5. As you can see, this mapping procedure is fairly simple—only two lines of code. Yet, if you have to do it a lot, remembering exactly what to type and typing it all out is tedious. What we want is a function that takes a data set and an interval and maps that data on to that interval. Graphically, we can represent this as

   What should the first line of mapint.m be? Try it yourself before checking the mapint.m file on the web.

6. Developing a computer program involves breaking a problem up into smaller and smaller pieces. A well-established procedure for developing computer programs is called iterative refinement. This procedure involves describing the essential stages of a program in English. This procedure is then applied to each stage, breaking them up into sub-stages, until you reach something that looks like computer code. For mapint, we start with a description of the problem:

   "map x on to the interval intv=[a,b]"

   From our work above, we can break this into two stages:

   "map x to [0 1]"
   "multiply by (b-a) then add a"

7. An effective strategy is to do the iterative refinement inside your editor by entering the descriptions as comments. From the Matlab prompt, type "edit" to open Matlab's editor. Then type the function line for mapint. Go ahead an put in some simple help comments—at the minimum, copy the function line and paste it as a comment so you know how to call the function. Then, type the description of the two stages as comments. Save the file to your working directory and call it mapint.m

8. At this point, we could just copy the code we developed in 3-4. However, mapping to [0 1] is a special case, and it might be nice to have a separate function to do this. The function should take a data set x and use its min & max to map it to [0 1]. What are the inputs and outputs for this function and what should the first line of map01.m be? Open a new file and start creating this function.

9. Copy the code to map x to [0 1] and paste it into map01.m and save the file. Then, try calling it from the command line:

   ```
   >>newx2=map01(x);
   ```

10. Once map01 works, put a call to it below the "map x to [0 1]" line in mapint. Then, write the code for the next stage. Note: we're assuming the interval is defined as a 1-by-2 vector called intv, not as scalars a and b. You have two options, you may either create a and b and fill them with values from intv or replace reference to a and b with references to inv. When you're done, save the file and try calling mapint from the command line:

    ```
    >>newx3=mapint(x,[a,b]);
    ```

11. Some issues. The code we've developed works, but there are some improvements we could make. What if are data are organized as a 2D array rather than a matrix. Would map01 and mapint work? Why not?

There are two logical strategies that you could employ. One would be to assume the data are organized in columns and do the mapping independently for each column. Another would be to assume that the data is a single entity and transform everything together. I'll leave it up to you to work out the details, but the second option has been used in the code on the web…