Lab Report #0: Matlab Practice

Revised: December 2, 2012

This will not be collected or graded, but please do it before the second or third class.

Matlab is a popular high-level computer language, which means that it's easier to use than C++ or Java and (far) more powerful than Excel. It's widely used in quantitative segments of the business world, including consulting, banking, and asset management.

What follows will get you started. It should take you about an hour once you have Matlab installed.

1. *Accessing Matlab at NYU.* You can access Matlab in three ways at NYU:

   - Buy a student version at the bookstore (roughly $100 for a one-year license)
   - Online with NYU id: vcl.nyu.edu
   - Online with Stern id: apps.stern.nyu.edu

   I haven't used either of the online versions, but have been told that they are sensitive to the browser you use and behave better with IE.

   If you're a little braver, Octave is an open source program that's virtually identical for basic things. It's missing the nice GUI and the symbolic math is less user-friendly. If you're interested, see: http://www.gnu.org/software/octave/.

   Your mission: choose one of the above. Once you do, start it up and type `demo` on the command line to get a demonstration of Matlab's basic features.

2. *Scalar operations.* We'll start by entering commands on the command line in what Matlab calls the Command Window. If you can't find it, look for ">>" (the "prompt").

   (a) Type each at the prompt:

   ```
   x=7
   x=7;
   x
   7^2
   ans
   ```

   What do each of these do? What does the semi-colon do? What if you skip it?

   (b) Now type these: `pi`, `exp(1)`, `e`, `x = exp(2)`, `log(x)`. What do they do?

3. *Vector operations.* Matlab treats everything as a vector or matrix. What this means is that you can do a bunch of things at once rather than copying the same command over and over again as you would in (say) Excel. Here's an example:

   (a) Generate a grid with the command: `x = [-3:0.5:3]`. What does `x` look like?

   (b) Now compute x^2 for all values of x:

```
xsquared = x.^2
```

The dot means to do the operation element by element. That way, we can compute the square for every element of x with one line.

(c) Plot the results this way: `plot(x,xsquared)`. Type `help plot` to learn how to control the type and color of the line. Can you make the line red? Dashed?

(d) Plot the functions `normpdf(x)` and `normcdf(x)`. (You can enter these expressions directly in the plot command.) Use the help command to find out what these functions are — for example, `help normpdf`. What do you see? How would you make the line smoother?

4. *Symbolic math.* We'll derive the moments of the normal distribution from its moment generating function with Matlab doing the most of the work. Type these commands in the command line:

```
syms mu sigma s
mgf = exp(mu*s + (sigma*s)^2/2)
mu1 = subs(diff(mgf,s,1),s,0)
mu2 = subs(diff(mgf,s,2),s,0)
mu4 = subs(diff(mgf,s,4),[mu sigma s],[0 1 0])
```

What do they do? What happens if you skip the first line? What is `mgf`? What is `diff(mgf,s,2)`? What does `subs` do?

5. *Data input and scripts.* Our last task is to input data from a spreadsheet using a "script": that is, we put the commands in a file so we can run them over again without typing everything in from scratch.

(a) Enter the following data into a spreadsheet:

$x1$	$x2$
1	2
3	4
5	6

Save it as an Excel workbook with name (say) `testdata.xls`. Make sure you know what directory it's in.

Comments: (i) You can use any file name you like, but the same name must be used in the script below. (ii) **Warning for Mac users:** In Windows, all Excel formats are recognized, but Matlab for Macs does not recognize xlsx files.

(b) Write a Matlab script to read the contents of the spreadsheet into Matlab: Inside Matlab, go to the upper left corner, click on File, and choose New and Script. This should open the Matlab editor. Now enter these commands:

```
%  Practice script for data input from spreadsheet (this is a comment)
format compact
disp('Spreadsheet input')
data = xlsread('testdata.xls')
```

Save these commands under some appropriate file name in the same directory as the spreadsheet. It will automatically be given the file extension m (for Matlab).

Now run the program by clicking on the green triangle at the top of the editor. (An alternative is to type the file name, without the m, on the command line.) Then go to the Command Window to see if it works. If not, welcome to the world of programming, where most of your time is spent fixing bugs. Remind yourself that patience is a virtue.

(c) Add to your program a line that plots x1 against x2 and marks each data point with a circle.