Econ 509
Common errors and some comments on Homework 2

Error #1 – Not checking number of iterations against maximum iterations – When utilizing a loop it is important to account for the event of the maximum iterations being reached before the solution (to the degree of accuracy desired) is obtained. If you don’t, you may end up returning or using values that are incorrect (this is really an issue when you use programs written by someone else that you are not as familiar with).

Below are two silly examples, one bad function, one good function. The functions calculate the value of ‘x’ that solves the equation (Number - x) = 0. Obviously, if the input number is, for example 42, then the function should return 42 (for simplicity assume positive integers). Notice that both functions will yield the same results if the input number is <=100. But for numbers larger than 100 the BAD example will always return 100 as the solution (so SillyExBad(500) will return 100). Obviously this is wrong. The Good example will instead give an error message and stop the program. You could use the warning command instead if you really want the program to continue, even with the incorrect answer.

function i = SillyExBAD(Number)
    MaxIt = 100; Difference = 0;
    for i = 1:MaxIt
        if (Number – i) == 0, return, end
    end

function i = SillyExGOOD(Number)
    Iters = 0; MaxIt = 100; Difference = 0;
    for i = 1:MaxIt
        Iters = Iters + 1;
        if (Number – i) == 0, return, end
        if Iters == MaxIt
            error('Max iterations reached before solution found'), end
    end

Error #2 – Not checking dimensions of matrices – For the function two_goods(d,s) you should check at the very beginning that the matrices are of the proper size, and if not, have an error message alerting the user as to what the problem is. For example:

if size(d) ~=[2,3] | size(s) ~=[2,3]
    error('The supply and demand matrices must be of dimensions 2x3')
end

Error #3 – Not properly formatting output – In questions 3-8 you were asked to output your answers for a fixed number of decimals. One way to do this is to use the fprintf command. For example, if I want to display PI to eight decimal places I would use fprintf and the %.8f conversion code:

fprintf('PI to 8 decimal places is %.8f',pi)
For more information, below are two links. The second is a quite lengthy tutorial which has good examples of all sorts of output formatting in Matlab (including a good section on generating tables, which can be quite confusing).

http://www.me.pdx.edu/~gerry/nmm/course/slides/ch03Slides.pdf

**Error #4 – Not checking if bounds form proper bracket** – Before solving using a root finding method (such as bisection) you should test whether the lower and upper bounds form a proper bracket. This can be done by:

    if sign(f(lower)*f(upper)) == 1
        error('Not a proper bracket. Choose new initial values') end

**Error #5** – Note that exp(8) is not the same as 1e8. exp(8) refers to the number 2.7183^8 while 1e8 refers to 10^8.

**Comment #1 – Inline functions** – Instead of writing the same equation again and again throughout your program, Matlab has a very nice built in function called inline. For example, if you need to refer to the function \( y = x^3 - x^2 +10 \) you can create the function (rather than typing it each time you need it) as follows:

    y = inline('x^3 - x^2 +10','x')

This creates a user-defined function that can be called by typing \( y(\text{number}) \). So for example \( y(1) \) would yield 10. **Please**, if you are not clear on how to use the inline command come see me because it is incredibly useful and will make coding easier.

**Comment #2 – Risk_3b.m** – Virtually everyone solved risk_3b.m (the case where initial wealth is a vector input) using a while loop. A number of solutions misspecified what the constraint should be or developed creative, but odd, methods to get the right answer. Two solutions (which have equivalent meanings ONLY IF \( b-a \) is a vector) would be:

    while  max(abs(b-a)) > epsilon         or             while  norm(b-a,inf) > epsilon

Both, for this case with vectors, will return the absolute value of the maximum distance between bounds.

**Comment #3** – Short, but insightful, comments are better than lengthy sentences that result in line overruns. Plus, they take less time to do if they are concise.

**Comment #4** – A number of students added additional features/options to their functions which is great, but it is always good to define defaults in the event someone chooses not to enter those values. For example, if in risk_3a.m you wanted to let a user be able to choose, if they want, the number of realizations of \( W \) (everyone called this variable ‘\( n \)’ then it is good to use the ‘nargin’ command. Nargin returns the number of arguments passed to a function. Here’s an example of how to use it:

    function pi = risk_3a(w0,mu,sigma,gamma,n)
    if nargin <5, n=3; end

That way, if someone passes all 5 variables to the function, it will use them. If the user does not pass a value for \( n \), then it will by default use \( n \) equal to three and proceed without error.