College of Engineering and Computer Science
Mechanical Engineering Department

California State University Northridge

Mechanical Engineering 309
Numerical Analysis of Engineering Systems

Spring 2014 Number: 15237 Instructor: Larry Caretto

Programming Assignment Six –Numerical Integration[1]

Background

This assignment asks you to prepare a function that does Romberg integration in both MATLAB and VBA. Although the language approaches will be somewhat different, the overall structure of both programs will be the same. Start in the language with which you are more comfortable and get a working code there before starting the code in the second language. You can use the pseudocode provided in the appendix to this assignment as a starting point for both MATLAB and VBA

Your function should evaluate the general definite integral, $\int_a^b f(x)dx$ for any function, and any upper and lower limits. You will be asked to apply the function to the integral shown below, which is known as the error function, erf(z).

$$\text{erf}(z) = \frac{2}{\sqrt{\pi}} \int_0^z e^{-x^2}$$

The error function is available in both MATLAB and the Excel spreadsheet so that you can check your numerical integration results against the computer functions.

1. Using MATLAB

Summarize your MATLAB results from parts a and b below as a table with six columns: (1) upper limit, z, (2) actual error function, (3) error function from your Romberg function,(4) error function from MATLAB integral function, (5) absolute value of difference between columns 2 and 3, (6) absolute difference between columns 2 and 4. The following link has information about preparing tables in MATLAB http://www.mathworks.com/help/matlab/ref/table.html.

a. Write a function myInt(f, a, b) that accepts a function handle, f, for the integrand, f(x), and the upper and lower limits, a and b, of the integral and returns a value for the integral using Romberg integration. Your function should exit with an "iterations exceeded" message after the value of N has been doubled more than 20 times and should exit when two approximations to the integral agree to within a relative difference of 10^{-10} or less. Use your integration function to compute the error function integral, erf(z), for a range of z values from 0.001 to 10. Use the MATLAB logspace function to generate these values.

b. Review the help file for the MATLAB function, **integral**. Use this function to compute the error function integral, erf(z), for the same range of z values that you used in part a.[2] **MATLAB functions for integrands must be able to process array operations; make sure that the code for your function handle can process a MATLAB array for x.**

[1] Assignment due Monday, April 28, at 11:59 pm. May be submitted by Wednesday, April 30, at 11:59 pm with a 30% penalty. No grade for assignments submitted after this date.

[2] In using the logspace function from an initial power of ten to a final power of ten it is convenient to have an exact intermediate point for each power of 10. For example, in generating the logspace entries for a range from 0.001 to 10, it would be convenient to have entries exactly at 0.01, 0.1 and 1. You can obtain such exact decade intervals by using a number of data points, N, such that N – 1 is evenly divisible by the number of decade ranges. In the example here, with four decade ranges (0.001 to 0.01, 0.01 to 0.1, 0.1 to 1, and 1 to 10) a number of points such as 25, 29, 33, *etc*. will give the desired result.

Jacaranda Hall 3314 Mail Code Phone: N/A
Email: lcaretto@csun.edu 8348 Fax: 818.677.7062

2. Using EXCEL VBA

Summarize your Excel results from parts a and b, below, as an Excel table with four columns: (1) upper limit, z, (2) actual error function from Excel function, erf, (3) error function from your Romberg function, (4) absolute value of difference between columns 2 and 3. Set an appropriate format to show the errors.

a. Write a function myInt(f, a, b), where f is a string argument giving the name of the function that computes the integrand[3], and a and b are, respectively, the upper and lower limits of the integral. These values of a and b are inputs to the function which returns a value for the integral using Romberg integration. The code in this function should use the Application.Run method of Excel that you used in your program for Newton's method in the third programming assignment. This allows anyone to use your Romberg function to compute any integral. The user simply has to write a function, `Function <integrand function>(x as double) as double`, to compute the integrand and pass the name of the integrand function to your integration routine as a string. For example, if you write a function called myErrFunction to compute the error function integrand, you would call the integration routine as myInt("myErrFunction", 0, 1) to integrate the error function between zero and one.

Your myInt function should allow the value of N to be doubled no more than 20 times and should exit when two approximations to the integral agree to within a relative difference of 10^{-10}. Use this function to compute error function integral, erf(z) for a range of z values from 0.001 to 10. You can simulate the MATLAB logspace function in Excel by defining the following factor: f = $(MAX/MIN)^{1/(N-1)}$, where MAX is the maximum table entry, MIN is the minimum table entry and N is the number of entries in the table. Enter the initial value (MIN) in a column (or row) and multiply that initial value and all subsequent values by the f factor you calculated. (You should obtain the value MAX at the end of N entries.)

Submission Requirements

No discussion is required for this assignment. You should submit only two files for this assignment. The first is a file that contains all your MATLAB results and code. The second is an Excel file with your VBA code and results.

Download and apply the programming guidelines. Attend to the use of good structure, useful comments, meaningful variable names, and well formatted code with indentations for different logical levels.

Appendix

The pseudocode for Romberg integration from the course notes is shown below. (This psuedocode uses a semicolon to separate multiple statements on the same line.)

```
maxDoubles = 20; err = 1e-10; N = 2;
cvg = false; k = 0; h = (b – a)/2;
T00 = h * [f(a) + 2 * f(a+h) + f(b)]/2;
while not cvg and k <= maxDoubles
    k = k + 1; N = 2 * N; h = (b – a)/N
    Loop to sum f(a+mh) for m = 1,3,5,…, N – 1   (odd m only)
    Tk0 = Tk-1,0/2 + h * sum
    Loop for m = 1, k to get: Tkm = (4^m Tk,m-1 – Tk-1,m-1) / (4^m – 1)
    cvg = abs(Tkk – Tk,k-1) <= err*abs(Tkk)
end while loop
if cvg then ans = Tkk else ans = errorCode
```

[3] See the solution to programming assignment three for the use of the Application.Run procedure in VBA that allows the user to send the name of a function as a string argument to another VBA procedure. Doing this for your numerical integration procedure provides VBA code that does not have to be modified for any other integral that you want to evaluate.