Math 4610 Tasksheet 1: Root Finding Algorithms

- Due Date: Friday, Sept. 16, 2022

The following tasks comprise the first homework assignment for the class. In the class, we have talked about a few basic content items. These include the download and installation of Linux to computers.

Task 1. The Root Finding Problem

The first problem is simple, but will cause some problems if your are not careful. Suppose that we consider the function

$$f(x) = x\ e^{-x}$$

as an example of a function we want to analyze. There is a root at $x^* = 0$. Build a code that will implement functional iteration and test on the fixed point functions.

$$g_1(x) = x - f(x)$$

and

$$g_2(x) = x + f(x)$$

You may not get convergence in your first efforts. However, it may be possible to modify the iteration functions, g_1 and g_2 to get convergence.

In the development, build function or module that, given f, the module, method, or subroutine will produce a result. Inputs for the module should include f, x_0 the initial approximation for a root, tol an upper bound for the error in the approximate solution, and as a precautionary measure a maximum number of iterations allowed before exiting the module. The routine should return the last approximation in the sequence of approximation.

Task 2. Tabulation of Results

Modify your code from the first task to produce a table of the results from functional iteration. That is, the code should accept a flag that will tell the code to provide a list of numerical values that include (1) the iteration number, (2) the approximate location of the root, and (3) the absolute value of the error in the approximation. In many cases, Linux commands use "-v" or "--v" to turn on verbose mode in the command. You might consider a parameter like this and put the logic in to use the parameter.

Task 3. Another Example of Functional Iteration

For this task, consider the function

$$f(x) = 10.14 \ \cdot \ e^{x^2} \ \cos\left(\frac{\pi}{x}\right)$$

on the interval $[-3, 7]$. Apply functional iteration to find roots of this problem. Check various initial points x_0. in the interval and determine if the algorithm converges at any points. Notice there is a singularity in the function near $x = 0$.

Task 4. Bisection for Root Finding Problems

Implement the Bisection Method for finding roots. Use the second version where the number of iterations is determined in the initialization steps. That is, compute

$$k = -\frac{\frac{\log(tol)}{\log(b-a)}}{\log(2)} + 1$$

where k is the number of iterations to be within the given tolerance.

The module or routine should accept inputs including the function, f, the interval endpoints, a and b, and the desired tolerance, tol. The output should include the last approximation computed. In addition, include the same verbose mode parameter, $-v$. Test your code on the functions in Tasks 1 and 3.

Task 5. Github and git start.

If you do not have a Github account, go to

Github site

and create an account. As part of this assignment build/perform the following.

- Create a repository called "math4610" for collecting work in the class.

- Using your local computer, clone the "math4610" repository locally.
- In the local version of this repository, create a file named "README.md" and include some text to identify/describe the contents of the repository. You should investigate using Markdown in this.
- Push the changes up to the Github version of the repository. Note there will be some issues with Dual Authentication algorithms. You will need to get used to this. There is a good deal of information on the Github site.
- Create a folder, for the root finding codes that you are creating. Use your local repository to push the codes you write to the Github site.