Math 131: Numerical Analysis
Homework Number 3
Due: February 24, 2024 5:00 PM

Special Instructions:

- If instructed to "write/implement a code" or "write/implement an algorithm", you should interpret this as meaning that you need to produce a code in Jupyter/python notebook.
- Unless explicitly stated, you may not use any system/python packages or functions that implement one of the algorithms for this assignment. Standard mathematical functions (e.g. pow, abs, log, sin, exp, etc.) are allowed.
- You should include all the code source you implemented as one python notebook when turning in your assignment. Your code must run as is (and provide the correct results) to receive full credit.
- PDF files of your code, screenshots, etc. will not be graded.
- All code should be properly documented and include (**at a minimum**) 1) a summary of what the code is doing, 2) a description of all parameters that are used, and 3) a description of the output.
- The notebook itself should also be properly documented. Points will be **heavily** deducted for any notebook and/or code that is not properly documented.

In addition to the above instructions, when implementing your methods below, all calls to function evaluators should take one input parameter and return the function value and optionally the derivative. Here is a bare-bone example definition (note: without proper documentation, which should be supplied by you) in python.

- Example Functions

  ```
  def myfcn(x):
    fvalue = (x-1.5)**3 - x + 2
    return fvalue
  ```

  or if you need derivatives

  ```
  def myfcn(x):
    fvalue = (x-1.5)**3 - x + 2
    fprime = 3*(x-1.5)**2 - 1
    return fvalue, fprime
  ```

1. Bisection (30 points)

(a) Implement in code the bisection method and use the code to solve the following problems up to tolerance of $|f(x)| \leq ftol = 10^{-6}$.

1. $f(x) = 2\cosh(x/4) - x$ on $[0, 10]$
2. $f(x) = \left[\left(\frac{\alpha}{x}\right)^{12} - 2\left(\frac{\alpha}{x}\right)^{6}\right]$ with $\alpha = 1$ on an interval $[0.5, 10]$.

Remark: function 2 is called the Lennard-Jones potential and can be used to describe the energy potential between two atoms or molecules. It is used in many chemistry, biology, and materials science applications.

(b) For each function, predict the number of iterations required to achieve the given tolerance and compare with the actual number of iterations your algorithm took. Analyze and explain the results.

2. Fixed-Point Method (30 points)

(a) For each of the following functions, design two (2) different functions $g(x)$ such that solving the corresponding fixed-point problem $x = g(x)$ is equivalent to solving the root finding problem $f(x) = 0$. Show that the root finding problem and the corresponding fixed-point problem are equivalent. Note: All $g(x)$ must be distinct and provably convergent.

1. $f(x) = x^5 + 5x^3 - x^2 + 1$ on the interval [-1, 2].
2. $f(x) = x^4 - 8x^3 + 24x^2 - 32x + 16$ on the interval [0, 3].

(b) Implement in code the fixed-point iterations for the problems you designed and solve the root finding problem up to accuracy of $|f(x)| \leq ftol = 10^{-12}$. For each of these problems compute the actual rate of convergence of the iteration.

(c) Compare and describe any differences you observe between the two formulations of $g(x)$ you chose.

3. Newton's method (40 points)

(a) Implement in code Newton's method to solve the following problem:

$$f(x) = e^{6x} + 1.441e^{2x} - 2.079e^{4x} - 0.333 = 0, x \in [-5, 5].$$

with $|f(x)| \leq ftol = 10^{-12}$.

(b) Run your code using the following 3 initial guesses: $x_0 = -5, 2, 5$. Summarize your numerical results.

(c) Numerically estimate the convergence rate for each of your computer runs. Analyze the results and discuss the advantages and disadvantages of this algorithm.

4. Bonus (10 points)

(a) Run your Newton code on the second function in problem 1 (Lennard-Jones), using initial values of $x_0 = 0.1, 1, 2$.

(b) Analyze the results. What patterns do you observe? How does the algorithm behave? Justify all conclusions.