CS/MATH 375, Fall 2024 — HOMEWORK # 3
Due : Sept. 13th at 10:00 pm on Canvas

Instructions

- **Report:** In general, your report needs to read coherently. That is, start off by answering question 1. Fully answer the question, and provide all the information needed to understand your answer. If Matlab code or output is part of the question, include that code or output (e.g., screenshot) alongside your narrative answer. If discussion is required for a question, include that. Overall, your report is your narrative explanation of what was done, your answers to the specific questions, and how you arrived at your answers. *Your report should include your Matlab scripts, code output, and any figures.*

- **What to hand in:** Submission must be one **single PDF** document, containing your entire report, submitted on Canvas.

- **Partners:** You are allowed to (even encouraged) to **work in pairs**. If you work with a partner, only one member of the group should need to submit a report. On Canvas, both partners should join a group (numbered 1 through 15). Then either member can upload the report for the entire group. Groups of more than 2 students are not allowed.

- **Typesetting:** If you write your answers by hand, then make sure that your handwriting is readable. Otherwise, I cannot grade it.

- **Plots:** All plots/figures in the report must be generated in Matlab or Python and not hand drawn (unless otherwise specified in the homework question).

  In general, make sure to (1) title figures, (2) label both axes, (3) make the curves nice and thick to be easily readable, and (4) include a legend for the plotted data sets. The font-size of all text in your figures must be large and easily readable.

Reading: Read Chapter 1 from the book.

Solving nonlinear equations

1. **Programming:** In this problem you will implement the 3 basic rootfinding algorithms discussed in class. For all three methods, your function should accept as inputs a function name/handle, two numbers which define the initial bracket interval, a tolerance (`tol`) and an integer (`Nmax`). Each function should return an array consisting of the iterates (i.e., the guesses of the root). That is, return the array $[x_1, x_2, x_3, ..., x_n]$, where $x_k \to$ root. The first entry x_1 is the initial root guess, and the last entry x_n is the final (and most accurate) root guess. As an example:

```
function x_arr = my_bisection(fnc, a, b, tol, Nmax)
```

Your methods should use three checks for the stopping criteria:

$$(|\Delta x| < \texttt{tol}) \text{ or } (|\texttt{fnc}(x_k)| < \epsilon_m) \text{ or } (n > \texttt{Nmax}).$$

Here, n is the number of iterates that your function has produced so far, `fnc` is the name of the function you are finding roots of, ϵ_m is machine epsilon, and Δx will be a bit different for each method.

(a) **Bisection:** Write a function that implements the bisection method. At each step, use the midpoint of your interval to generate the guess x_n. For the quantity Δx, use the length of your interval (bracket) at that iteration.

(b) **Secant:** Write a function that implements the Secant method. At the first step, use $x_0 = \texttt{a}$ and $x_1 = \texttt{b}$ as your first two guesses. At each step, calculate $\Delta x = x_k - x_{k-1}$

(c) **Newton's:** Write a function that implements Newton's method. Be careful: Newton's method will need to have a structure like

```
function x_arr = my_newton(fnc, df,a, tol, Nmax)
```

Here `fnc` is the name of the function which evaluates $f(x)$ and `df` is the name of the function which evaluates $f'(x)$. Also, we only need to specify `a` as an initial guess, not a complete interval. At each successive step, calculate $\Delta x = x_k - x_{k-1}$.

2. **Finding roots:** It is a very common strategy (used by many calculators) to calculate radicals of real numbers using Newton's method (or similar methods). For example, the problem of calculating

$$x = \sqrt[p]{a},$$

can be reformulated as finding the roots of the function

$$f(x) = x^p - a.$$

In this problem, we're going to use the code from Problem 1 to calculate fifth roots.

(a) Plot $f(x)$ for $p = 5$ and $a = 32$. How many zeros are there? Are they simple or repeated roots? Suggest a good interval (bracket) to begin a root finding algorithm.

(b) Use each of your three root finding algorithms on the above function. Use an initial interval of $[1, 4]$ for the first two methods and an initial guess of 4 for Newton's Method. Set your tolerance as $\texttt{tol}= 10^{-10}$. Make `Nmax` large enough that the methods stop due to one of the other convergence criteria (i.e. `Nmax` is just so that you don't get caught in an infinite loop). For all three methods, report what the root computed (x_n), the value of the function at this root ($f(x_n)$), and how many iterations the method took (n).

3. **Slow Newton's Method:** From class, we know that Newton's method sometimes converges quadratically, sometimes linearly, and sometimes not all. For this problem, you will use your Newton code to find one of the roots of $f(x) = 8x(x - 1/2)^3$.

   (a) Find the positive root of the given function $f(x)$ using your Newton's method code. Report your initial guess, the computed approximate root (x_n), the value of the function at this root ($f(x_n)$), and how many iterations the method took (n).

   (b) Report the computed iterates x_k, and the ratio of errors
   $$\frac{e_{k+1}}{e_k^2}$$
   for the final 10 (or so) iterates. Does this ration appear to be bounded by a constant $C < 1$? Why or why not?

   (c) Repeat the previous part, but report the ratio of errors
   $$\frac{e_{k+1}}{e_k}$$
   for the final 10 iterates. Is this ratio bounded by a constant $C < 1$? Discuss.

4. **Orders of Convergence:** Consider the function
   $$f(x) = e^{\sin^3(x)} + x^6 - 2x^4 - x^3 - 1$$
   on the interval $[-2, 2]$. This function has three roots. One root is $r = 0$, the others are difficult to find exact formulas for.

   (a) Use Matlab's built in `fzero` function to find the other two roots. Use a tolerance on the input x of 3×10^{-16}. Report the commands you use to set the options for `fzero`, the initial guess/bracket you use, and the returned value for each root. For the rest of this problem treat your answers to this part (and zero) as the "true" roots.

   (b) Now use your three root-finding algorithms from problem 1 on the given function.

   Report your results by giving three different tables for each root, one for each method (nine total). Each table should have four columns. The first column lists the iteration number, the second lists the iterate x_k, the third lists the error e_k and the fourth lists
   $$\frac{e_{k+1}}{e_k^r}.$$

Choose an appropriate value of r for the three methods. The value of r might be different for each method and/or root.

What is the rate of convergence that you observe for each method? Explain your reasoning for your observed convergence rate. Remember that numerical data may be a bit noisy.