Homework 3

(Due: April 11, 2025)

This homework covers numerical solution of equations, linear matrix algebra, and modeling of small engineering systems with matrices and NumPy.

Question 1: 10 points. Figure 1 plots the function

$$f(x) = \left[\frac{x}{2} - \frac{1}{x} \right]^3 + \left[\frac{x}{2} - \frac{1}{x} \right] - 30. \tag{1}$$

over the range $[-5, 8]$.

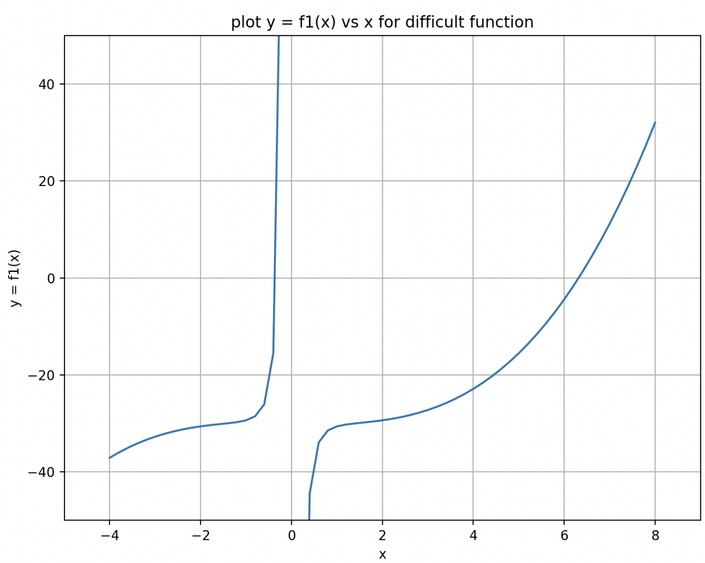

Figure 1: Plot y = f(x) vs x.

Theoretical considerations indicate that equation 1 has two real roots:

$$[r_1, r_2] = \frac{6}{2} \pm \frac{\sqrt{44}}{2} \tag{2}$$

Now suppose that we have Figure 1, and can see that the two roots lie in the intervals $[-1, 0]$ and $[6, 8]$, but for some reason don't know about equation 2.

Write a Python program that:

1. Uses the **method of bisection** to compute the lower and upper roots to equation

2. Computes the roots with the **method of Newton Raphson** iteration.

3. Investigates behavior of **Newton Raphson iteration** when we seek solutions to the lower root and the starting value x_o = -2.

Briefly discuss the strengths and weaknesses of bisection and newton raphson algorithms in terms of reliablity and efficiency of computations.

Question 2: 10 points.

This question covers numerical solutions to the root of the equation:

$$f(x) = 1 - e^{-(x-1)^2} = 0. \tag{3}$$

at x = 1. Figure 2 plots f(x) over the range $[-2, 4]$.

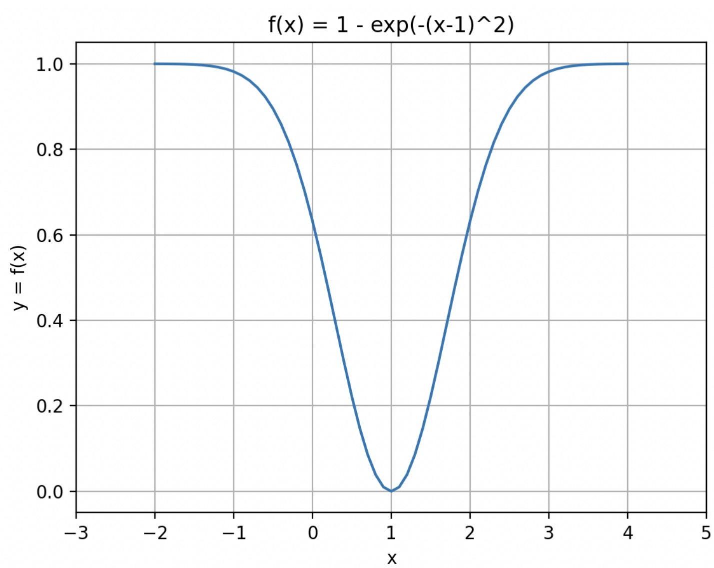

Figure 2: Plot y = f(x) vs x.

Write a Python program to plot equation 3 over the range [-2, 4], and then demonstrate that while the **method of Newton Raphson** struggles to find value(s) of x for which f(x) = 0 (why?), the **method of Modified Newton Raphson** computes the same roots with ease.

Question 3: 10 points. Figure 3 shows an elastic column fixed at its base and pinned at the top. The critical buckling load, P_{cr}, corresponds to the first positive solution to

$$\lambda = \tan[\lambda] \tag{4}$$

where $\lambda = L \cdot \sqrt{\frac{P_{cr}}{EI}}$.

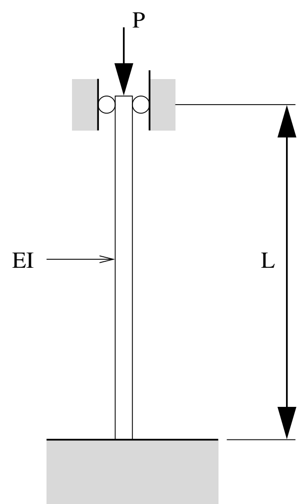

Figure 3: Elastic column carrying an axial load.

Use Python to create a single plot of y_1 = x and y_2 = tan(x) – the intersection of contours on this plot should give you an approximate range within which an accurate solution exists. Use the **method of bisection** to compute the numerical solution to equation 4. Print the buckling load corresponding to your solution?

Question 4: 10 points. Consider the family of matrix equations AX = B defined by

$$\begin{bmatrix} 1 & 2 & -3 \\ 3 & -1 & 5 \\ 4 & 1 & a^2 - 14 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 4 \\ 2 \\ a+2 \end{bmatrix}. \tag{5}$$

Determine the values of 'a' for which matrix A will be singular. Then,

1. Develop a program that uses NumPy to store matrices A and B, and then systematically evaluates the determinant and rank of A, and the rank of augmented matrix [A | B] for the values of 'a' that make A singular.

1. Develop a second program that uses SymPy to store matrices A and B symbolically, and then computes symbolic solution to the matrix equations 5. For the values of 'a' that make A singular, evaluate the determinant and rank of A, and the rank of augmented matrix [A | B].

You should find that the Python module SymPy is considerably more powerful than its numerical counterpart NumPy.

Question 5: 10 points. In the design of highway bridge structures and crane structures, engineers are often required to compute the maximum and minimum member forces and support reactions due to a variety of loading conditions.

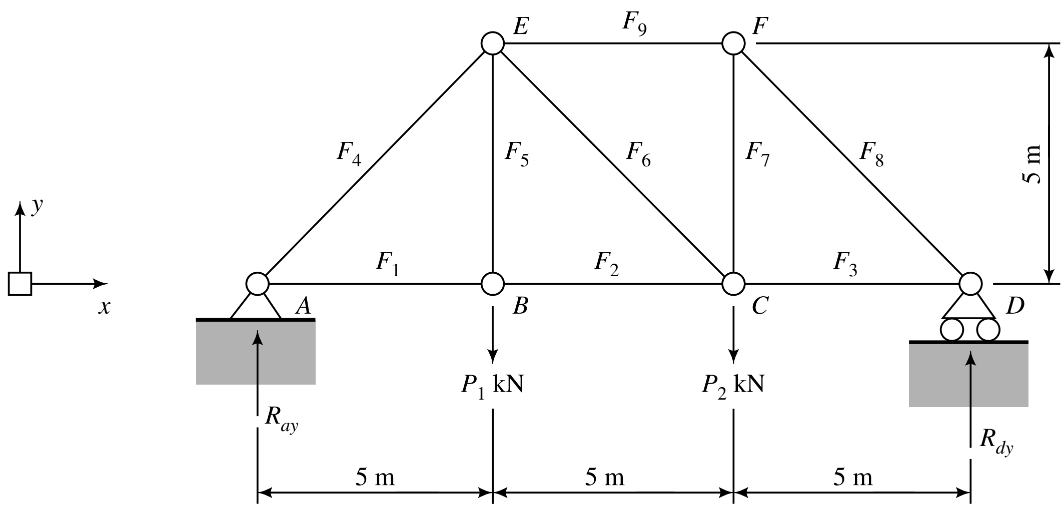

Figure 4: Front elevation of pin-jointed bridge truss.

Figure 4 shows a nine bar pin-jointed bridge truss carrying vertical loads P_1 kN and P_2 kN at joints `B` and `C`. The symbols F_1, F_2, $\cdots$ F_9 represent the axial forces in truss members 1 through 9, and R_{ay} and R_{dy} are the support reactions at joints `A` and `D`. You can also include a horizontal reaction force at A, R_{ax}, but its value will be zero.

Write down the equations of equilibrium for joints `A` through `F` and put the equations in matrix form. Now suppose that a heavy load moves across the bridge and that, for engineering purposes, it can be represented by the sequence of external load vectors

$$\begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} 100 \\ 0 \end{bmatrix}, \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} 100 \\ 100 \end{bmatrix}, \begin{bmatrix} P_1 \\ P_2 \end{bmatrix} = \begin{bmatrix} 50 \\ 100 \end{bmatrix} \tag{6}$$

Develop a Python program that will solve the matrix equations for each of the external load conditions, and compute and print the minimum and maximum support reactions at nodes A and D, and axial forces in each of the truss members.

Hint: See the folder python-code.d/applications/structures/ for examples of similar problems and their solutions.