NE336 Assignment 2 Due Oct 11th

Instructions

- All programming problems should be completed as separate .py files. The first line in each file should contain a comment with your complete name and student ID.
- Your py file should include : comments and test cases mentioned in the question.
- An online dropbox will be available for your submissions on learn. This dropbox will only be available until the 11:59 pm of the due date. Make sure your submission is received by this time since *no exceptions will be made.*

Questions

Systems of Equations

Note : You only need to solve 1 out of 3 questions in this section. You may choose which.

Solving 2 correctly will get you a 5% bonus on this assignment grade and solving all 3 correctly will get you up to 10%.

You still have to answer the question on the last page, so one question from this section and the one question for ODEs.

Question 1.

Solve the following system of equations

$$\begin{aligned} 2x_1 - 6x_2 - x_3 &= -38 \\ -3x_1 - x_2 + 7x_3 &= -34 \\ -8x_1 + x_2 - 2x_3 &= -20 \end{aligned}$$

using

i) Gauss Seidel, writing only the first two iterations solved by hand (no code required). Start with a guess of zeros. If necessary, rearrange the equations to achieve convergence. Find the error at each iteration.

ii) LU decomposition by one of the python builtin methods.

iii) Python `linalg.solve` method.

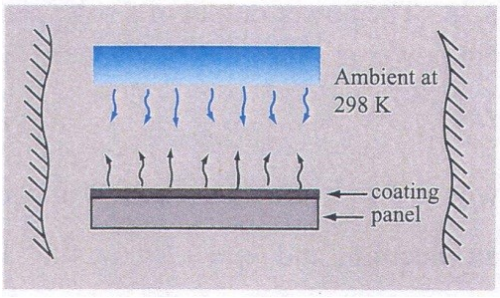
iv) the inverse of A.

Print the value of x (the solution vector) from each method and comment on the advantages or disadvantages of the method.

Submit in a file called `systems_question1.py`.

Question 2

A coating on a panel surface is cured by radiant energy from a heater. The temperature of the coating is determined by radioactive and convective heat transfer processes. If the radiation is treated as diffuse and gray, the following nonlinear system of simultaneous equations determine the unknowns:



$$
\begin{aligned}
5.67 \times 10^{-8} T_c^4 + 17.41 T_c - J_c &= 5188.18 \\
J_c - 0.71 J_h + 7.46 T_c &= 2352.71 \\
5.67 \times 10^{-8} T_h^4 + 1.865 T_h - J_h &= 2250 \\
J_h - 0.71 J_c + 7.46 T_h &= 11093
\end{aligned}
$$

Solve for J_h,T_h, J_c, T_c using:

a) The built-in function `fsolve`.

b) Newton's method (tol=1e-4%).

c) Newton's method but evaluate the Jacobian matrix numerically using centered finite difference (tol=1e-4%).

Use the following initial values: $T_h = T_c = 298K$,$J_c = 3000W/m^2$, and $J_h = 5000W/m^2$. Compare and discuss the accuracy of the solution, the computational time needed by each method (use `perf_counter` from the `time` module) and the number of iterations required.

Submit in a file called `systems_question2.py`.

Question 3

To infer the surface shape of an object from images taken of a surface from three different directions, one needs to solve the following set of equations:

$$
\begin{bmatrix} 0.2425 & 0 & -0.9701 \\ 0 & 0.2425 & -0.9701 \\ -0.2357 & -0.2357 & -0.9428 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 247 \\ 248 \\ 239 \end{bmatrix}
$$

The right hand side values are the light intensities from the middle of the images, while the coefficient matrix is dependent on the light source directions with respect to the camera. The unknowns are the incident intensities that will determine the shape of the object.

Implement the Gauss-Seidel method to find the values of x_1, x_2, and x_3. Assume an initial guess of 10 for x_1, x_2 and x_3. For this system, can you ensure convergence while using GS?

Submit in a file called `systems_question3.py`.

Ordinary differential equations

Consider the following first order differential equation:

$$\frac{\mathrm{d}y}{\mathrm{d}t} = (1 + 4t)\sqrt{y}$$

Solve this problem over the interval from $t = 0$ to $t = 1$ where $y(t = 0) = 1$ using the following approaches. For numerical methods, use a step size of $\Delta t = h = 0.25$.

a) Analytical.

b) Euler's method.

c) Heun's method without iterations.

d) Ralston's method.

e) Range-Kutta Fourth order.

f) `solve_ivp` solver in scipy.

Display all of your results on the same graph (plot y vs t from all methods).

Note: for only the analytical solution, plot for 50 points between 0 and 1 so that you get a continuous curve and display the rest of the methods as markers (no lines) on graph.

Submit in a file called `odes.py`.