MATH 578 LAB ASSIGNMENT 3

DUE SUNDAY DECEMBER 6, 22:00 EST

1. Implement forward and backward Euler methods (for vector valued functions), where the backward Euler code uses Newton's iteration as a nonlinear solver. In each step of the backward Euler, start the Newton iteration with the initial guess given by a step of the forward Euler method, and stop the Newton iteration when the distance between two successive iterates falls within some specified tolerance (e.g., equal to 10^{-6}).
2. Use both the forward and backward Euler methods on the three model problems $y' = \lambda y$, $y(0) = 1$ where (i) $\lambda = -23$, (ii) $\lambda = 1$ and also (iii)
$$y' = \begin{pmatrix} -1 & 0 \\ 0 & -100 \end{pmatrix} y, \qquad y(0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}.$$
using time steps $h = 0.1, 0.05, 0.02, 0.01, 0.005, \ldots$.
   For each method for each problem
   (a) Plot a graph of $\mathtt{log}\|y(t_n) - y_n\|$ against $\mathtt{log}h$, at the fixed time $t = 2$ where n satisfies $nh = 2$, that is we are computing many numerical solutions, each for a different fixed step-size h, for $t \in [0, 2]$ and comparing the error between the exact and numerical solution at the final time as a function of h. Explain the results.
   (b) To compare the performance of the methods more fairly, we should consider the extra work required when computing with the backward Euler method. So, repeat the first part, but this time plot $\mathtt{log}\|y(t_n) - y_n\|$ against N where N is the total number of function evaluations in the integration. For the forward Euler method $N = n$, but for the backward Euler method it will be the total number of Newton iteration in the first n steps. Which method is most efficient for h small? Is there a difference for h large?
   (c) For one fixed h plot graphs of $\mathtt{log}\|y(t_n) - y_n\|$ against t_n. Ensure that the interval of time integration is sufficiently long to see the qualitative difference in the behaviour between cases (i) and (ii). Explain this difference. Which method performs best? Does changing h change the relative differences between the methods? Plot the overimposed graphs of the computed solutions and the exact solution (for case (iii) one can separate the components of y to get 2D plots).
3. Using the forward Euler code as a template, implement Runge's second order method

$$\begin{array}{c|cc} 0 & & \\ 1/2 & 1/2 & \\ \hline & 0 & 1 \end{array}$$

Date: Fall 2020.

and the four stage method (sometimes called "The Runge-Kutta method"):

$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 1/3 & 1/3 & 1/6 \end{array}$$

Compare the performance of these methods with the forward and backward Euler methods on the three differential equations considered in the previous exercise (use the same comparisons as in (a), (b) and (c) from that exercise).

4. In this exercise, we will study the motion of the five outer planets relative to the sun using different numerical integrators. The necessary data is given in file `outer.txt`. In the data file the units are chosen so that the sun has mass 1, the distances are in astronomical units (1a.u.=149,597,870km), and times in earth days. The sun and the inner planets are to be collectively treated as one object. When you copy and run `outer.txt` in Python, the data will be loaded to three variables `g`, `m`, and `x`; where `g` holds the value of the universal gravitational constant, `m` is an array consisting of the masses of the 6 objects, and `x` is an array with initial positions and velocities of the objects, corresponding to September 5, 1994 at UTC0:00. Further comments can be found in the file `outer.txt`.

   Implement the following numerical integrators and use them to study the gravitational 6 body problem with the data given in `outer.txt`. For each of the following cases, run the simulation for at least one to few thousand years (in simulated time of course), and describe what happens to the planets during the simulations. Adjust the stepsize of the method and the simulation time accordingly. Comment on the performance of the methods.
   (a) Explicit Euler with stepsize 10 days
   (b) Implicit Euler with stepsize 10 days
   (c) Your favorite Runge-Kutta method of order at least 2, with stepsize 30 days
   (d) Symplectic Euler with stepsize 100 days
   (e) Störmer-Verlet with stepsize 200 days

Homework policy

Please submit a link to your Jupyter notebook on GitHub through the `Assignments` tab on MyCourses.

- Include theory and commentaries in your Jupyter notebook, with Markdown cells.
- Notebooks without sufficient commentaries will not be accepted.
- You can update your notebook up to the time limit, but please do not commit any overdue updates.

You are welcome to consult each other provided (1) you list all people and sources who aided you, or whom you aided and (2) you write-up the solutions independently, in your own language. If you seek help from other people, you should be seeking general advice, not specific solutions, and must disclose this help. This applies especially to internet fora such as `MathStackExchange`.

Similarly, if you consult books and papers outside your notes, you should be looking for better understanding of or different points of view on the material, not solutions to the problems.