AMATH 481/581 - Autumn 2022

Homework #1

Submissions accepted until 11:59 PM (PT) Wednesday, October 19, 2022

To submit this assignment, upload your main homework file (.m, .py, or .ipynb) to Gradescope. Additionally you may upload a .pdf you create to demonstrate mastery of one or both presentation skills.

1. Consider the ODE

$$\frac{dy}{dt} = -3y(t)\sin(t), \qquad y(t=0) = \frac{\pi}{\sqrt{2}},$$

which has the exact solution $y_{\text{true}}(t) = \pi e^{3(\cos t - 1)}/\sqrt{2}$ (you *can* verify that on your own, perhaps using Mathematica if you'd like). In this problem you will implement the Forward Euler and Heun's methods for this ODE for $t \in [0, 5]$ to test the error as a function of Δt.

   (a) Solve the ODE numerically using the Forward-Euler method,

$$y_{n+1} = y_n + \Delta t f(t_n, y_n), \tag{1}$$

   where $\Delta t = 2^{-2}, 2^{-3}, \ldots, 2^{-8}$. For each of these Δt values, calculate the error $E = |y_{\text{true}}(5) - y_N|$, where N corresponds to $t_N = 5$. Plot $\log(\Delta t)$ on the x-axis and $\log(E)$ (loglog plot) on the y axis. Use `polyfit` to find the slope of the best fit line through the data. The slope of this line is the order of accuracy (global Error) for the Forward Euler Method.

   (b) Solve the ODE numerically using Heun's method

$$y_{n+1} = y_n + \frac{\Delta t}{2}\left(f(t_n, y_n) + f(t_{n+1}, y_n + \Delta t f(t_n, y_n))\right),$$

   where $\Delta t = 2^{-2}, 2^{-3}, \ldots, 2^{-8}$. For each of these Δt values, calculate the error $E = |y_{\text{true}}(5) - y_N|$, where N corresponds to $t_N = 5$. Plot $\log(\Delta t)$ on the x-axis and $\log(E)$ (loglog plot) on the y axis. Use `polyfit` to find the slope of the best fit line through the data. The slope of this line is the order of accuracy (global Error) for Heun's method.

(c) Repeat the steps in parts (a) and (b) using the Adams predictor-corrector method demonstrated in class (1.1.29 in the notes),

$$y_{n+1}^P = y_n + \frac{\Delta t}{2}\left(3f(t_n, y_n) - f(t_{n-1}, y_{n-1})\right),$$
$$y_{n+1} = y_n + \frac{\Delta t}{2}\left(f(t_{n+1}, y_{n+1}^P) + f(t_n, y_n)\right).$$

We need a second initial condition to use this multistep method. Since the method is second-order accurate, we would like to use a second-order accurate approximation to y_1. Use RK2,

$$y_{n+1} = y_n + \Delta t f\left(t_n + \frac{\Delta t}{2}, y_n + \frac{\Delta t}{2} f(t_n, y_n)\right),$$

to generate y_1 from y_0.

Deliverables: For Forward Euler, save your last numerical solution, (with $\Delta t = 2^{-8}$), as a column vector named `A1`. Save the error values in a row vector with seven components to the variable `A2`. Save the slope of the best-fit line to the variable `A3`.

Do the same for Heun's method, and save the variables to `A4`, `A5`, and `A6` respectively.

Do the same for the predictor-corrector method, and save the variables to `A7`, `A8`, and `A9` respectively.

For presentation mastery: To demonstrate mastery of creating 2D plots, plot $\log(E)$ vs. $\log(\Delta t)$ for all three methods on the same plot. Use a `loglog` plot so that the tickmarks on the axes are correct. Include lines with slope n where n is the order of accuracy for each of the methods. These lines should match up with the data and should clearly show the order of accuracy for each of the methods. Your figure should have (at least) legends, clear plot markers, and clearly labeled axes and title.

2. Consider the van der Pol oscillator,

$$\frac{d^2y(t)}{dt^2} + \epsilon\left(y^2(t) - 1\right)\frac{dy}{dt} + y(t) = 0,$$

where ϵ is a parameter indicating the strength of the nonlinearity in the system.

(a) With $\epsilon = 0.1$, $y(t = 0) = \sqrt{3}$, and $dy/dt(t = 0) = 1$, solve the equation for $t \in [0, 32]$ with $\Delta t = 0.5$ using a built-in method for RK45 (`ode45` in MATLAB and `scipy.integrate.solve_ivp` in python). Repeat the process for the same initial conditions with $\epsilon = 1$ and $\epsilon = 20$.

Deliverables: Save the numerical solution `y` for the three-different ϵ used as a matrix with 3 columns in `A10`.

(b) Here we will explore how the RK45 algorithm uses adaptive step sizes to reach certain error goals. To do that, we need to tell the software what our *absolute* and *relative* error tolerance goals are. When using `ode45` in MATLAB, error tolerance is set using, e.g.,

```
tol = 1e-4;
options = odeset('AbsTol', tol, 'reltol', tol);
[T, Y] = ode45(dydt, tspan, y0, options);
```

In python, using `scipy.integrate.solve_ivp`, use e.g.,

```
tol = 1e-4
sol = scipy.integrate.solve(dydt, tspan, y0, atol=tol, rtol=tol)
T = sol.t
Y = sol.y
```

With $\epsilon = 1$, $y(t = 0) = 2$, and $dy/dt(t = 0) = \pi^2$, solve the IVP for $t \in [0, 32]$ using a built-in method for RK45. Here the step size is not specified and you should not specify a step size in the built-in method (let it decide the step size). Use the `diff` or `numpy.diff` and `mean` or `numpy.mean` on the time output `T` to calculate the average step size Δt needed to solve the problem for each of the following error tolerance values, 10^{-4}, $10^{-5}, \ldots, 10^{-10}$. Letting Δt_{avg} be the average step size used for each error tolerance, plot $\log(\Delta t_{\text{avg}})$ on the x axis and $\log(\texttt{tol})$ on the y axis. Use `polyfit` to find the slope of the best-fit line through the data. This slope gives us an idea of how the necessary average Δt changes as we refine our error tolerance. Repeat this process with RK23 (`ode23` in MATLAB, `scipy.integrate.solve_ivp(..., method='RK23')` in python) and then with an implicit multi-step variable-order method (`ode15s` in MATLAB, `scipy.integrate.solve_ivp(..., method='BDF')` in python).

Deliverables: Save the slope of the best-fit line for RK45, RK23, and the implicit multi-step method to the variables `A11-A13` respectively.

3. (More background about this system will be given in class in Week 2). The Fitzhugh (FH) model is an ODE model used for simulating the dynamics of a neuron upon excitation (a voltage spike input). The FH model simulates the dynamics of the voltage (v) and the refractory variable (w) in the system. The refractory variable is included to model the recovery (leading to reset) of the neuron after a voltage spike.

   To explore the interaction between two neurons, implement two Fitzhugh neurons coupled via linear coupling:

$$\begin{aligned}
\frac{dv_1}{dt} &= -v_1^3 + (1+a_1)v_1^2 - a_1 v_1 - w_1 + I + d_{12}v_2, \\
\frac{dw_1}{dt} &= bv_1 - cw_1, \\
\frac{dv_2}{dt} &= -v_2^3 + (1+a_2)v_2^2 - a_2 v_2 - w_2 + I + d_{21}v_1, \\
\frac{dw_2}{dt} &= bv_2 - cw_2,
\end{aligned}$$

   with parameters $a_1 = 0.05,\ a_2 = 0.25,\ b = c = 0.1$, and $I = 0.1$. The different indices, e.g., v_1 and v_2 represent the voltage of neuron 1 and neuron 2 respectively. I represents the input current. The constant d_{12} and d_{21} represent the interaction between the two neurons. Start the simulations with the initial condition of $(v_1(0), v_2(0)) = (0.1, 0.1)$ and $(w_1(0), w_2(0)) = (0, 0)$. Use a variable-order implicit multi-step method, `ode15s` in MATLAB or `scipy.integrate.solve_ivp(..., method='BDF')` in python, to solve the equations. You will explore several different choices for the interaction parameters, d_{12} and d_{21}.

   Deliverables: Solve the ODEs with five sets of different interaction parameters: $(d_{12}, d_{21}) = (0, 0),\ (0, 0.2),\ (-0.1, 0.2),\ (-0.3, 0.2)$, and $(-0.5, 0.2)$. For each interaction, solve the system of ODEs for $t \in [0, 100]$ with $\Delta t = 0.5$. Save the computed result (the numerically computed values of the functions $(v_1,\ v_2,\ w_1,\ w_2)$ – in that order) as a 201×4 matrix. Save the five different matrices created (for the different choices of the interaction parameters) to the variables `A14-A18`.

For presentation mastery: To demonstrate mastery of discussing results from a physical perspective, plot the solutions for each of the five different sets of interaction parameters and discuss how the interaction parameters play a role in the solution. You will not be *strictly graded* on the plot you create, but your discussion should be clear from the plots you share. Your discussion should be complete, including discussing the difference between different sign and magnitudes of the parameters. If your explanation would be improved by considering other sets of interaction parameters, you may do that.