Homework 1

AMATH 481/581

Problem 1 - Relevant to the Written Report

Consider the IVP

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -4x\sin(t) \text{ with } x(0) = 1$$

The exact solution to this ODE is $x(t) = e^{4(\cos(t)-1)}$. In this problem, you will approximate the solution to this ODE from $t = 0$ to $t = 8$ using various values of Δt and three different methods: Forward Euler, Heun's method and RK2.

(a) Solve this ODE with the forward Euler method using $\Delta t = 2^{-5}$. Calculate the local truncation error $\epsilon_1 = |x(t_1) - x_1|$ and store the result in a variable named `A1`. Calculate the global error $E_N = |x(t_N) - x_N|$ (where N is the total number of steps required) and store the result in a variable named `A2`. (Note the absolute values on each error!)

Repeat this process with $\Delta t = 2^{-6}$. Store the local truncation error ϵ_1 in a variable named `A3` and the global error E_N in a variable named `A4`.

For the written report: Solve the IVP with forward Euler using $\Delta t = 2^{-5}$, $\Delta t = 2^{-6}$, $\Delta t = 2^{-7}$, $\Delta t = 2^{-8}$, $\Delta t = 2^{-9}$, $\Delta t = 2^{-10}$ and $\Delta t = 2^{-11}$ and calculate the global error for each value of Δt. Calculate the logarithms of E_N and Δt (i.e., $\ln(E_N)$ and $\ln(\Delta t)$). Find the best fit line through this data and plot it along with the data on a log-log plot.

(b) Solve this ODE with Heun's method using $\Delta t = 2^{-5}$. Calculate the local truncation error $\epsilon_1 = |x(t_1) - x_1|$ and store the result in a variable named `A5`. Calculate the global error $E_N = |x(t_N) - x_N|$ (where N is the total number of steps required) and store the result in a variable named `A6`. (Note the absolute values on each error!)

Repeat this process with $\Delta t = 2^{-6}$. Store the local truncation error ϵ_1 in a variable named `A7` and the global error E_N in a variable named `A8`.

For the written report: Solve the IVP with Heun's method using $\Delta t = 2^{-5}$, $\Delta t = 2^{-6}$, $\Delta t = 2^{-7}$, $\Delta t = 2^{-8}$, $\Delta t = 2^{-9}$, $\Delta t = 2^{-10}$ and $\Delta t = 2^{-11}$ and calculate the global error for each value of Δt. Calculate the logarithms of E_N

and Δt (i.e., $\ln(E_N)$ and $\ln(\Delta t)$). Find the best fit line through this data and plot it along with the data on a log-log plot.

(c) Solve this ODE with RK2 using $\Delta t = 2^{-5}$. Calculate the local truncation error $\epsilon_1 = |x(t_1) - x_1|$ and store the result in a variable named `A9`. Calculate the global error $E_N = |x(t_N) - x_N|$ (where N is the total number of steps required) and store the result in a variable named `A10`. (Note the absolute values on each error!)

Repeat this process with $\Delta t = 2^{-6}$. Store the local truncation error ϵ_1 in a variable named `A11` and the global error E_N in a variable named `A12`.

For the written report: Solve the IVP with RK2 using $\Delta t = 2^{-5}$, $\Delta t = 2^{-6}$, $\Delta t = 2^{-7}$, $\Delta t = 2^{-8}$, $\Delta t = 2^{-9}$, $\Delta t = 2^{-10}$ and $\Delta t = 2^{-11}$ and calculate the global error for each value of Δt. Calculate the logarithms of E_N and Δt (i.e., $\ln(E_N)$ and $\ln(\Delta t)$). Find the best fit line through this data and plot it along with the data on a log-log plot.

For the written report: This approach is often used to test the order of accuracy for numerical methods. How do the slopes of the best fit lines relate to the order for each method? Explain why you should expect to see such a relationship. (The derivation is not particularly complicated, but should involve a few lines of math.)

Problem 2

Consider the IVP

$$\frac{\mathrm{d}x}{\mathrm{d}t} = 8\sin(x) \quad \text{with} \quad x(0) = \frac{\pi}{4}$$

The exact solution to this problem is

$$x(t) = 2\arctan\left(\frac{e^{8t}}{1+\sqrt{2}}\right)$$

In this problem, you will approximate the solution to this ODE using the predictor-corrector method discussed in class (1.1.29 in the notes). In particular, we will use the method defined by

$$\begin{aligned} x_{k+1}^P &= x_k + \frac{\Delta t}{2}\left[3f(x_k, t_k) - f(x_{k-1}, t_{k-1})\right] \\ x_{k+1} &= x_k + \frac{\Delta t}{2}\left[f(x_{k+1}^P, t_{k+1}) + f(x_k, t_k)\right] \end{aligned}$$

We will use $t_0 = 0$, $t_N = 2$ and various values of Δt. Because the Adams-Bashforth method is a two step method, we need to determine x_1 with an alternative method. In particular, we will calculate x_1 using the RK2 method:

$$x_1 = x_0 + \Delta t \cdot f\left(x_0 + \frac{\Delta t}{2} f(t_0, x_0),\ t_0 + \frac{\Delta t}{2}\right)$$

(a) Solve the ODE using the method outlined above using $\Delta t = 0.1$. Store your final approximation x_N in a variable named `A13` and store the global error $E_N = |x_N - x(t_N)|$ in a variable named `A14`.

(b) Solve the ODE using the same method but with $\Delta t = 0.01$. Store your final approximation x_N in a variable named `A15` and store the global error $E_N = |x_N - x(t_N)|$ in a variable named `A16`.

Problem 3

The FitzHugh-Nagumo model is a system of ordinary differential equations used to describe the excitation of a neuron membrane:

$$\begin{aligned}\frac{\mathrm{d}v}{\mathrm{d}t} &= v - \frac{1}{3}v^3 - w + I(t)\\ \frac{\mathrm{d}w}{\mathrm{d}t} &= \frac{a + v - bw}{\tau}\end{aligned}$$

In this model, $v(t)$ is the membrane voltage at time t and $w(t)$ is a variable representing the activity of several types of membrane channel proteins. The function $I(t)$ represents an external electrical current, and the parameters a, b and τ are constants controlling the channel protein activity.

In this problem, we will assume that $a = 0.7$, $b = 1$ and $\tau = 12$ and that

$$I(t) = \frac{1}{10}\left(5 + \sin\left(\frac{\pi t}{10}\right)\right)$$

We will also assume that the initial voltage is 0.1 and the initial channel activity is 1. That is,

$$v(0) = 0.1 \text{ and } w(0) = 1$$

We will solve from time $t_0 = 0$ to time $t_N = 100$.

(a) Use the builtin method for RK45 (either `ode45` in MATLAB or `scipy.integrate.solve_ivp` in python) to solve this system of ODEs. You should specify the initial and final time, but **not** Δt. However, you should specify the absolute and relative tolerance as 10^{-4}. If you are using MATLAB, you will also need to tell MATLAB not to add extra points to smooth out the graph (called "refining" the solution). Python does not refine by default. You can do this in MATLAB with the code

```
options = odeset('AbsTol', 1e-4, 'RelTol', 1e-4, 'Refine', 1);
[T, X] = ode45(f, tspan, x0, options);
```

and in python with the code

```
sol = scipy.integrate.solve_ivp(f, tspan, x0, atol=1e-4, rtol=1e-4)
T = sol.t
X = sol.y
```

You will find that the time steps are not evenly spaced. Calculate the average time step

$$\overline{\Delta t} = \frac{1}{N}\sum_{k=1}^{N}(t_k - t_{k-1})$$

(There are quite a few ways to calculate this value in code. It may help to simplify by hand first.)

Store the final approximation of the voltage v_N in a variable named `A17` and the average time step in a variable named `A18`.

(b) Repeat the previous part using an absolute and relative tolerance of 10^{-9}. Store the final approximation of the voltage v_N in a variable named `A19` and the average time step in a variable named `A20`.