AMATH 301 – Winter 2023
Homework 6

Due: 11:59pm, February 23, 2023.

Instructions for submitting:

- Coding problems are submitted to Gradescope as a python script/Jupyter notebook (`.py` or `.ipynb` file). (If you use Jupyter Notebook, remove all "magic": the lines that begin with %). You have **8 attempts** (separate submissions to Gradescope) for the coding problems.
- Writeup problems are submitted to Gradescope as a single `.pdf` file that contains text, plots, and code at the end of the file for reference. Put the problems in order and label each writeup problem. When you submit, **you must indicate which problem is which on Gradescope (including the code that belongs to the problem). Failure to do so will result in a 10% penalty.. All code used for each problem should be included at the end of that problem in your .pdf file and be marked as part of the problem on Gradescope. Failure to include code will result in a 25% penalty.**

Coding problems

1. This problem comes from *Topics in Mathematical Modeling* by (UW AMATH's own) K.K. Tung. Most snail species are *Dextral* (right-handed) in their shell pattern. *Sinistral* (left-handed) snails are exceedingly rare (this corresponds to the direction of the pattern on their shell). Let $p(t)$ be the ratio of dextral snails in the population of snails. $p = 1$ means that all snails are right handed and $p = 0$ means that all snails are left handed. A model equation for $p(t)$ can be

$$\frac{dp}{dt} = p(1-p)\left(p - \frac{1}{2}\right),$$

which has no left-right bias. We want to solve this equation for $p(0) = 0.9$ (in other words, 90% of snails are right handed and 10% are left handed). We want to solve for the snail population over the interval $0 \leq t \leq 10$ with a time step of $\Delta t = 0.5$.

   (a) Solve this ODE using Forward Euler. Create an array containing the solution at all of the t values to the variable `A1`.

   (b) Solve this ODE using Backward Euler. In order to solve using Backward Euler, you will need to solve a root-finding problem at each step. Use

`scipy.optimize.fsolve` to solve this root-finding problem. When you use these algorithms, you will need to input an initial guess for the root. You should use the same convention we used in class for the initial guess.

Save the solution array at the corresponding t values to the variable `A2`.

(c) Solve this ODE using the midpoint method. Save the solution array to the variable `A3`.

(d) Solve this ODE using RK4. Do not use a built-in integrator here: you should code up RK4 on your own. Save the solution array to the variable `A4`.

2. In this problem we will consider two partners, Ray and Jun, who are exploring their relationship. A model for Ray and Jun's feelings of affection towards eachother is

$$\begin{aligned} R'(t) &= aR(t) + J(t) \\ J'(t) &= -R(t) - aJ(t), \end{aligned}$$

where

$$\begin{aligned} R(t) &= \text{Ray's feelings for Jun at time t,} \\ J(t) &= \text{Jun's feelings for Ray at time t.} \end{aligned}$$

A positive R or J means positive feelings (endearment) and a negative feeling means negative feelings (animus). The constant $a > 0$ represents how strongly the two react to their own feelings. For example, if $a = 0$ then the two are not in touch with their own feelings. For large a, they care more about their feelings towards the other person than they care about the others' feelings towards them. In this problem we will use $a = 1/2$. To solve this ODE we now have to keep track of two variables. By defining

$$x(t) = \begin{pmatrix} R(t) \\ J(t) \end{pmatrix},$$

we now have an ODE for $x'(t)$, the vector. We will use the initial condition

$$x(0) = \begin{pmatrix} 2 \\ 1 \end{pmatrix}.$$

(a) We will first solve this system of ODEs using the built-in solver `scipy.integrate.solve_ivp`. Solve the system of ODEs for $0 \leq t \leq 20$ letting the algorithm determine the step size. Save the resulting solution array for $R(t)$ to the variable `A5` and the resulting solution array for $J(t)$ to the variable `A6`.

(b) We will use the solutions you just calculated as the "true" solution for calculating error. Save an array with 2 elements consisting of the last value of $R(t)$ and $J(t)$ to the variable `A7`. In other words, `A7` $= [R(20), J(20)]$.

(c) We will next solve this system of ODEs using the Forward-Euler method. Using the definition of x above the Forward-Euler method becomes

$$x_{k+1} = \begin{pmatrix} R_{k+1} \\ J_{k+1} \end{pmatrix} = \begin{pmatrix} R_k + \Delta t f_R(R_k, J_k) \\ J_k + \Delta t f_J(R_k, J_k) \end{pmatrix}$$

where

$$f_R(R, J) = aR + J,$$
$$f_J(R, J) = -R - aJ.$$

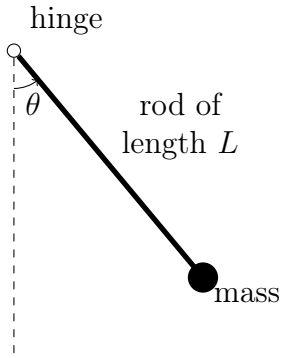
Alternatively, you can keep track of R_k and J_k separately and update both in the for loop.

Solve this system of ODEs for $0 \leq t \leq 20$ using $\Delta t = 0.1$. Save the resulting array for $R(t)$ to the variable `A8` and the resulting array for $J(t)$ to the variable `A9`.

(d) We are going to calculate the error for the Forward-Euler method. To do so, we need to calculate $R(20)$ and $J(20)$ from Forward-Euler and save it to a vector. Do this and save the result ($[R(20), J(20)]$) to the variable `A10`.

(e) To calculate the error for the Forward-Euler method with $\Delta t = 0.1$ we want to know *how far away our Forward-Euler solution is from the true solution at* $t = 20$. Since the answer at $t = 20$ is an array with 2 elements, we can't just take the absolute value of the difference; we need to use a norm! Calculate the 2-norm of the difference between `A10` and `A7`. Save the result to the variable `A11`.

3. Consider a pendulum made by connecting a mass to a rod of length L to a hinge. This scenario is described by the following cartoon.



The equation for motion of this pendulum is given by the nonlinear pendulum equation with damping,

$$\theta''(t) = -\frac{g}{L}\sin(\theta(t)) - \sigma\theta'(t). \tag{1}$$

Here $g = 9.8$ is the acceleration due to gravity, $L = 11$ is the length of the pendulum, and $\sigma = 0.12$ is a damping coefficient. Damping may come from friction at the hinge, friction between air and the pendulum, or other forces. The unknown that we must find is θ (theta), the angle of deflection of the pendulum from the vertical.

(a) By defining v (v for velocity) as the derivative of θ, $v(t) = \theta'(t)$, we can convert this single second-order ODE into a system of two first-order ODEs (you will practice this in the Activity). Doing so gives two ODEs,

$$\theta'(t) = v(t), \tag{2}$$
$$v'(t) = -\frac{g}{L}\sin(\theta(t)) - \sigma v(t). \tag{3}$$

We are going to solve this system of differential equations in terms of $\theta(t)$ and $v(t) = \theta'(t)$ over the time interval $0 \leq t \leq T$ for $T = 50$. In order to solve this system of differential equations we must define initial conditions (this is the initial setup of the system, then we let it go). We will use

$$\theta(t = 0) = \theta(0) = -\pi/8,$$
$$v(t = 0) = v(0) = -0.1.$$

In other words, the pendulum starts with angle $-\pi/8$ and we push it to the left with speed 0.1.

(b) We will solve this using `scipy.integrate.solve_ivp`. To do so, you need to create a function that has two inputs, `t, p`, where `p` represents an array of our *dependent variables*, θ and v. Call this function `odefun` and save the output of `odefun` evaluated at $t = 1$, $p = [2, 3]$ to the variable `A12`. The output should be an array.

(c) Now use `scipy.integrate.solve_ivp` to solve the initial-value problem for the nonlinear pendulum equation with damping. The output of the *dependent* variables will be a $2 \times N$ array, where N corresponds to the N times at which we get the solution. Save the $2 \times N$ array to the variable `A13`.

Writeup problems

1. This problem mirrors Coding problem 3, the pendulum. Here you will construct a phase portrait with $\theta(t)$ on the horizontal axis and $v(t)$ on the vertical axis. You should turn in the plot created in (a)-(e), the explanations in (f), and the code you used for this problem.

   (a) Use `meshgrid` to generate a grid of points between $-3\pi \leq \theta \leq 3\pi$ and $-3 \leq v \leq 3$ with 25 equally spaced points in both directions.

   (b) Before we used quiver plots to represent the slope defined by our ODE. For this phase portrait, since we have $\theta(t)$ on the horizontal axis and $v(t)$ on the vertical axis, arrows will represent

   $$\frac{dv}{d\theta} = \frac{dv/dt}{d\theta/dt}.$$

   Use the `quiver` function to draw a grid of arrows with components $(\theta'(t), v'(t))$, where $\theta'(t)$ and $v'(t)$ are given by the ODEs of the nonlinear damped pendulum system found in Coding Problem 3 (a). Use the same parameters as used there.

(c) Include labels for the axes. To make the θ symbol in python, you need to use `r'$\theta$'`.

(d) We now want to include trajectories in the phase portrait.

i. Use `scipy.integrate.solve_ivp` to solve the system with the following initial conditions.

- $\theta(0) = \pi,\ v(0) = 0.1$
- $\theta(0) = \pi,\ v(0) = -0.1$
- $\theta(0) = 2\pi,\ v(0) = -3$
- $\theta(0) = -2\pi,\ v(0) = 3$

In each case solve for $0 \leq t \leq 50$ with $\Delta t = 0.01$ (using `t_eval`). Add these solution trajectories to the phase portrait. These solution trajectories should flow with the arrows from `quiver`.

(e) Set the axes to display from $-3\pi \leq \theta \leq 3\pi$ and $-3 \leq v \leq 3$. You do not want to use equal axes here. You do not need to add a legend, because it is self-explanatory which trajectory belongs to which initial condition.

(f) We now want to explain what is happening physically. To do so, you should be thinking about a few key things. When you say "oscillating" or "going around": what specifically do you mean? There are two key distinctions here I want you to think about. First, the pendulum can *go all of the way around the hinge*, in other words, it can loop and loop around. Imagine if you hit it very very hard, this is what would happen. Second, the pendulum can *oscillate* around $\theta = 0$ (imagine if you hit the pendulum very lightly). Be careful with your wording.

i. As time increases, the solutions with the different initial conditions are all going to the same physical situation (as $t \to \infty$). Describe that physical scenario in terms of the pendulum. **Hint:** How are $\theta = 0,\ -2\pi$, and 2π related physically?

ii. Compare the two solutions with initial conditions $(\theta(0), v(0)) = (\pi, 0.1)$ and $(\theta(0), v(0)) = (\pi, -0.1)$. Describe what is happening to these two solutions. How do they start? What is their motion immediately after $t = 0$? How are the two solutions different?

iii. Describe what is physically happening to the pendulum with the initial conditions $(\theta, v) = (2\pi, -3)$ and $(\theta, v) = (-2\pi, 3)$.