COMPUTER SCIENCE AND ENGINEERING

Indian Institute of Technology Palakkad

CS5016: Computational Methods and Applications

15 Mar, 2023

***Coding Assignment 6***

***Ordinary Differential Equations***

1. Write a function that uses the *forward Euler method* to solve the ODE $x^{'}(t) = -2x(t)$, with initial condition $x(0) = 5$, in the interval $[0, 10]$, computes a polynomial that passes through the discrete solution points of the ODE. Visualize the solution for discretization step sizes $\{0.1, 0.5, 1, 2, 3\}$ along with the exact solution, all on the same figure.

2. Write a function that uses the *backward Euler method* to solve the ODE $x^{'}(t) = -2x(t)$, with initial condition $x(0) = 5$, in the interval $[0, 10]$, computes a polynomial that passes through the discrete solution points of the ODE. Visualize the solution for discretization step sizes $\{0.1, 0.5, 1, 2, 3\}$ along with the exact solution, all on the same figure.

3. A *simple gravity pendulum* is an idealized mathematical model of a real pendulum. It has a weight (or bob) on the end of a massless cord suspended from a pivot, without friction[1]. The ODE which represents the motion of a simple pendulum is

$$\frac{d^2(\theta(t))}{dt^2} + \frac{g}{L} \sin\theta(t) = 0$$

   where $g$ is acceleration due to gravity, $L$ is the length of the pendulum, and $\theta$ is the angular displacement. Use the *forward Euler method* to estimate the pendulum's position. You are also expected to animate motion of the pendulum using Python's `matplotlib` library.

4. Since its introduction in the 1920's, the *Van der Pol equation*[2] has been a prototype for systems with self-excited limit cycle oscillations. This equation is now considered as a basic model for oscillatory processes in physics, electronics, biology, neurology, sociology and economic. The equation is described by the following second order ODE:

$$\frac{d^2x(t)}{dt^2} - \mu(1 - x(t)^2)\frac{dx(t)}{dt} + x(t) = 0$$

   It is known that solution of the above equation exhibits a limit cycle when $\mu > 0$. Write a function that takes the parameter $\mu$ (a positive real number) and initial condition as arguments, and computes period of the limit cycle. Your code is also expected to plot the solution.

   **HINT: Use the function `scipy.integrate.solve_ivp`.**

5. In physics and classical mechanics, the three-body problem involves taking the initial positions and velocities (or momenta) of three point masses and solving for their subsequent motion according to *Newton's laws of motion* and *Newton's law of universal gravitation*. Unlike two-body problems, no general closed-form solution exists for this problem, and numerical methods are generally required[3].

[1] `https://en.wikipedia.org/wiki/Pendulum_(mathematics)`
[2] `https://en.wikipedia.org/wiki/Van_der_Pol_oscillator`
[3] `https://en.wikipedia.org/wiki/Three-body_problem`

Question 5 continues on the next page...

Consider 3 bodies of mass $1/G$ (where $G$ is the gravitational constant) that are placed in the x-y plane with initial vector position $\boldsymbol{r}_i(0) = (x_{0,i}, y_{0,i})$ for $i \in \{1, 2, 3\}$ and zero initial velocity. Let $\boldsymbol{r}_i(t)$ denote the position vector of the $i^{\text{th}}$ body at time $t$. Then, the bodies are governed by the following system of ODEs

$$\begin{aligned}
\ddot{\boldsymbol{r}}_1(t) &= \frac{\boldsymbol{r}_2(t) - \boldsymbol{r}_1(t)}{\|\boldsymbol{r}_2(t) - \boldsymbol{r}_1(t)\|^3} + \frac{\boldsymbol{r}_3(t) - \boldsymbol{r}_1(t)}{\|\boldsymbol{r}_3(t) - \boldsymbol{r}_1(t)\|^3} \\
\ddot{\boldsymbol{r}}_2(t) &= \frac{\boldsymbol{r}_1(t) - \boldsymbol{r}_2(t)}{\|\boldsymbol{r}_1(t) - \boldsymbol{r}_2(t)\|^3} + \frac{\boldsymbol{r}_3(t) - \boldsymbol{r}_2(t)}{\|\boldsymbol{r}_3(t) - \boldsymbol{r}_2(t)\|^3} \\
\ddot{\boldsymbol{r}}_3(t) &= \frac{\boldsymbol{r}_1(t) - \boldsymbol{r}_3(t)}{\|\boldsymbol{r}_1(t) - \boldsymbol{r}_3(t)\|^3} + \frac{\boldsymbol{r}_2(t) - \boldsymbol{r}_3(t)}{\|\boldsymbol{r}_2(t) - \boldsymbol{r}_3(t)\|^3}
\end{aligned}$$

Write a function that takes the initial position of the 3 bodies as its argument, and visualizes their trajectories with help of Python's `scipy.integrate` and `matplotlib` libraries.