**Control & Instrumentation Lab, Autumn 2020-21**

**Session 2: Solving ODEs and Roots**

At the end of the session, the students shall be able to

1. define an (vector valued) ODE (initial value problem) as a MATLAB .m file and solve using in-built function ode45 <https://in.mathworks.com/help/matlab/ref/ode45.html>
2. implement Runge-Kutta (RK4) method to solve the above problem
3. plot phase portrait of a dynamical system specified by an ODE and superimpose the trajectory on the phase portrait

**Theory:**

Consider a first-order ordinary differential equation (ODE) stated as:

where the goal is to solve for a trajectory that satisfies the above ODE over an interval with the initial condition specified.

This is a fundamental problem that arises in many areas of science and engineering, and particularly in dynamical systems and control theory.

For some problems, an explicit solution of the ODE can be found. Otherwise, we need to approximate the solution using numerical techniques. These techniques are largely based on approximating the integral in the following equation which is a consequence of the fundamental theorem of calculus:

In general, , i.e., it is a vector-valued signal, the function and the derivative/integral operator(s) act component-wise.

**Euler’s method:**

This is a first order method that approximates the above integral by the function value for a suitable choice of sufficiently small step-size .

The algorithm proceeds iteratively as:

The parameter controls the trade-off between accuracy and computations. The discretized trajectory approximate the continuous trajectory.

**Runge–Kutta method (4th order) (RK4):**

This is a fourth order method that approximates the above integral by evaluating the function at four different intermediate points. The algorithm proceeds as described below.

The above algorithms are two of the most fundamental class of numerical methods to solve ODEs. There are advanced algorithms that have been developed, including those that choose the step size adaptively.

Matlab has several in-built ODE solvers. See here <https://www.mathworks.com/help/matlab/math/choose-an-ode-solver.html> for a discussion on how to choose the appropriate solver for the problem at hand.

The solver ode45 is based on the RK4 method discussed above. Type ‘help ode45’ to see how to pass different parameters and function handles to it and get the solution.

**Practice Problems and Solutions (Please add your answer (Matlab command + output) below each question in a different font color)**

1. Write a function that takes as input the handle of an user-defined function, time interval, initial value and step size and returns as output the trajectory and computed by the Euler’s method.
2. Solve the following ODEs using Euler’s method you have written in Q1.
   1. over the interval for values of . Compare the trajectory with the actual solution of the ODE evaluated at time instants on the same figure with values in the legend.
   2. over the interval for values of . Compare the trajectory with the actual solution of the ODE on the same figure with values in the legend.

What do you notice about accuracy and value of h? To determine accuracy, compute the norm of the difference of the trajectory vector generated by the algorithm and the vector of actual solution evaluated at the same time instants as the algorithm.

Use ‘tic’ ‘toc’ to find the time it took for your program to generate the trajectories. What do you notice about computation time and value of h?

1. Write a function that takes as input the handle of an user-defined function, time interval, initial value and step size and returns as output the trajectory and computed by the RK4 method.
2. Solve problem 2b using the RK4 method you have implemented in problem Q3. Compare the accuracy and computation time with the trajectory given by the Euler’s method. Specifically, fill out the following table for problem 2b.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Step size h** | **Computation Time (s)** | | **Accuracy//ERROR** | |
| **Euler** | **RK4** | **Euler** | **RK4** |
| 0.1 | 0.0094 | 0.0043 | 12.66 | 3.907e-14 |
| 0.01 | 0.0015 | 0.0053 | 3.77 | 2.05e-13 |
| 0.001 | 0.0184 | 0.0497 | 1.18 | 2.61e-12 |

1. **Pendulum:** The equation of motion of a simple undamped pendulum is given by

where , is the length of the string connecting the mass and is the angular displacement. Let m.

First introduce variables and and write the above differential equation as a coupled linear ODE. Then, solve the above ODE with initial condition over the interval using Matlab’s inbuilt function ode45, and your implementation of Euler and RK4 methods with step sizes and . Plot the trajectories side by side using ‘subplot’ command of Matlab.

Repeat the above for a damped pendulum with equations of motion given by

where is the mass of the object and is the damping coefficient.

1. Consider a series RL circuit connected with a dc voltage source with V = 1 volt. Let R = 0.5 ohm and L = 1 H. Numerically solve for the current through the circuit with I(0) = 0 amp and I(0) = 3amp by the RK4 method over the time interval [0,20s]. Does the result match with your theoretical predictions?
2. In this question, we will learn how to geometrically represent the trajectories of a two dimensional time invariant dynamical system (i.e., one where the function is independent of and ). General procedure is defined below.

Step 1: Create a 2-D grid using the function ‘meshgrid’ by specifying the range and

granularity of y1 and y2.

Step 2: Define two vectors y1dot and y2dot that encode the values of the gradients by

evaluating the function f at all points of the grid created above.

Step 3: Use the function ‘quiver’ to plot y1dot and y2dot on the grid y1 and y2. Label

axes properly.

The small arrows indicate the direction (derivative) of the trajectory.

Using the above procedure, create the phase portrait both the undamped and damped pendulum with parameters given in Q5 over interval [-1,1] for both y1 and y2 with granularity 0.05.

Now, solve for the trajectory using ode45 over the interval [0,20s] with initial condition [0;0.5] and superimpose the trajectory on the phase portrait (use ‘hold on’ command) with a different color. Mark the start and end points with a circle and a square.

Repeat the above for b = 0 and b = 0.3 and notice how oscillation is represented on the phase plot.

1. Consider a linear time-invariant dynamical system with Plot the phase portrait and the trajectory on the phase portrait with a different color for the following values of A and initial conditions.
   1. over interval [0,10]
   2. over interval [0,20]
   3. over interval [0,5] and [0,10]
   4. over interval [0,10], over interval [0,4], over interval [0,4].

Also plot the state trajectories vs. time by solving the ODE using ode45 or your own implementation of RK4.

Now, find the eigenvalues of each of the A matrices and discuss how they are related to the behavior of the system trajectory.