MSAS – Assignment #1: Simulation

Tarun Singh, 10894156

1 Implicit equations

Exercise 1

Consider the physical model of a mechanical system made of four rigid rods $\mathbf{a}_i$ with $i \in [1;4]$ in a kinematic chain, as shown in Fig. 1. The system has only two degrees of freedom. The objective is to determine, fixing the angle β, the corresponding angle of equilibrium α between the rods $\mathbf{a}_1$ and $\mathbf{a}_2$. Writing the kinematic closure, it is possible to obtain the following equation

$$\frac{a_1}{a_2}\cos\beta - \frac{a_1}{a_4}\cos\alpha - \cos(\beta-\alpha) = -\frac{a_1^2 + a_2^2 - a_3^2 + a_4^2}{2a_2a_4} \tag{1}$$

where a_i is the length of the corresponding rod. This equation, called the equation of Freudenstein, can be clearly re-written in the form $f(\alpha) = 0$ and solved, once β is fixed, for a particular value of α. Assume that $a_1 = 10$ cm, $a_2 = 13$ cm, $a_3 = 8$ cm, and $a_4 = 10$ cm and

1) Implement a general-purpose Newton solver (NS).

2) Solve Eq. (1) with NS for $\beta \in [0, \frac{2}{3}\pi]$ with the analytical derivative of $f(\alpha)$ and tolerance set to 10^{-5}. For every β value, use two initial guesses, -0.1 and $\frac{2}{3}\pi$. What do you observe? You can validate your results with the Matlab© built-in function `fzero`.

3) Repeat point 2) using the derivative estimated through finite differences and compare the accuracy of the two methods.

4) Eq. (1) can be solved analytically only for specific values of β. Find at least one and compare the analytical solution with yours.

5) Repeat point 2) extending the β interval up to π. What do you observe?

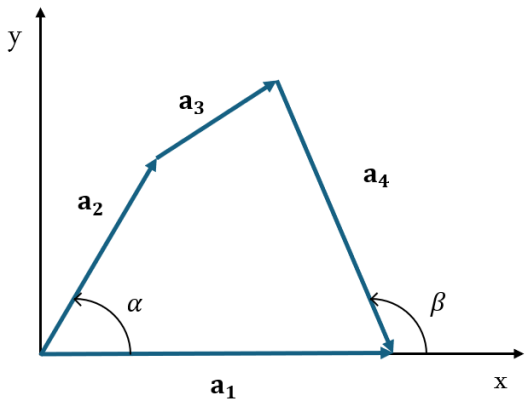


Figure 1: Kinematic chain.

(4 points)

Answer 1

1) Implemented in zipped code as a Matlab© function named the Newton_Method.

   - Inputs: Analytical function, Analytical derivative of function, initial conditions of α, tolerance, maximum allowed iterations, β value, condition for the finite difference method, step size for FD (h = 1e-6).
   - Outputs: Converged value of α, number of iterations taken, flagging success of the iteration.

2) Eq. (1) was solved for 69 equally spaced values (to have $\pi/2$ in discrete points) of $\beta \in [0, \frac{2}{3}\pi]$ and two initial guesses of α, -0.1 and $\frac{2}{3}\pi$. Two sets of solutions (shown in Fig. 2(a)) were obtained for two different initial values of α referring to two different orientations of the kinematic chain. In one orientation, the link a_2 of the kinematic chain is below the x-axis ($\alpha < 0$), initially, and in another orientation, the link a_2 is above the x-axis ($\alpha > 0$).

3) The comparison of results provided by Newton solver using analytical derivative and Newton solver with the finite difference method (central difference scheme) is presented in Fig. 2(b)). Moreover, Fig. 3 compares the accuracy of both methods by demonstrating the absolute error w.r.t. analytical (`fzero`) solution.

4) After putting the values of lengths of the links in Eq. (1) reduces to the following equation

$$52\cos\alpha(1+\cos\beta) + 52\sin\alpha\sin\beta - 40\cos\beta - 61 = 0 \tag{2}$$

   Eq. (2) can be solved analytically for $\beta = 0$ and $\frac{\pi}{2}$.

   - For $\beta = 0$: $\cos\alpha = \frac{101}{104} \implies \alpha = \pm 0.2407733958$ rad
   - For $\beta = \frac{\pi}{2}$: $\sin\alpha + \cos\alpha = \frac{61}{52} \implies \alpha = 1.377999392$ & 0.1927969346 rad (with $\sin\alpha = \frac{2\tan\alpha/2}{1+\tan^2\alpha/2}$ & $\cos\alpha = \frac{1-\tan^2\alpha/2}{1+\tan^2\alpha/2}$)

   Table 1 shows the values of α obtained through NS, `fzero`, and NS with finite difference for $\beta = 0$ and $\frac{\pi}{2}$ with initial guesses of α, -0.1 and $\frac{2}{3}\pi$, along with analytical results:

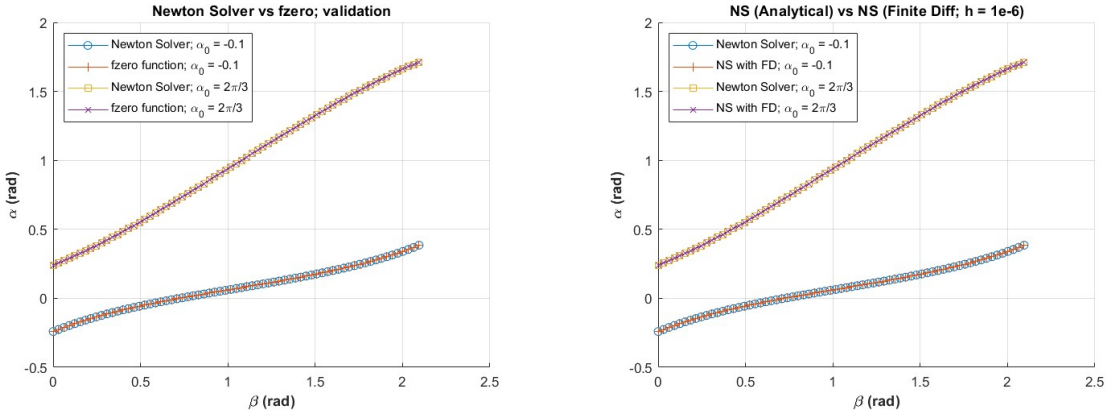
	Initial guess of α	**NS**	**`fzero`**	**NS(FD)**	**Analytical**
$\beta = 0$	$\alpha_0 = -0.1$	-0.24077	-0.24077	-0.24077	-0.2407733958
	$\alpha_0 = 2\pi/3$	0.24078	0.24077	0.24078	0.2407733958
$\beta = \pi/2$	$\alpha_0 = -0.1$	0.1928	0.1928	0.1928	0.1927969346
	$\alpha_0 = 2\pi/3$	1.378	1.378	1.378	1.377999392

Table 1: Comparison of solutions for $\beta = 0$ & $\frac{\pi}{2}$.

   The results in Table 1 demonstrate that the two roots derived analytically from Eq. (2) for each value of β correspond to two distinct solutions computed using the Newton solver (analytical and finite difference), each with a different initial condition.

5) With a gradual increase in the angle β, the solution from Newton Solvers converges to the same value regardless of the two initial conditions. The value of β corresponding to this scenario can be considered the upper limit for the range of the angle β, which was found to be 2.6365 rad through extension (shown in Fig. 4).

   This indicates that only one orientation (positive value of α) of the kinematic chain is achievable for the upper limit of the angle β.



(a) (b)

Figure 2: (a) Validation of Newton Solver with `fzero`. (b) The accuracy of the comparison of Newton solver with analytical derivative and derivative is estimated through finite differences.

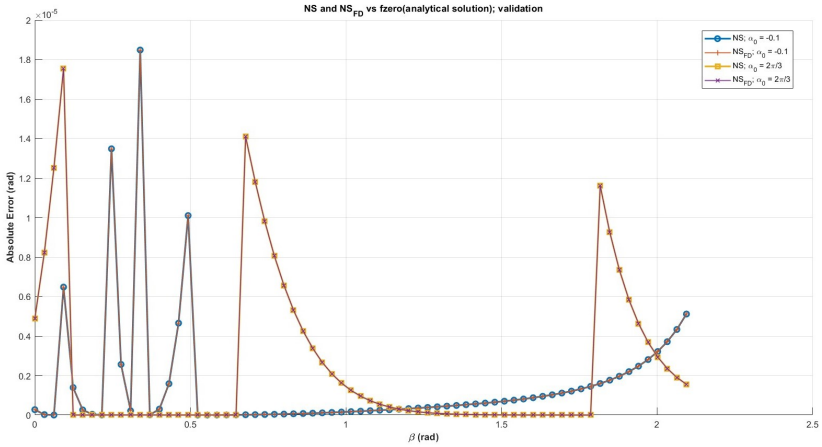


Figure 3: Accuracy comparison of both methods w.r.t. analytical solution

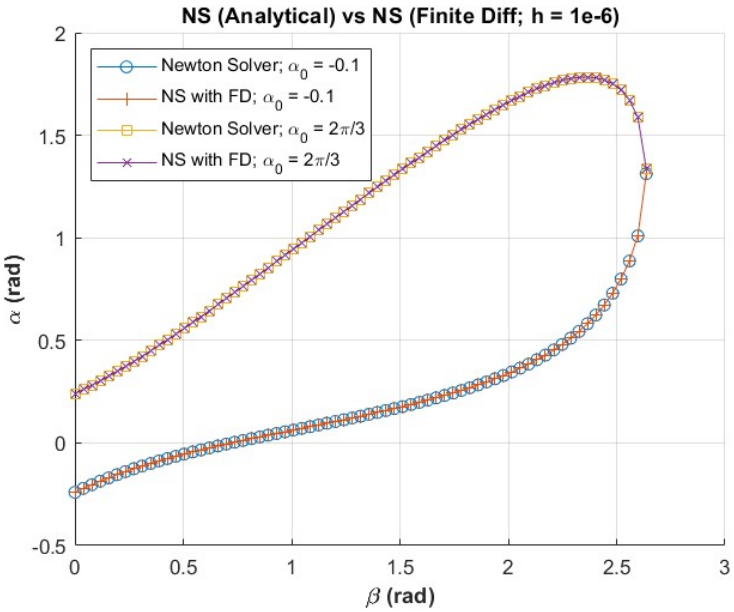


Figure 4: Upper limit of β

2 Numerical solution of ODE

Exercise 2

The physical model of a moving object with decreasing mass is shown in Fig. 5. The mathematical model of the system is expressed by the equations:

$$m(t)\frac{dv}{dt} = \overline{F} + f(t) - \alpha \cdot v \tag{3}$$

$$m(t) = m_0 - c_m \cdot t \tag{4}$$

$$\overline{F} = -\ 0.5 \cdot \rho \cdot C_d \cdot A_m \cdot v^2 \tag{5}$$

Assuming that: $v(0) = 0$ [m/s], $m_0 = 20$ [kg], $c_m = 0.1$ [kg/s], $f(t) = 1$ [N], $\alpha = 0.01$ [Ns/m], $\rho = 0$ [kg/m^3], and Eq. (3) has the exact solution:

$$v(t) = \frac{f(t)}{\alpha} - \left[\frac{f(t)}{\alpha} - v(0)\right]\left[1 - \frac{c_m \cdot t}{m_0}\right]^{\frac{\alpha}{c_m}} \tag{6}$$

Answer to the following tasks:

1) Implement a general-purpose, fixed-step Heun's method (RK2);
2) Solve Eq. (3) using RK2 in $t \in [0, 160]$s for $h_1 = 50$, $h_2 = 20$, $h_3 = 10$, $h_4 = 1$ and compare the numerical vs the analytical solution;
3) Repeat points 1)–2) with RK4;
4) Trade-off between CPU time & integration error.

Now, assuming that: the fluid density is $\rho = 900$ [kg/m^3], $C_d = 2.05$, and $A_m = 1$ [m^2], answer to the following tasks:

1) Solve Eq. (3) using RK2 in $t \in [0, 160]$s for $h_1 = 1$;
2) From the results of the previous point, use a proper ode of Matlab to solve Eq. (3) ;
3) Discuss the results.

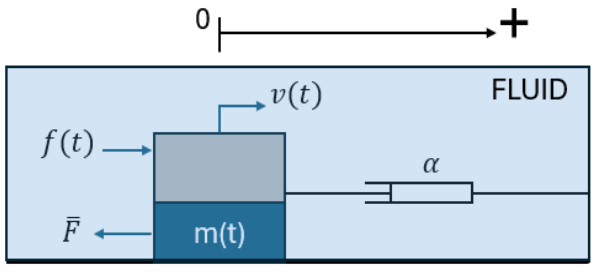


Figure 5: Moving object with decreasing mass.

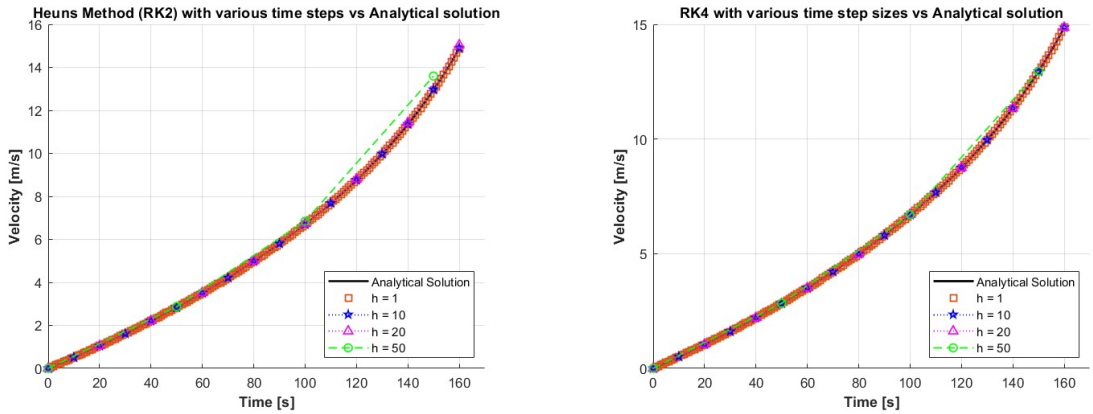
(5 points)

Answer 2

1) Implemented in zipped code as a Matlab© function named the heuns_method.
   - Inputs: ODE, time span of the problem, velocity initial conditions, time step size.
   - Outputs: Velocity (solution) vector, time vector.
2) Implemented Heuns method(RK2) function was used to solve Eq. (3) in provided time span for provided time step-sizes. The analytical solution was obtained by plotting Eq. (6) with a time step of 0.1. Fig. 6(a) and Fig. 7(a) compare four numerical solutions vs analytical solutions..
3) Points 1)-2) were repeated with RK4. Fig. 6(b) and Fig. 7(b) compares four numerical solutions vs analytical solution. Fig. 7 shows that the error reduces with the decreasing step size for both RK2 and RK4.
4) In Fig. 8, the trade-offs between CPU time and integration errors (absolute and max) are demonstrated for both Runge-Kutta methods.

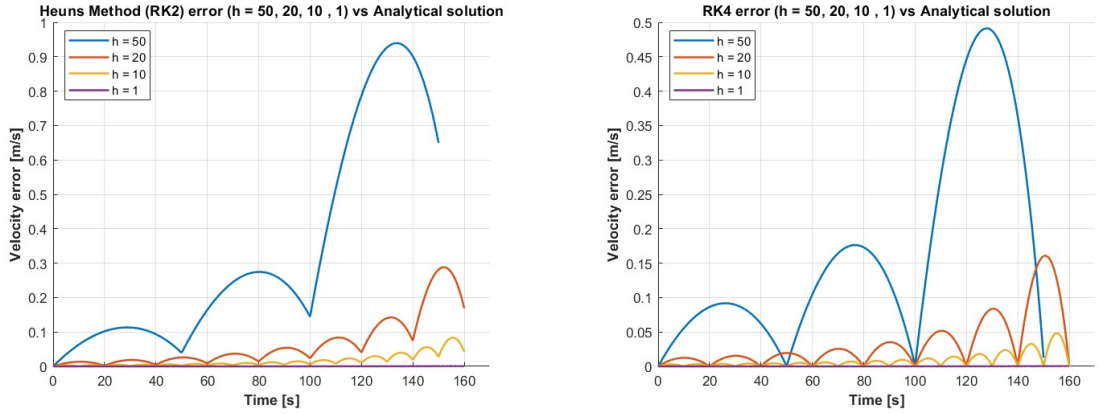
For $\rho = 900$ [kg/m^3], $C_d = 2.05$, and $A_m = 1$ [m^2]:

1) Solving Eq. (3) using RK2 for $h = 1$ does not provide a stable solution as $h > h_{max}$ (Figure of unstable solution can be obtained from running the Matlab script). A stable solution can be obtained for $h < h_{max}$, but this makes the computation very expensive. The modified ODE of the problem requires a solver with adaptive step size.
2) `ode45` was selected automatically by Matlab based on the information (non-stiff) related to the modified ODE. It is implemented in the code using the built-in Matlab function `ode` (`F = ode`). `F.SelectedSolver` command is used to show the selected solver in the command window. Fig. 9 shows that extremely small step sizes were adapted in the initial stage of the period and later increased.



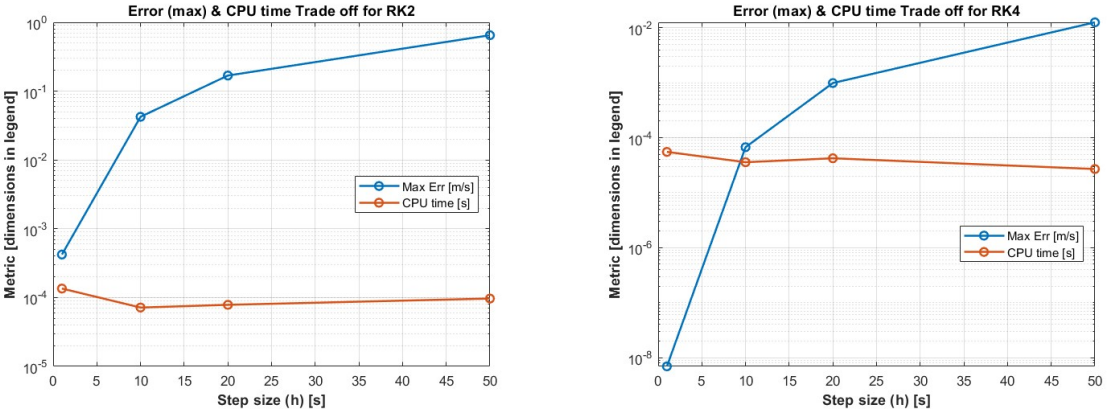
(a) RK2 (b) RK4

Figure 6: RK2 and RK4 solutions (h = 50, 20, 10, 1) vs Analytical solution.



(a) RK2 (b) RK4

Figure 7: RK2 and RK4 errors (h = 50, 20, 10, 1) with respect to Analytical solution.



(a) RK2 (b) RK4

Figure 8: CPU time and integration error trade off.

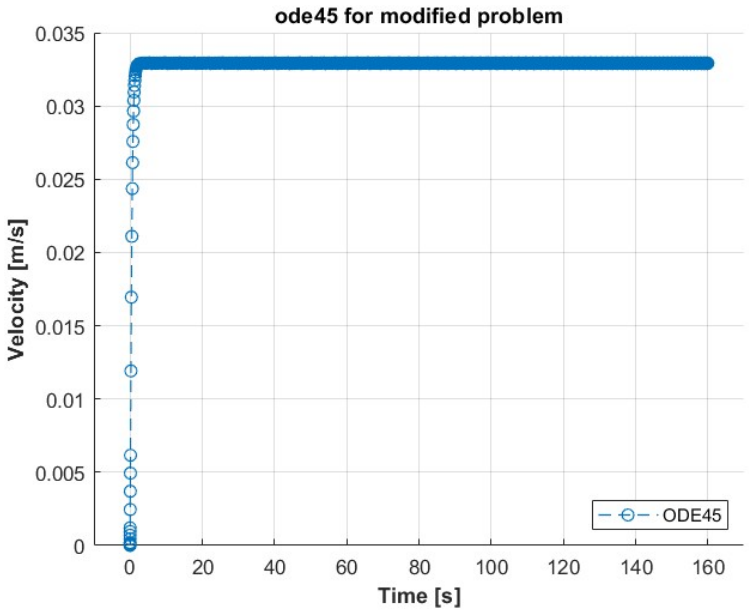


Figure 9: Solution of modified problem using `ode45`; adaptive step size can be seen.

Exercise 3

Let $\dot{\mathbf{x}} = A(\alpha)\mathbf{x}$ be a two-dimensional system with $A(\alpha) = [0, 1; -1, 2\cos\alpha]$. Notice that $A(\alpha)$ has a pair of complex conjugate eigenvalues on the unit circle; α denotes the angle from the $\mathrm{Re}\{\lambda\}$-axis.

a) Runge-Kutta methods

1) Write the operator $F_{\mathrm{RK2}}(h, \alpha)$ that maps $\mathbf{x}_k$ into $\mathbf{x}_{k+1}$, namely $\mathbf{x}_{k+1} = F_{\mathrm{RK2}}(h, \alpha)\,\mathbf{x}_k$.
2) With $\alpha = \pi$, solve the problem "Find $h \geq 0$ s.t.$\max(|\mathrm{eig}(F(h, \alpha))|) = 1$".
3) Repeat point 2) for $\alpha \in [0, \pi]$ and draw the solutions in the $(h\lambda)$-plane.
4) Repeat points 1)–3) with RK4.

b) Backinterpolation methods

Consider the backinterpolation method $\mathrm{BI2}_{0.3}$.

1) Derive the expression of the linear operator $B_{\mathrm{BI2}_{0.3}}(h, \alpha)$ such that $\mathbf{x}_{k+1} = B_{\mathrm{BI2}_{0.3}}(h, \alpha)\mathbf{x}_k$.
2) Using the same approach of a), draw the stability domain of $\mathrm{BI2}_{0.3}$ in the $(h\lambda)$-plane.
3) Derive the domain of numerical stability of $\mathrm{BI2}_\theta$ for the values of $\theta = [0.2,\ 0.4,\ 0.6,\ 0.8]$.

(8 points)

Answer 3

a) Runge-Kutta Methods

1) Operator $F_{\mathrm{RK2}}(h, \alpha)$ can be derived from $RK2$ algorithm with following steps:

$$
\begin{aligned}
Predictor: \quad & \dot{\mathbf{x}}_k = A \cdot \mathbf{x}_k \\
& \mathbf{x}^P_{k+1} = \mathbf{x}_k + h \cdot \dot{\mathbf{x}}_k \\
Corrector: \quad & \dot{\mathbf{x}}^P_{k+1} = A \cdot \mathbf{x}^P_{k+1} \\
& \mathbf{x}^C_{k+1} = \mathbf{x}_k + \frac{h}{2} \cdot (\dot{\mathbf{x}}_k + \dot{\mathbf{x}}^P_{k+1})
\end{aligned}
$$

Substituting $\dot{\mathbf{x}}_k$ and $\dot{\mathbf{x}}^P_{k+1}$ for $\mathbf{x}^C_{k+1}$:

$$
\begin{aligned}
\implies \quad & \mathbf{x}^C_{k+1} = \mathbf{x}_k + \frac{h}{2} \cdot (A \cdot \mathbf{x}_k + A \cdot (\mathbf{x}_k + h \cdot A \cdot \mathbf{x}_k)) \\
\implies \quad & \mathbf{x}_{k+1} = (I + h \cdot A + \frac{(h \cdot A)^2}{2}) \cdot \mathbf{x}_K \\
\implies \quad & F_{\mathrm{RK2}}(h, \alpha) = I + h \cdot A + \frac{(h \cdot A)^2}{2}
\end{aligned}
$$

2) The values of h for methods $RK2$ and $RK4$ with $\alpha = \pi$ is 2.00 and 2.78529, respectively. These results are shown in the Matlab command window after running the related section of the script.

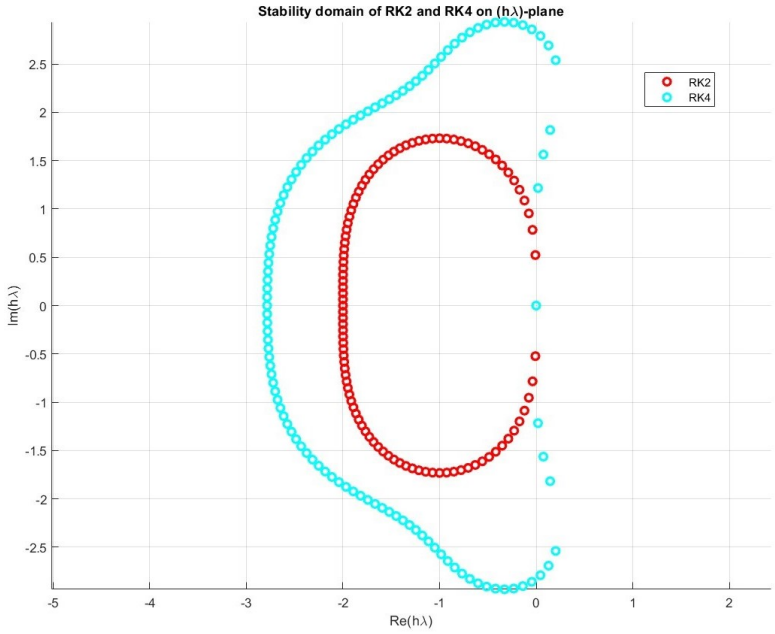


Figure 10: Stability domain of $RK2$ and $RK4$ methods drawn on $(h\lambda)$-plane.

3) The solution of the problem provides the stability domain for $RK2$ method. In Fig. 10, the domain of numerical stability of $RK2$ method is drawn with small red circles.

4) Operator $F_{\text{RK4}}(h, \alpha)$ can be derived from $RK4$ algorithm with following steps:

$$
\begin{aligned}
0^{th} stage: \quad & \dot{\mathbf{x}}_k = A \cdot \mathbf{x}_k \\
1^{st} stage: \quad & \mathbf{x}^{P_1} = \mathbf{x}_k + \frac{h}{2} \cdot \dot{\mathbf{x}}_k \\
& \dot{\mathbf{x}}^{P_1} = A \cdot \mathbf{x}^{P_1} \\
2^{nd} stage: \quad & \mathbf{x}^{P_2} = \mathbf{x}_k + \frac{h}{2} \cdot \dot{\mathbf{x}}^{P_1} \\
& \dot{\mathbf{x}}^{P_2} = A \cdot \mathbf{x}^{P_2} \\
3^{rd} stage: \quad & \mathbf{x}^{P_3} = \mathbf{x}_k + \frac{h}{2} \cdot \dot{\mathbf{x}}^{P_2} \\
& \dot{\mathbf{x}}^{P_3} = A \cdot \mathbf{x}^{P_3} \\
4^{th} stage: \quad & \mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{6} \cdot (\dot{\mathbf{x}}_k + 2\dot{\mathbf{x}}^{P_1} + 2\dot{\mathbf{x}}^{P_2} + \dot{\mathbf{x}}^{P_3})
\end{aligned}
$$

Substituting $\dot{\mathbf{x}}_k$, $\dot{\mathbf{x}}^{P_1}$, $\dot{\mathbf{x}}^{P_2}$, and $\dot{\mathbf{x}}^{P_3}$ for $\mathbf{x}_{k+1}$:

$$
\begin{aligned}
\implies \quad & \mathbf{x}_{k+1} = (I + h \cdot A + \frac{(h \cdot A)^2}{2} + \frac{(h \cdot A)^3}{6} + \frac{(h \cdot A)^4}{24}) \cdot \mathbf{x}_k \\
\implies \quad & F_{\text{RK4}}(h, \alpha) = I + h \cdot A + \frac{(h \cdot A)^2}{2} + \frac{(h \cdot A)^3}{6} + \frac{(h \cdot A)^4}{24}
\end{aligned}
$$

5) Similarly to $RK2$ problem, the $RK4$ problem was solved to determine the corresponding domain of numerical stability shown in Fig. 10, drawn with small cyan circles.

b) Backinterpolation method

Backinterpolation method $\text{BI2}_{0.3}$ is a special case of second order θ-method with $\theta = 0.3$. θ-methods are a combination of RK (θ, forward) algorithms and BRK ($(1-\theta)$, backward) algorithms. In θ-method, we find $\mathbf{x}_{k+1}$ by equating two partial steps, $\mathbf{x}_{k+\theta}$ (estimated using RK) and $\mathbf{x}_{k+(1-\theta)}$ (estimated using BRK).

1) Linear operator $B_{\text{BI2}_{0.3}}(h, \alpha)$ can be derived with following algorithm:
*RK*2:

$$
\begin{aligned}
Predictor: \quad & \dot{\mathbf{x}}_k = A \cdot \mathbf{x}_k \\
& \mathbf{x}^P_{k+\frac{\theta}{2}} = \mathbf{x}_k + \frac{\theta h}{2} \cdot \dot{\mathbf{x}}_k \\
Corrector: \quad & \dot{\mathbf{x}}^P_{k+\frac{\theta}{2}} = A \cdot \mathbf{x}^P_{k+\frac{\theta}{2}} \\
& \mathbf{x}^C_{k+\theta} = \mathbf{x}_k + \theta h \cdot \dot{\mathbf{x}}^P_{k+\frac{\theta}{2}} \\
\implies \quad & \mathbf{x}_{k+\theta} = (I + \theta h \cdot A + \frac{(\theta h \cdot A)^2}{2}) \cdot \mathbf{x}_k
\end{aligned}
$$

*BRK*2:

$$
\begin{aligned}
Predictor: \quad & \dot{\mathbf{x}}_{k+1} = A \cdot \mathbf{x}_{k+1} \\
& \mathbf{x}^P_{k+\frac{(1-\theta)}{2}} = \mathbf{x}_{k+1} - \frac{(1-\theta)h}{2} \cdot \dot{\mathbf{x}}_{k+1} \\
Corrector: \quad & \dot{\mathbf{x}}^P_{k+\frac{(1-\theta)}{2}} = A \cdot \mathbf{x}^P_{k+\frac{(1-\theta)}{2}} \\
& \mathbf{x}^C_{k+(1-\theta)} = \mathbf{x}_{k+1} - (1-\theta)h \cdot \dot{\mathbf{x}}^P_{k+\frac{(1-\theta)}{2}} \\
\implies \quad & \mathbf{x}_{k+(1-\theta)} = (I - (1-\theta)h \cdot A + \frac{(1-\theta)h \cdot A)^2}{2}) \cdot \mathbf{x}_{k+1}
\end{aligned}
$$

Equating $\mathbf{x}_{k+\theta}$ and $\mathbf{x}_{k+(1-\theta)}$ and rearranging for $\mathbf{x}_{k+1}$:

$$
\begin{aligned}
\implies \quad & \mathbf{x}_{k+1} = [I - (1-\theta)h \cdot A + \frac{((1-\theta)h \cdot A)^2}{2}]^{-1} \cdot [I + \theta h \cdot A + \frac{(\theta h \cdot A)^2}{2}] \cdot \mathbf{x}_k \\
\implies \quad & B_{\text{BI2}_{0.3}}(h, \alpha) = [I - (1-\theta)h \cdot A + \frac{((1-\theta)h \cdot A)^2}{2}]^{-1} \cdot [I + \theta h \cdot A + \frac{(\theta h \cdot A)^2}{2}]
\end{aligned}
$$

2) Fig. 11 shows the domain of numerical stability for $\text{BI2}_{0.3}$ drawn in $(h\lambda)$-plane.

3) Similarly, Fig. 12 illustrates the domain of numerical stability for BI2_θ in the $(h\lambda)$-plane for $\theta = [0.2, 0.4, 0.6, 0.8]$. In Fig. 12, the stability domains for $\theta = [0.2, 0.4]$ are located in the right semi-plane, where the *BRK* algorithm demonstrates greater dominance compared to the *RK* algorithm. Conversely, for $\theta = [0.6, 0.8]$, the stability domains are in the left semi-plane, where the *RK* algorithm is more dominant than the *BRK* algorithm.

   For $\theta = [0.2, 0.4]$, the stable domain is enclosed within the curvature, while the unstable domain lies outside the curvature. In contrast, for $\theta = [0.6, 0.8]$, the stable domain is outside the curvature, with the unstable domain confined within the curvature.

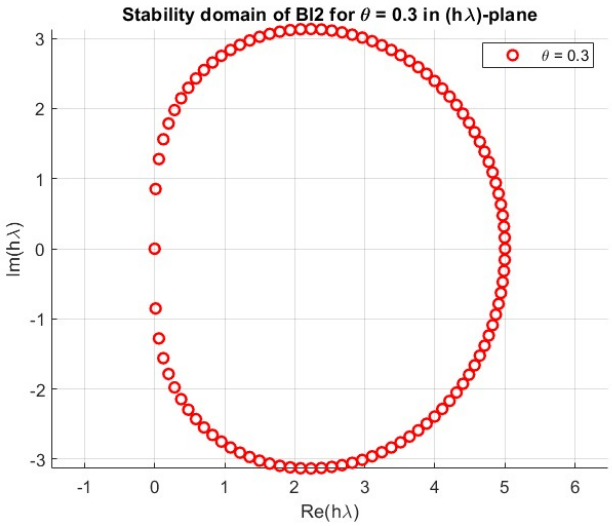


Figure 11: Stability domain of $\text{BI2}_{0.3}$ method drawn on $(h\lambda)$-plane.

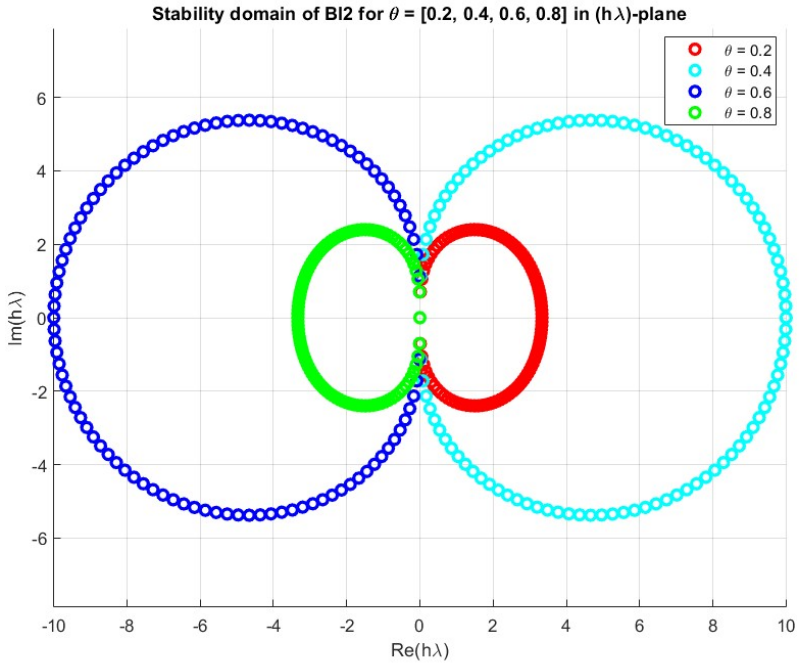


Figure 12: Stability domain of BI2_θ method for the values of $\theta = [0.2, 0.4, 0.6, 0.8]$ drawn on $(h\lambda)$-plane.

Exercise 4

The cooling problem of a high-temperature mass is shown in Fig. 13. Assuming that: K_c= 0.0042 [J/(s K)] and K_r = 6.15 x 10^{-11} [J/(s K^4)] are the convective and radiation heat loss coefficients respectively, $\widetilde{T}$ is the mass temperature in time, T_a = 277 [K] is the surrounding air temperature, C= 45 [J/K] is the mass thermal capacity, and $\widetilde{T}(0)$ = 555 [K] is the initial mass temperature.

1) Write the mathematical model of the system.
2) Solve the mathematical model using: RK2 with h= 720, and RK4 with h= 1440.
3) Compare the solution of point 2) with the "exact" one. Discuss the results. (Hint: use an ODE solver of Matlab).

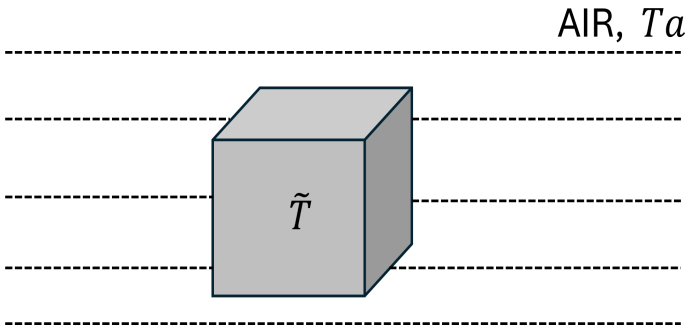


Figure 13: The cooling of high-temperature mass.

(4 points)

Answer 4

1) Mathematical model:
   - Convective heat transfer: $Q_c = K_c(T - T_a)$
   - Radiative heat transfer: $Q_r = K_r(T^4 - T_a^4)$
   - Energy balance equation: $C\frac{dT}{dt} = -Q_c - Q_r = -[K_c(T - T_a) + K_r(T^4 - T_a^4)]$
   - Rearranging for ODE representation: $f(t, T) = \frac{dT}{dt} = -\frac{1}{C}[K_c(T - T_a) + K_r(T^4 - T_a^4)]$
   - Note: $\widetilde{T}$ is written as T in the mathematical model.
2) General Matlab functions of RK2(Heun's Method) and RK4 written for Ex. 2 was used to solve the mathematical model with $h = 720$ and $h = 1440$, respectively.
3) Fig. 14 shows a comparison of solutions from two solvers and the exact solution. Fig. 15 illustrates the error analysis (*wrt* exact solution) of the two solvers. During the initial stages, the errors for RK4 are approximately double those of RK2, which can be attributed to the larger step size used by RK4 (twice that of RK2). However, the RK4 error decreases to zero more rapidly than the RK2 error, indicating faster convergence of RK4 compared to RK2.

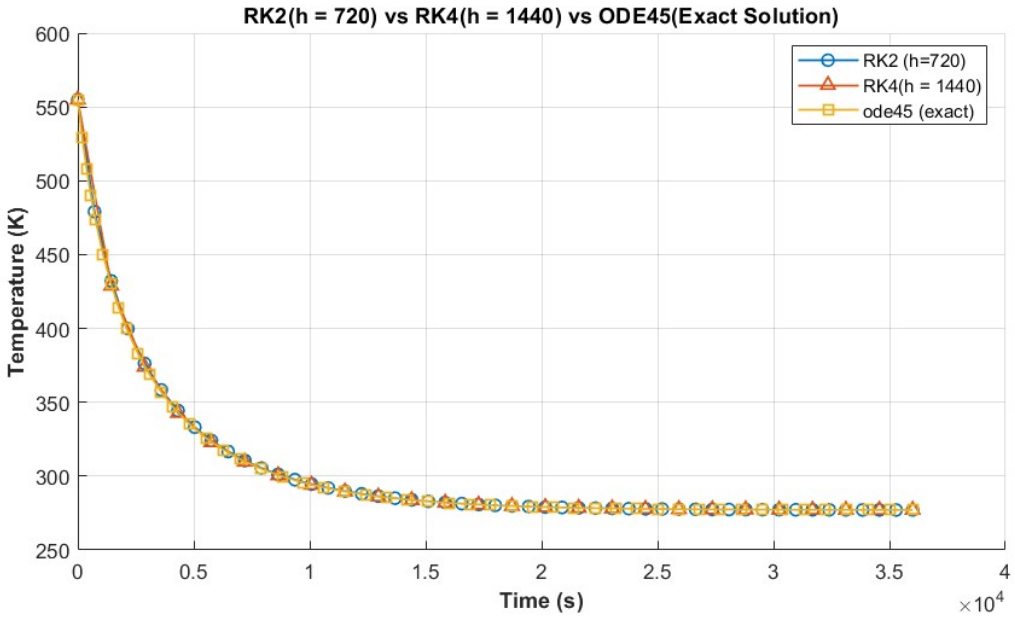


Figure 14: Comparison of solutions from RK2 and RK4 solvers with exact solution

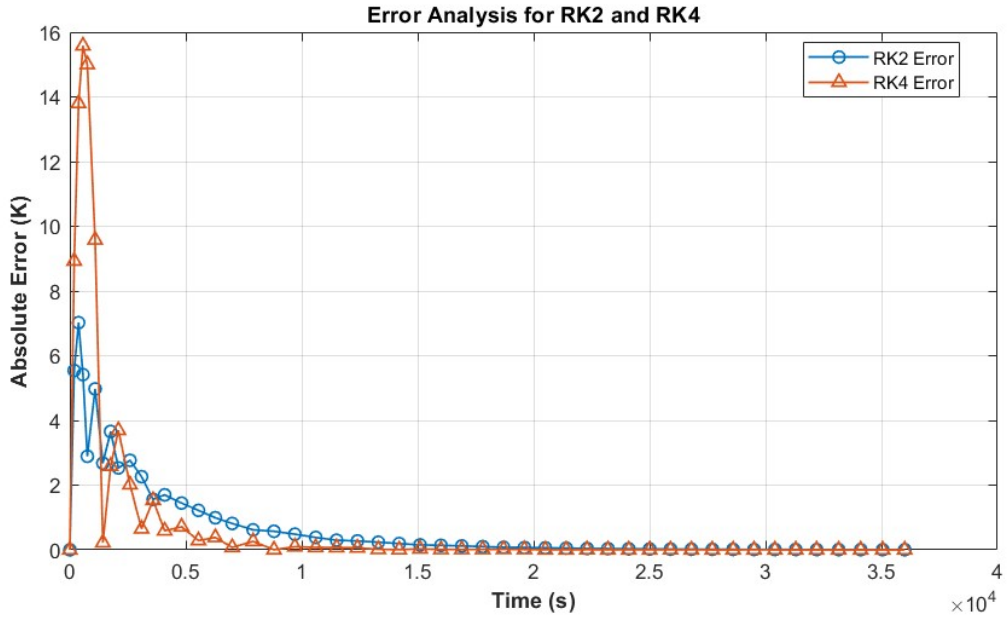


Figure 15: Error analysis of both solvers *wrt* exact solution

Exercise 5

Consider the electrical circuit in Fig. 16. At time t>=0 the switch disconnects the battery and the capacitor discharges its stored energy to the circuit. Answer to the following tasks:

1) Find the state variable form of the circuit with x_1=q and x_2=$\frac{dq}{dt}$, where q is the charge on capacitor.

2) Show that the system is stiff when the circuit parameters are R=25 [Ω], L = 20 [mH], C = 200 [mF], and v_0 = 12 [V]. Represent the eigenvalues on the ($h\lambda$)-plane both for RK2 and IEX4 stability domain.

3) Use IEX4 to simulate the transient response of the system with the parameters of point 2), and determine the largest step size such that RK2 yields a stable and accurate solution.

4) Discuss the results.

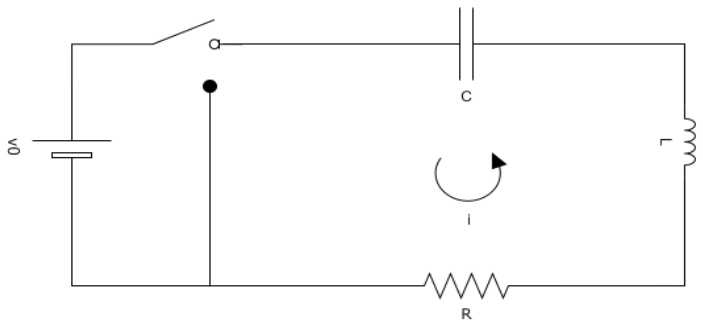


Figure 16: Electric circuit model.

(4 points)

Answer 5

1) The following steps were undertaken to express the given electric circuit model in the state-space format, where $\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ represents the state variable.

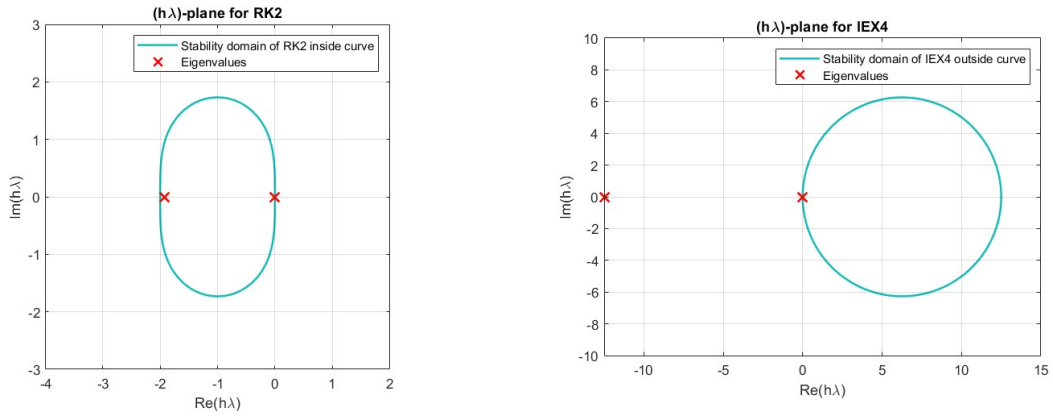
- Kichoff's voltage law across the circuit:
$$V_C + V_L + V_R = 0$$
- Substituting:
$$V_C = \frac{q}{C}, V_L = L\frac{di}{dt} = L\frac{d^2q}{dt^2}, V_R = R\frac{dq}{dt}$$
- Therefore:
$$\frac{q}{C} + L\frac{d^2q}{dt^2} + R\frac{dq}{dt} = 0$$
$$L\frac{d^2q}{dt^2} = -\frac{q}{C} - R\frac{dq}{dt}$$
- Replace $x_1 = q$ and $x_2 = \frac{dq}{dt}$:
$$L\frac{dx_2}{dt} = -\frac{1}{C}x_1 - Rx_2 \text{ and } x_2 = \frac{dx_1}{dt}$$
- This can be represented in matrix form as:
$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{1}{LC} & -\frac{R}{L} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \quad (1)$$
- (1) can be re-written with state variable and matrix abbreviate (A): $\dot{\underline{x}} = A\underline{x}$

2) For the given circuit parameters, the eigenvalues of the system matrix A were calculated using the built-in MATLAB function `eig(A)`. The eigenvalues are -0.2 and -1249.8, differing by an order of magnitude of $O(10^4)$. This significant disparity implies that the system is stiff.
Fig. 17 represents the eigenvalues on the $(h\lambda)-$plane for RK2 and IEX4 stability domains.

- For RK2, $(h) = \frac{2}{1300}$ ensures $(h < h_{max})$ $h\lambda$ lies inside the stability domain (Fig. 17(a)), as discussed in point 3).
- For IEX4, with $(h) = 0.01$, the second eigenvalue is $h\lambda = -12.498$, visible on the left edge of the Fig. 17(b).

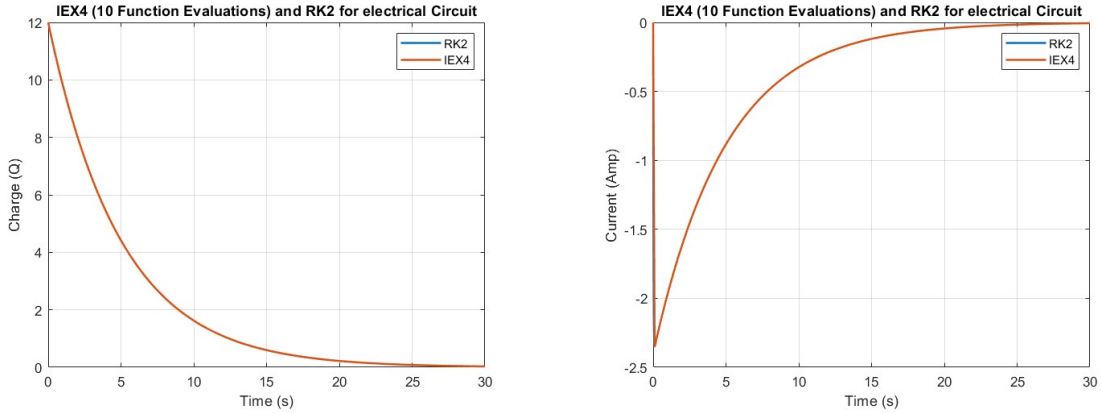
3) Fig. 18 illustrates the simulation results of the transient response of the system. The subfigure Fig. 18(a) shows the first component of the state variable, interpreted as the charge on the capacitor. Meanwhile, Fig. 18(b) displays the second component of the state variable, which corresponds to the current in the electrical circuit.
The eigenvalues of the system only lie on the real axis. For RK2, we find h_{max} such that $|h_{max}\lambda| = 2$ (marginally stable system). Consequently, for the leftmost eigenvalue (-1249.8), $h_{max} = 2/1249.8 \sim 0.00160025$.
For an accurate solution, $h\lambda$ must remain well within the stable region. This largest step size (h_{acc}) can be estimated using a trial-and-error approach by evaluating the absolute error of the RK2 method relative to IEX4, specifically for the second component of the system variable (current). In the Matlab script, applying $h_{acc} = \frac{2}{1400}$ gives a maximum absolute error of order 10^{-3} as shown in Fig. 19.

4) For the given parameters, the condition $R^2 > \frac{4L}{C}$ indicates that the system is overdamped, causing slower decay of the state variable. The capacitor discharges to 99% of its initial charge in approximately $5RC$ seconds. Both the charge and current reach zero after 25 seconds, consistent with the overdamped behavior. This is verified in the transient results obtained using both RK2 and IEX4 methods, showing exponential decay without oscillations.



(a) RK2 (b) IEX4

Figure 17: Eigenvalues on RK2 and IEX4 stability domains.



(a) Charge on the capacitor.

(b) Current in the circuit.

Figure 18: Transient responses of the system for $t \geq 0$.

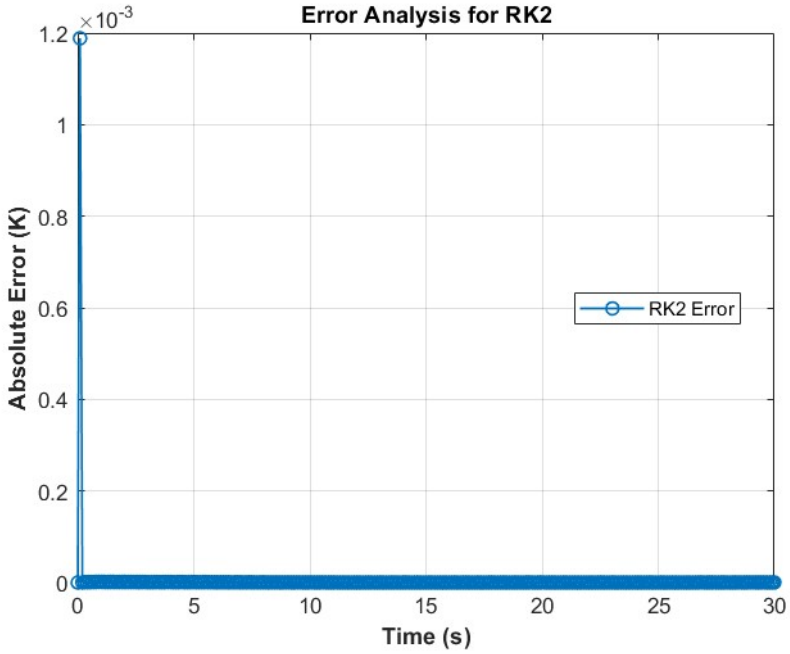


Figure 19: Absolute error for $h_{acc} = \frac{2}{1400}$ *wrt* IEX4

Exercise 6

Consider the bouncing ball physical model in Fig. 20. The mathematical model of the system is represented by the following equations:

$$\frac{dv}{dt} = -g + \frac{F_D}{m_b} + \frac{F_A}{m_b} \tag{7}$$

$$\frac{dx}{dt} = v \tag{8}$$

$$v^+ = -k \cdot v^- \tag{9}$$

$$F_D = -\ 0.5 \cdot \rho \cdot C_d \cdot A_b \cdot v \cdot |v| \tag{10}$$

$$F_A = \rho \cdot V_b \cdot g \tag{11}$$

where the initial conditions are: $v(0) = 0$ [m/s], and $x(0) = 10$ [m].

Assuming that: the ball velocity before (v^-) and after (v^+) the ground impact is governed by the attenuation factor $k = 0.9$, the air density is $\rho = 1.225$ [kg/m^3], the ball mass is $m_b = 1$ [kg], the ball area is $A_b = 0.07$ [m^2], the ball volume is $V_b = 0.014$ [m^3], and the drag coefficient is C_d=1.17, answer to the following tasks:

1) Solve the mathematical model in the time interval $t \in [0, 10)$s using an ode integrator of Matlab.

2) Repeat point 1) for ρ=15 [kg/m^3]. Which is the difference with respect to point 1)?

3) Now solve the problem when the fluid density is ρ=60 [kg/m^3] in $t \in [0, 10)$s. Discuss the result.

4) Repeat point 3) using RK4 for $h = 3$. Discuss the result.

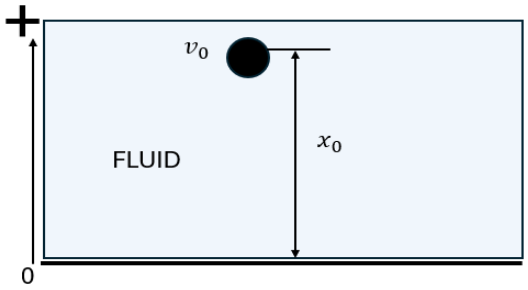


Figure 20: Bouncing ball system.

(5 points)

Answer 6

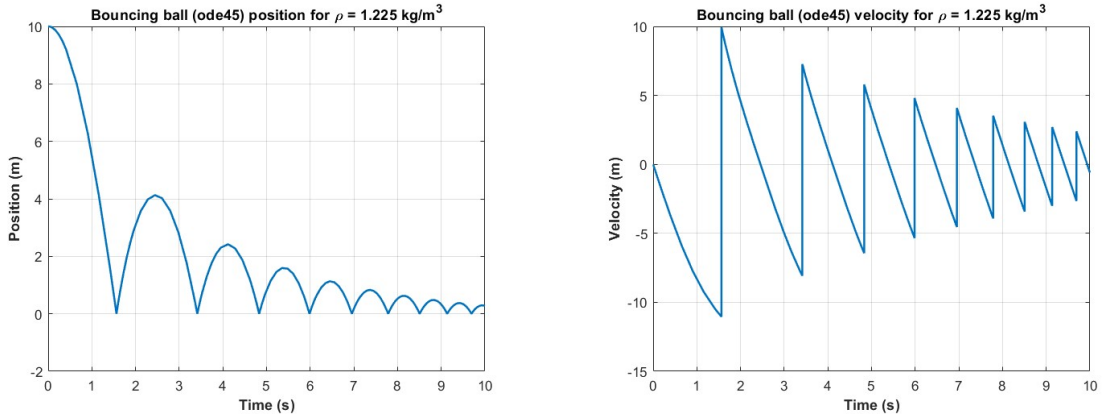
1) The mathematical model was solved using Matlab in-built solver `ode45`. Fig. 21 shows the position [m] and velocity [m/s] of the ball vs time for $\rho = 1.225$ [kg/m^3]. The position plot shows decreasing peaks due to energy loss, and the velocity alternates between positive (upward) and negative (downward), with diminishing amplitude over time.

2) Point 1) was repeated for $\rho = 15$ [kg/m^3]. The `ode45` solver keeps on running and doesn't converge for $t_{final} = 10s$ because the stopping criterion is not well defined. Due to significant drag, the ball continues bouncing with diminishing velocity and height, with

velocity theoretically approaching zero asymptotically but never reaching exactly zero. Since the solution is expected to stop when velocity becomes exactly zero, the simulation was instead terminated at $t_{final} = 7.6$s, and Fig. 22 illustrates corresponding solutions. An appropriate stopping criterion would be setting a tolerance value for the velocity (e.g. $|v| < 10^{-3}$).

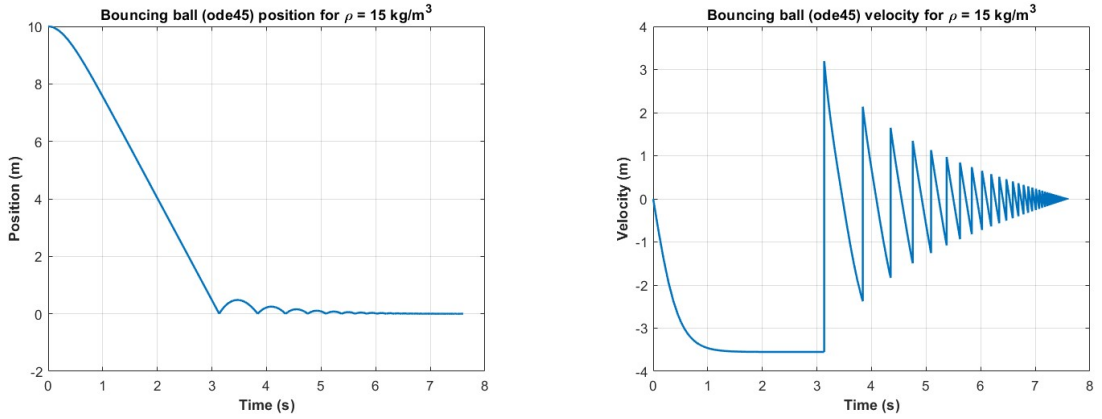
3) Fig. 23 corresponds to the solutions for $\rho = 60$ $[kg/m^3]$, demonstrating that within the given time ($t \in [0, 10)$s), the ball does not reach the ground to bounce. The velocity converges to a fixed value as the ball reaches a force equilibrium due to the dominance of the drag force.

4) Point 3) was repeated using RK4 for h = 3. The solution does not converge because $h > h_{max}$. The solution starts to converge for $h < 0.7$. The implementation of RK4 can be found in the attached Matlab code.



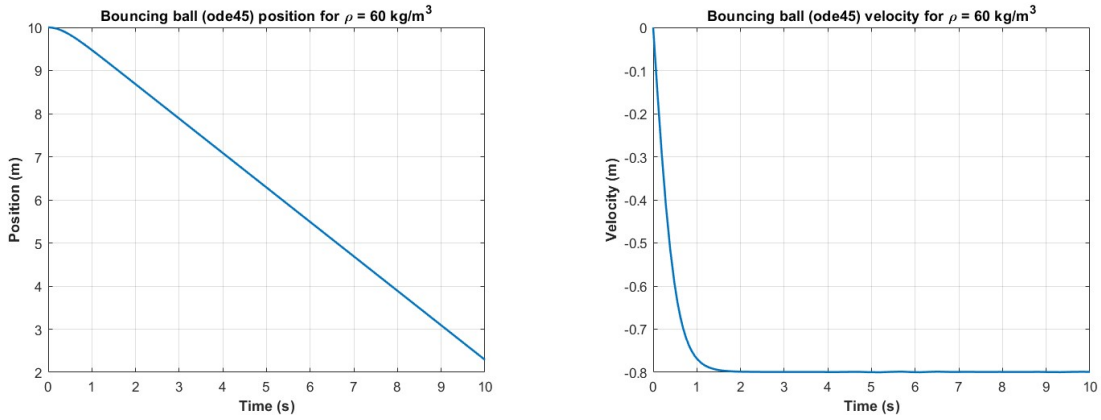
(a) Position (b) Velocity

Figure 21: Ball position and velocity vs time.



(a) Position (b) Velocity

Figure 22: Ball position and velocity vs time.



(a) Position (b) Velocity

Figure 23: Ball position and velocity vs time.