MSAS – Assignment #1: Simulation

Marcello Pareschi, 252142

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?

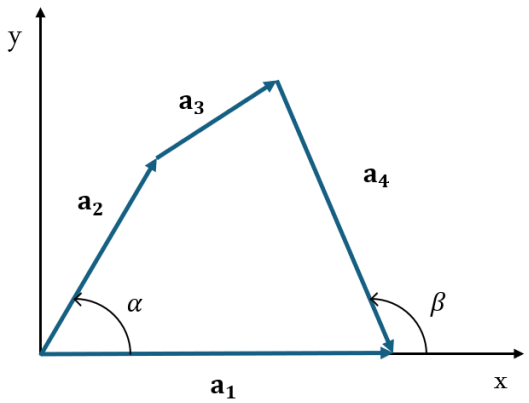


Figure 1: Kinematic chain.

(4 points)

Acronyms

BE Backward Euler

$BI2_\theta$ Second order backinterpolation methods

BRK2 Backward Heun's

FD Finite Differences

FE forward Euler's method

IEX4 Implicit Extrapolation Method of order 4

LTI linear time-invariant

KVL Kirchhoff's Voltage Law

LPA lumped parameters approach

NM Newton Method

NS Newton Solver

RHS right-hand-side

RK2 Heun's method

RK4 4^{th} order Runge-Kutta method

RK2 Heun's method

RK4 $\mathbf{4^{th}}$ order Runge-Kutta method

1.1 Point 1

The Newton Method (NM) is an iterative numerical technique to find the roots of a nonlinear equation of the form $f(x) = 0$. Given an initial guess x_0, the root estimate at iteration $k+1$ is computed as a function of x_k according to Equation 2:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)} \tag{2}$$

where $f'(x)$ is the first derivative of $f(x)$. This iterative procedure continues until a convergence criterion is met. The implemented Newton Solver (NS) stops the iteration process when the absolute value of the difference between two successive iterates falls below a user-specified tolerance.
In general, NM does not converge for all possible values of x_0 but only for those within a suitable neighborhood of a root of $f(x)$. Furthermore, if a stationary point (where $f'(x) = 0$) is encountered, the method fails, as evaluating Equation 2 at such a point results in a division by zero. Geometrically, NM approximates $f(x)$ by its tangent line at x_k and determines the next estimate x_{k+1} as the x-intercept of this tangent. At a stationary point, the tangent is horizontal and does not intersect the x-axis, leading to failure.
NM exhibits quadratic convergence, meaning that the error at step $k+1$ is proportional to the square of the error at step k, resulting in rapid convergence when sufficiently close to the root [1].

1.2 Point 2

Once the NS was implemented, the roots of the Freudenstein equation were found for $\beta \in \left[0, \frac{2}{3}\pi\right]$. For each value of β within this interval, the corresponding α was calculated using the NS, starting from two different initial guesses: $\alpha_0^{(1)} = -0.1$ rad and $\alpha_0^{(2)} = \frac{2}{3}\pi$ rad, respectively. The expressions for the function $f(\alpha)$, whose roots are sought, and its derivative $f'(\alpha)$, are given in Equation 3.

$$\begin{aligned} f(\alpha) &= \frac{a_1}{a_2}\cos\beta - \frac{a_1}{a_4}\cos\alpha - \cos(\beta-\alpha) + \frac{a_1^2 + a_2^2 - a_3^2 + a_4^2}{2a_2a_4}, \\ f'(\alpha) &= \frac{a_1}{a_4}\sin\alpha - \sin(\beta-\alpha). \end{aligned} \tag{3}$$

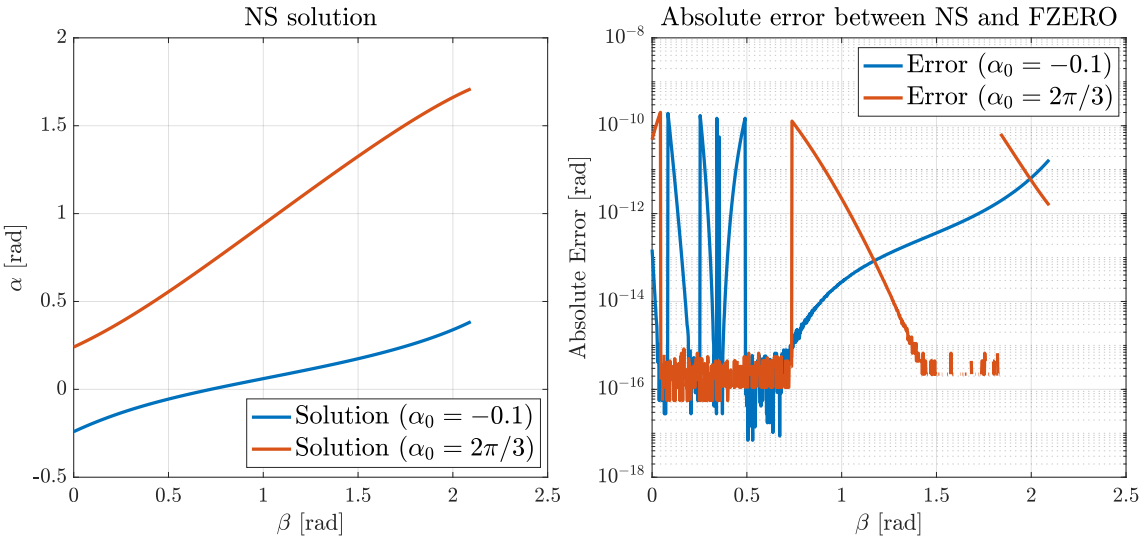


Figure 2: Solutions computed with the NS (left) and absolute error with respect to FZERO (right) for initial guesses $\alpha_0 = -0.1$ rad and $\alpha_0 = \frac{2}{3}\pi$

In Figure 2 are illustrated the results, obtained using in the NS a stopping criterion based on the differences between successive iterates ($|\alpha_{k+1} - \alpha_k| < tol$) and setting the tolerance to 10^{-5}.

The plot on the left in Figure 2 illustrates that the solution obtained by the NS strongly depends on the initial condition α_0. In both cases, α exhibits a monotonic increase with β; however, for the same value of β, the equilibrium point found by the NS varies depending on the initial condition α_0. This behavior arises because the NM converges to the root closest to the initial guess α_0. If the function has multiple zeros within the search domain, it is expected that different initial guesses α_0 will lead to different solutions. To validate the results, MATLAB's built-in function `fzero` was used to solve the Freudenstein equation under the same conditions.

The graph on the right in Figure 2 shows the absolute error between the solutions obtained with NS and `fzero` for each value of β. It can be seen that the discrepancy between the NS and the `fzero` solution remains small for the whole β interval analyzed, except for some peaks which in any case remain well below the specified tolerance of 10^{-5}, indicating that the NS provides a good approximation to the solution found by `fzero`.

1.3 Point 3

The NM can also be implemented without specifying the analytical derivative of the function describing the equation to be solved, approximating it numerically using a proper Finite Differences (FD) scheme. It was chosen to approximate $f'(\alpha)$ using a centered scheme as it provides a second-order approximation of $f'(\alpha)$ with respect to $\Delta\alpha$:

$$f'(\alpha) = \frac{f(\alpha + \Delta\alpha) - f(\alpha - \Delta\alpha)}{2\Delta\alpha} + \mathcal{O}(\Delta\alpha^2) \tag{4}$$

The choice of $\Delta\alpha$ is crucial: a too large stepsize $\Delta\alpha$ increases the truncation error, while a too small $\Delta\alpha$ exacerbates round-off errors due to finite machine precision. To balance these effects, it was adopted the heuristic

$$\Delta\alpha_k = \max\left(|\alpha_k|\sqrt{\varepsilon}, \sqrt{\varepsilon}\right) \tag{5}$$

where ε is the machine epsilon $\left(\approx 2.2 \cdot 10^{-16}\right.$ for double precision$\left.\right)$ and α_k is the value of α at the k-th Newton iteration. This choice ensures that $\Delta\alpha_k$ scales with $|\alpha|$ when α is large, while enforcing a minimum stepsize of $\sqrt{\varepsilon}$ when α approaches zero, thereby reducing the susceptibility to numerical noise [2].

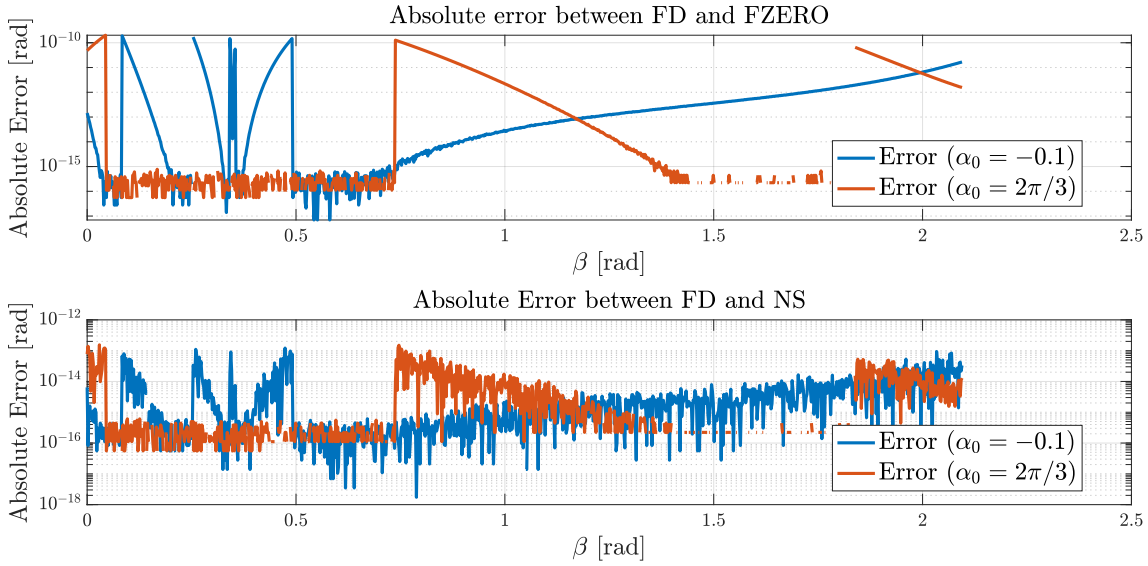


Figure 3: Absolute error between FD and FZERO (top) and NS vs FD (bottom)

The graph at the bottom of Figure 3 shows the absolute error between the solution obtained using the analytical derivative and the one obtained by approximating the derivative using finite differences. The discrepancy between the two solutions remains below 10^{-12}, indicating that the accuracy of the NM is not significantly affected by the use of FD for derivative approximation. This high accuracy can be attributed to the smoothness of the Freudenstein function within the given interval, which ensures that FD approximates the derivative with sufficient precision. Additionally, adjusting the perturbation step size based on α_k, as described earlier, further enhances the accuracy of the method by optimizing the balance between truncation and round-off errors.

It is then possible to analyze how the performance of the implemented NM varies when using the analytical derivative in NS versus approximating it with finite differences. From the graph at the top of Figure 3, it can be observed that the absolute error obtained with FD is of the same order of magnitude as the one obtained with the analytical derivative, confirming the reliability of the finite difference approach.

1.4 Point 4

For specific values of β, an analytical solution to Equation 3 can be found, providing a useful reference to assess the accuracy of the NM. Equation 6 presents the analytical solution for the cases $\beta = 0$ and $\beta = \pi$:

$$\alpha = \begin{cases} \pm\cos^{-1}\left(\dfrac{a_1^2 + a_2^2 - a_3^2 + a_4^2 + 2a_1a_4}{2a_2a_4\left(1 + \dfrac{a_1}{a_4}\right)}\right), & \text{if} \quad \beta = 0 \\ \pm\cos^{-1}\left(-\dfrac{a_1^2 + a_2^2 - a_3^2 + a_4^2 - 2a_1a_4}{2a_2a_4\left(1 - \dfrac{a_1}{a_4}\right)}\right), & \text{if} \quad \beta = \pi \end{cases} \tag{6}$$

In this specific case, since $a_1 = a_4$, the solution for $\beta = \pi$ is not physically meaningful, as the denominator in the second equation approaches zero, leading to an undefined expression. Consequently, only the solution for $\beta = 0$ is considered.
For $\beta = 0$, the analytical solutions are given by $\alpha_{1,2} \approx \pm 0.2408$ [rad]. Table 1 reports the absolute error obtained using the Newton method with the analytical derivative (NS) and with finite differences (FD) for both α_1 and α_2. The observed errors are very small and fall well below the specified tolerance of 10^{-5} both for NS and FD, demonstrating the accuracy and robustness of the implemented NM.

Table 1: Maximum error of NM with the analytical derivative (NS) and with finite differences (FD)

Initial guess [rad]	$\boldsymbol{\alpha_{\text{anal}}}$ **[rad]**	**NS Error [rad]**	**FD Error [rad]**
-0.1	-0.2408	$1.515454 \cdot 10^{-13}$	$1.451339 \cdot 10^{-13}$
$\frac{2}{3}\pi$	0.2408	$4.838660 \cdot 10^{-11}$	$4.830952 \cdot 10^{-11}$

1.5 Point 5

Finally, the procedure described in subsection 1.2 was repeated, extending the interval of β to $[0,\ \pi]$. Figure 4 shows that the solution initially behaves as expected; however, it starts

to diverge once a certain value of $\beta_{\max} \approx 2.6364$ [rad] is reached. This divergence occurs because, beyond the limit value $\beta_{\max}$, the function $f(\alpha)$ described in Equation 3 enters a region where no real zeros exist. As a result, Equation 1 has no solutions in that region. This concept is illustrated more clearly in Figure 5, which depicts the contour plot of $f(\alpha, \beta)$. The figure highlights the *zero-level* of $f(\alpha, \beta)$, representing the locus of solutions to Freudenstein's equation. It can be observed that in the region of the plane where $\beta > \beta_{\max}$, no solutions exist, confirming the absence of valid roots beyond this threshold.

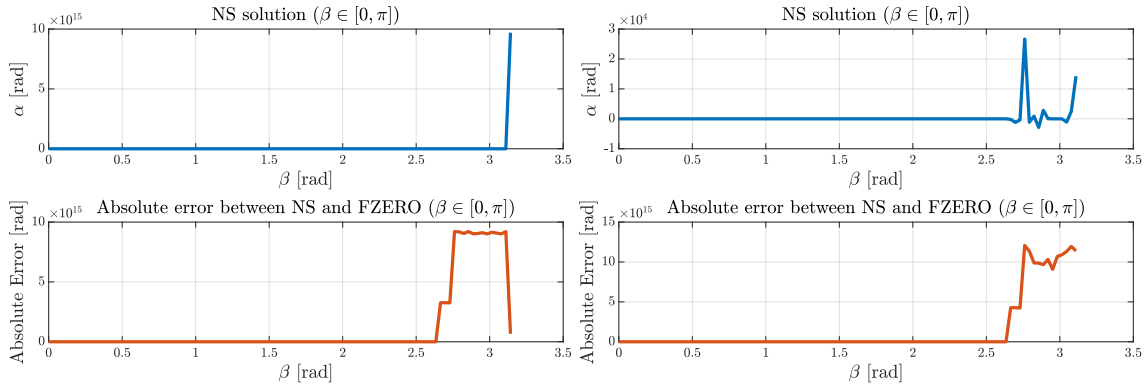


Figure 4: Absolute error between FD and FZERO (top) and NS vs FD (bottom)

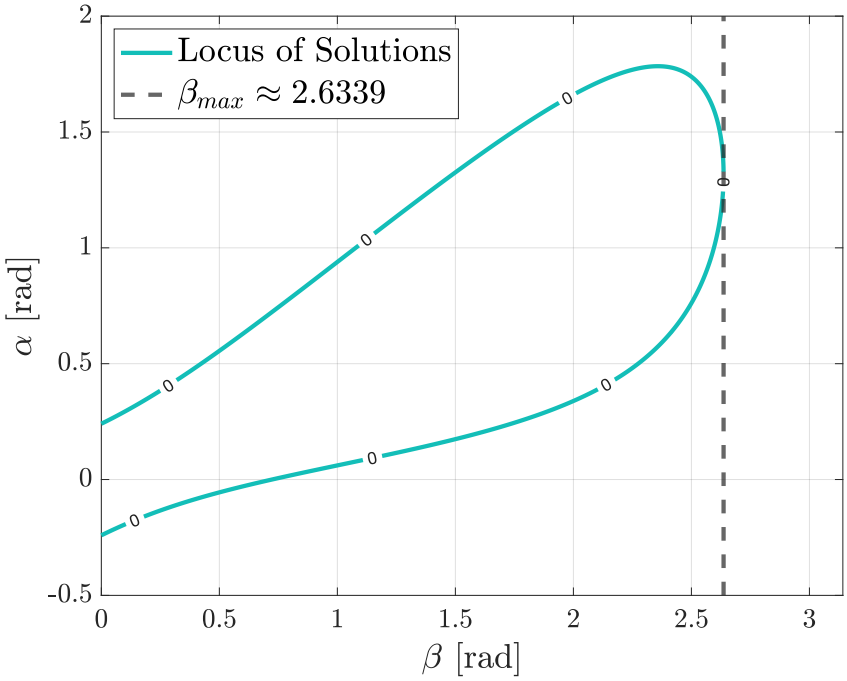


Figure 5: Locus of solutions to Freudenstein's equation

2 Numerical solution of ODE

Exercise 2

The physical model of a moving object with decreasing mass is shown in Fig. 6. The mathematical model of the system is expressed by the equations:

$$m(t)\frac{dv}{dt} = \overline{F} + f(t) - \alpha \cdot v \tag{7}$$

$$m(t) = m_0 - c_m \cdot t \tag{8}$$

$$\overline{F} = -\ 0.5 \cdot \rho \cdot C_d \cdot A_m \cdot v^2 \tag{9}$$

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. (7) has the exact solution:

$$v(t) = \frac{f(t)}{\alpha} - \left[\frac{f(t)}{\alpha} - v(0)\right]\left[1 - \frac{c_m \cdot t}{m_0}\right]^{\frac{\alpha}{c_m}} \tag{10}$$

Answer to the following tasks:

1) Implement a general-purpose, fixed-step Heun's method (RK2);
2) Solve Eq. (7) 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. (7) 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. (7) ;
3) Discuss the results.

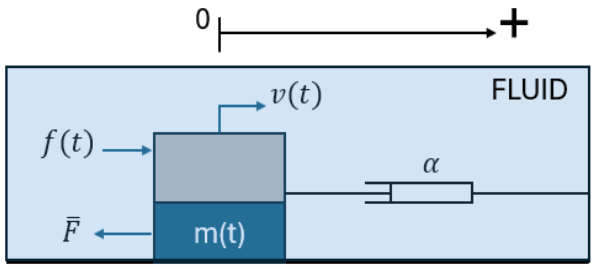


Figure 6: Moving object with decreasing mass.

(5 points)

2.1 Point 1

Heun's method (RK2) is a numerical integration technique employed to approximate the solution of the Cauchy problem in the form:

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{f}(t, \mathbf{x}) \\ \mathbf{x}(t_0) = \mathbf{x}_0 \end{cases} \tag{11}$$

RK2 can be interpreted as a predictor-corrector technique, which means it uses an initial prediction of the solution at the next time step and then corrects this prediction to achieve better accuracy. The prediction step uses the forward Euler's method (FE) to make an initial estimate of the solution at the next time step $\left(\mathbf{x}^*_{k+1}\right)$. The correction step refines the initial prediction by averaging the derivatives at the current point $(t_k, \mathbf{x}_k)$ and the predicted point $\left(t_{k+1}, \mathbf{x}^*_{k+1}\right)$ [1]. The RK2 algorithm is reported in Equation 12:

$$\begin{aligned} &\text{prediction} \quad \text{step}: \quad \mathbf{x}^*_{k+1} = \mathbf{x}_k + h\mathbf{f}(t_k, \mathbf{x}_k) \\ &\text{correction} \quad \text{step}: \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left[\mathbf{f}(t_k, \mathbf{x}_k) + \mathbf{f}\left(t_{k+1}, \mathbf{x}^*_{k+1}\right)\right] \end{aligned} \tag{12}$$

2.2 Point 2

The general-purpose RK2 described in subsection 2.1 was implemented in MATLAB and used to solve the Cauchy problem described in the text of the exercise in the case of $\rho = 0 \left[\text{kg/m}^3\right]$. In this specific case, the right-hand-side (RHS) of Equation 11 assumes the form

$$\mathbf{f}(t, \mathbf{x}) = \begin{bmatrix} \dfrac{1 - \alpha \cdot v}{m} \\ -c_m \end{bmatrix} \tag{13}$$

where $\mathbf{x}(t) = \begin{bmatrix} v(t) & m(t) \end{bmatrix}^T$ is the state vector. In this case the ODE describing the system dynamics admits an analytical solution in terms of $v(t)$ (Equation 10), allowing to analyze the performances of the implemented RK2 against the analytical solution for different values of the step size h. The results are displayed in Figure 7. From the plots it can be observed that

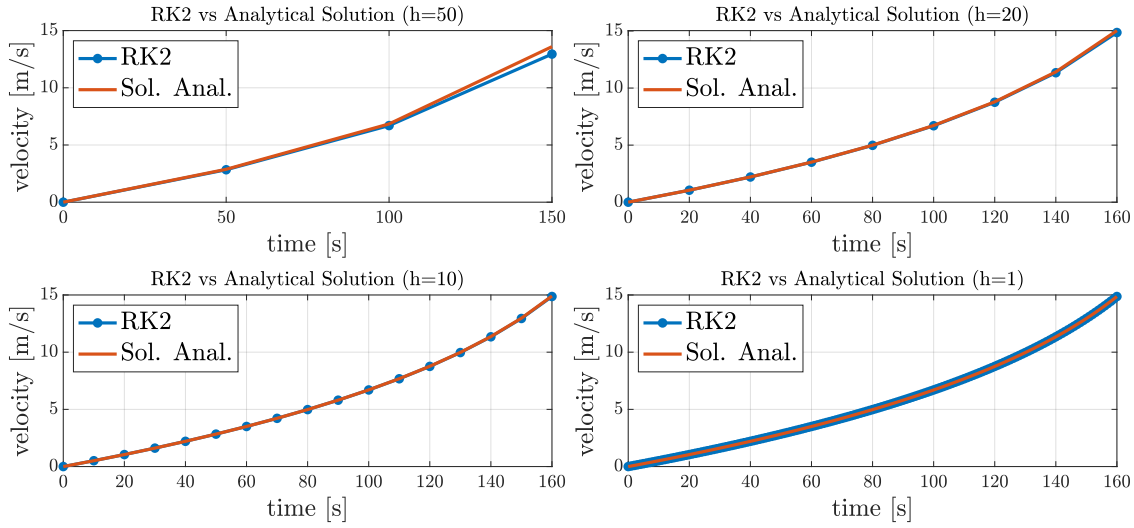


Figure 7: RK2 solution for the $\rho = 0 \left[\text{kg/m}^3\right]$ case

the numerical solution follows the analytical one for all the analyzed time steps, highlighting the numerical stability of RK2 for the analyzed problem. From the graph corresponding to

a timestep of $h = 50$ [s] it can be noted a discrepancy between analytical and numerical solution at the final time, suggesting a loss of accuracy of RK2 over time. The accuracy of RK2 can be better analyzed looking at Figure 8, which depicts the time history of the absolute error $e(t) = |v_{\text{anal}} - v_{\text{num}}|$. The graph is presented on a semi-logarithmic scale for improved visualization. As expected, larger time steps result in larger integration errors. This occurs because larger step sizes reduce the number of evaluation points along the solution curve, limiting the method's ability to accurately capture the system's dynamics. It can be also noted that, regardless of the timestep, errors tend to accumulate over time, leading to a progressive loss of accuracy in the numerical solution.

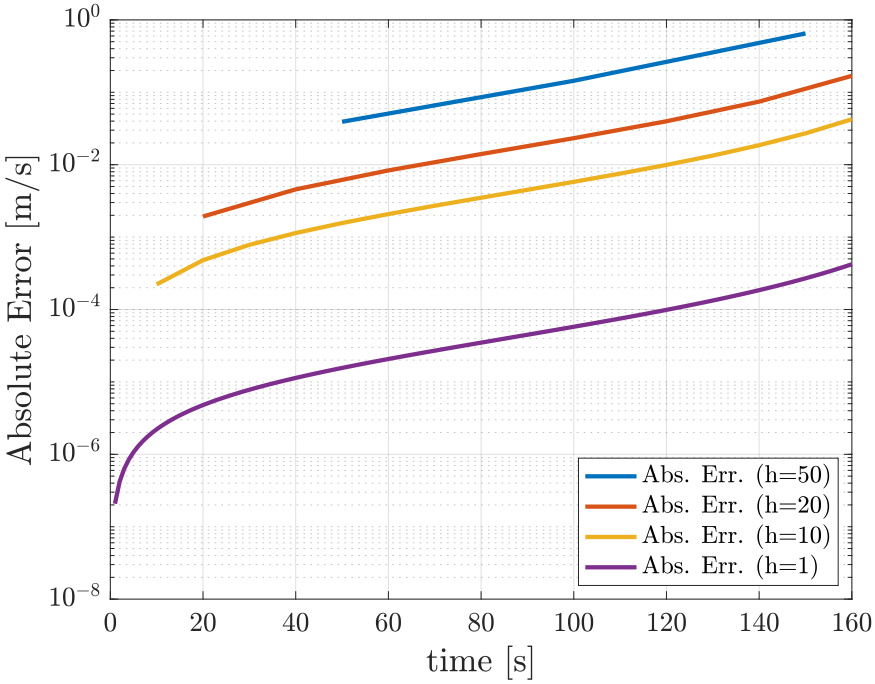


Figure 8: Time history of the absolute error between the exact and numerical solution computed with RK2

2.3 Point 3

Better accuracy can be achieved by utilizing higher-order methods, such as the $\mathbf{4^{th}}$ order Runge-Kutta method (RK4), an explicit method that requires four function evaluations per timestep. As a result, although the error for the same timestep is lower compared to the RK2, which is only second-order accurate, the computational cost is higher due to the need for four function evaluations instead of two. The implemented general-purpose RK4 method is presented in Equation 14 [1].

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{6}\left(\mathbf{K}_1 + 2\mathbf{K}_2 + 2\mathbf{K}_3 + \mathbf{K}_4\right) \tag{14}$$

where

$$\begin{aligned}
\mathbf{K}_1 &= \mathbf{f}\left(t_k, \mathbf{x}_k\right) \\
\mathbf{K}_2 &= \mathbf{f}\left(t_k + \frac{h}{2}, \mathbf{x}_k + \frac{h}{2}\mathbf{K}_1\right) \\
\mathbf{K}_3 &= \mathbf{f}\left(t_k + \frac{h}{2}, \mathbf{x}_k + \frac{h}{2}\mathbf{K}_2\right) \\
\mathbf{K}_4 &= \mathbf{f}\left(t_k + h, \mathbf{x}_k + h\mathbf{K}_3\right)
\end{aligned}$$

The same Cauchy problem solved in subsection 2.2 was also solved using RK4, and the results are shown in Figure 9. From the graphs, it can be observed that the numerical solution obtained with RK4 follows the analytical solution much more closely than the one obtained using RK2, even for large timesteps. This is because RK4, being a fourth-order method, provides higher accuracy than RK2 for the same timestep.

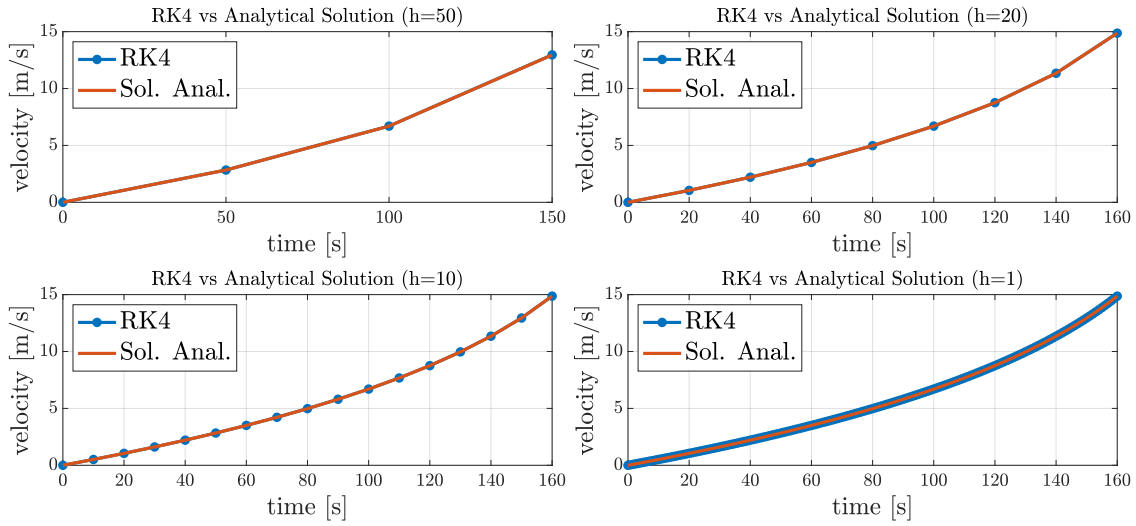


Figure 9: RK4 solution for the $\rho = 0 \left[\mathrm{kg/m^3}\right]$ case

The superior accuracy of RK4 compared to RK2 is further illustrated in Figure 10, which presents the time history of the absolute error. It is immediately evident that the error is several orders of magnitude lower than that of the RK2 solution, shown in Figure 8. Furthermore, Figure 10 confirms that the error exhibits the same monotonically increasing behavior described in subsection 2.2, resulting from the accumulation of numerical errors over time.

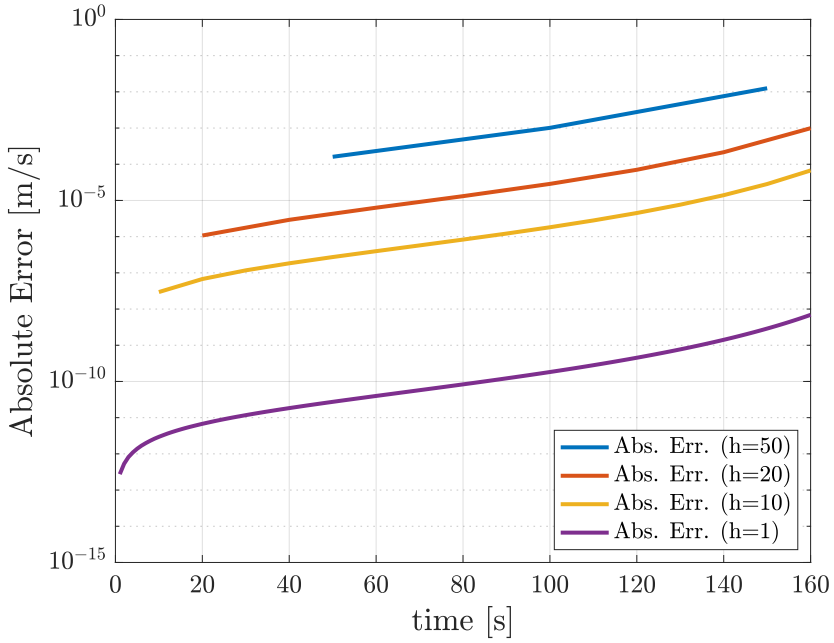


Figure 10: Time history of the absolute error between the exact and numerical solution computed with RK4

2.4 Point 4

To assess the trade-off between execution time and integration error, a *Monte Carlo* simulation consisting of 10000 runs was performed. This approach was necessary because the execution time of the two implemented algorithms is strongly influenced by external factors, as the script does not run in an isolated environment. The figures present the average execution time and integration error for all the analyzed timesteps. For each timestep, the numerical error was evaluated at the final time. This represents the maximum error of each simulation since, as observed in Figure 8 and Figure 10, the numerical error tends to accumulate over time. The outputs of the *Monte Carlo* simulation are illustrated in Figure 11, and the numerical results in terms of execution time (mean μ_t and standard deviation σ_t) and absolute error are reported in the Table 2 and Table 3.

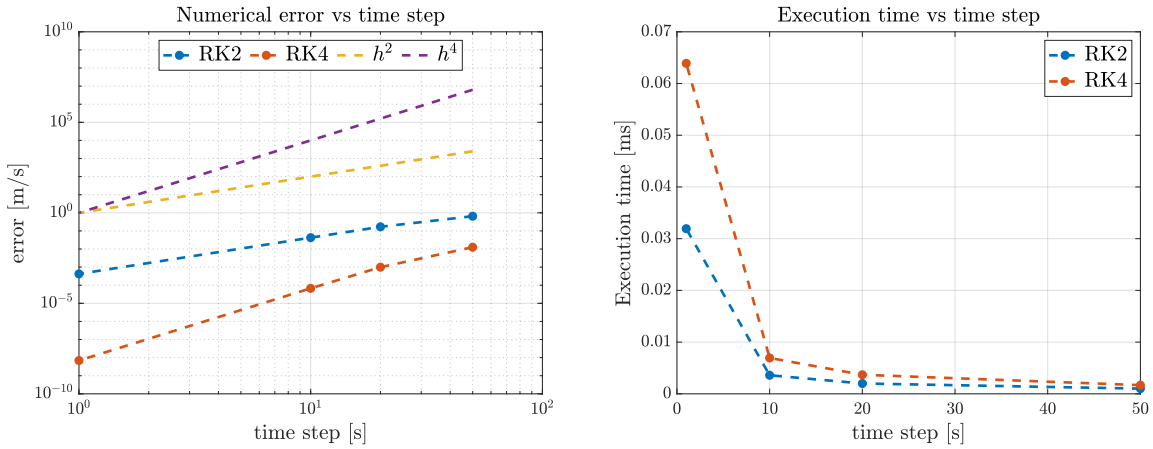


Figure 11: Numerical error (left) and execution time (right) vs time step

As expected, the execution time is higher for RK4 than for RK2 at each timestep. This occurs because RK4 requires more function evaluations per integration step, resulting in a greater computational effort. From the error graph in logarithmic scale, it can be observed that the RK2 error forms a straight line parallel to the one corresponding to h^2, while the RK4 error is parallel to h^4. This indicates that the errors of RK2 and RK4 scale with h^2 and h^4, respectively. This result is expected, as RK2 is a second-order accurate method, while RK4 is a fourth-order method.

Table 2: *Monte Carlo* simulation results for RK2

Time step [s]	μ_t [ms]	σ_t [ms]	Absolute Error [m/s]
50	$1.0058 \cdot 10^{-3}$	$1.9149 \cdot 10^{-3}$	$6.0788 \cdot 10^{-1}$
20	$2.0026 \cdot 10^{-3}$	$1.7007 \cdot 10^{-3}$	$1.6867 \cdot 10^{-1}$
10	$3.5864 \cdot 10^{-3}$	$1.1951 \cdot 10^{-3}$	$4.2406 \cdot 10^{-2}$
1	$3.1930 \cdot 10^{-2}$	$3.2983 \cdot 10^{-3}$	$4.2193 \cdot 10^{-4}$

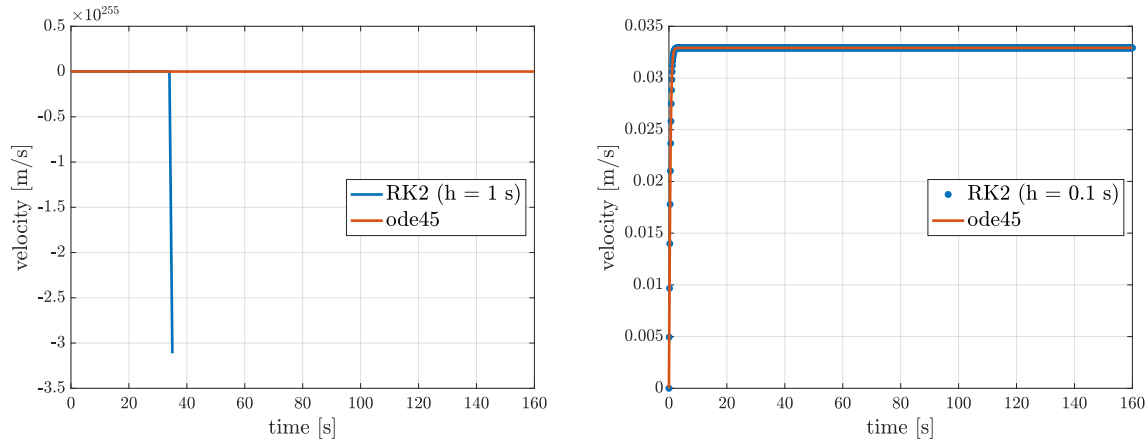
Table 3: *Monte Carlo* simulation results for RK4

Time step [s]	μ_t [ms]	σ_t [ms]	Absolute Error [m/s]
50	$1.6770 \cdot 10^{-3}$	$1.4321 \cdot 10^{-3}$	$1.2576 \cdot 10^{-2}$
20	$3.6870 \cdot 10^{-3}$	$1.4557 \cdot 10^{-3}$	$9.8964 \cdot 10^{-4}$
10	$6.9359 \cdot 10^{-3}$	$1.7537 \cdot 10^{-2}$	$6.7300 \cdot 10^{-5}$
1	$6.3907 \cdot 10^{-2}$	$5.0110 \cdot 10^{-2}$	$6.9197 \cdot 10^{-9}$

In conclusion, the best trade-off between accuracy and computational effort seems to be RK4 with a time step of 10 s: indeed it guarantees a better accuracy of RK2 even with the time step of 1 s, with an average execution time lower by one order magnitude.

2.5 Point 5

The same system analyzed in the previous sections was finally integrated using RK2 with a fixed step size of $h = 1$ s, considering a fluid density of $\rho = 900$ kg/m^3. The solution obtained with RK2 was compared to the one computed using MATLAB's built-in function `ode45`, which implements a variable-step RK5 algorithm. In this comparison, both the relative and absolute tolerances of `ode45` were set to 10^{-12}.



Unstable solution: $h = 1$ s Stable solution: $h = 0.1$ s

Figure 12: RK2 solutions for $\rho = 900$ kg/m^3 for $h = 1$ s and $h = 0.1$ s

From Figure 12, it is evident that the RK2 solution computed with a fixed timestep of $h = 1$ s diverges. This divergence occurs because accounting for a nonzero fluid density introduces a fluid-dynamic drag term into the governing ODE, which is proportional to v^2 and evolves much more rapidly than the other terms. Consequently, one of the system's eigenvalues falls outside the stability region of RK2, leading to numerical instability. To accurately integrate the system's ODE with RK2, the timestep must be reduced. Indeed, when the timestep is decreased to $h = 0.1$ s, the stability of the RK2 solution is restored, enabling it to closely follow the solution computed by `ode45`, which was used for validation.

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)

2.6 Runge-Kutta methods

2.6.1 Heun's method

The linear operator $\mathbf{F}_{RK2}(h, \alpha)$ that maps $\mathbf{x}_k$ to $\mathbf{x}_{k+1}$ for the RK2 method can be derived by substituting $\dot{\mathbf{x}} = \mathbf{A}(\alpha)\,\mathbf{x}$ into the general-purpose RK2 algorithm given in Equation 12. After some straightforward algebraic manipulations, the expression for $\mathbf{F}_{RK2}(h, \alpha)$ is obtained as:

$$\mathbf{F}_{RK2}(h, \alpha) = \mathbf{I}_{2\times 2} + h\,\mathbf{A}(\alpha) + \frac{h^2}{2}\,\mathbf{A}^2(\alpha). \tag{15}$$

Thus, the continuous-time system $\dot{\mathbf{x}} = \mathbf{A}(\alpha)\,\mathbf{x}$ is discretized as $\mathbf{x}_{k+1} = \mathbf{F}_{RK2}(h, \alpha)\,\mathbf{x}_k$. The discrete-time solution is stable if $|\sigma_i| < 1$ for $i = 1, 2$, where σ_i are the eigenvalues of $\mathbf{F}_{RK2}(h, \alpha)$. It can be shown that if $\mathbf{A}(\alpha)$ is diagonalizable, then the eigenvalues σ_i and the eigenvalues λ_i of $\mathbf{A}(\alpha)$ are related by

$$\sigma_i = 1 + h\,\lambda_i + \frac{h^2}{2}\,\lambda_i^2, \quad i = 1, 2. \tag{16}$$

In this particular case, the two eigenvalues of $\mathbf{A}(\alpha)$ are complex conjugates lying on the unit circle [3]:

$$\lambda_{1,2}(\alpha) = cos(\alpha) \pm \mathrm{j} sin(\alpha) = e^{\pm \mathrm{j}\alpha}. \tag{17}$$

Substituting Equation 17 into Equation 16 yields a pair of complex conjugate eigenvalues for $\mathbf{F}_{RK2}$. Since complex conjugate pairs share the same magnitude, the stability analysis can be focused on σ_1 corresponding to $\lambda_1 = e^{\mathrm{j}\alpha}$. Thus, to ensure the stability of the discrete-time solution, it is sufficient to verify that the absolute value of the stability function

$$R(h, \alpha) = 1 + h\,e^{\mathrm{j}\alpha} + \frac{h^2}{2}\,e^{2\mathrm{j}\alpha} \tag{18}$$

satisfies $|R(h, \alpha)| < 1$. By solving the equation $|R(h, \alpha)| = 1$ for $\alpha \in [0, \pi]$, one can determine the maximum allowable step size h for each value of α that guarantees numerical stability of the RK2 method [3]. In case of $\alpha = \pi$, the stability function assumes the particularly simple form of $R(h,\ \pi) = 1 - h + \dfrac{h^2}{2}$, which is a real function, always positive. Hence, the equation $|R(h, \pi)| = 1$ admits two real solutions: the trivial one $h_0 = 0$ and $h_{max}^{(\pi)} = 2$. Once $h_{max}^{(\pi)}$ was calculated, h_{max} was computed for $\alpha \in [0, \pi)$ by solving Equation 18 for decreasing values of α. This iterative procedure was implemented using MATLAB's `fsolve` function, with a `FunctionTolerance` set to 10^{-12}, and using the value of $h_{\max}$ from the previous iteration as the initial guess for the next. Figure 13 illustrates the computed h_{max} for each value of α. It

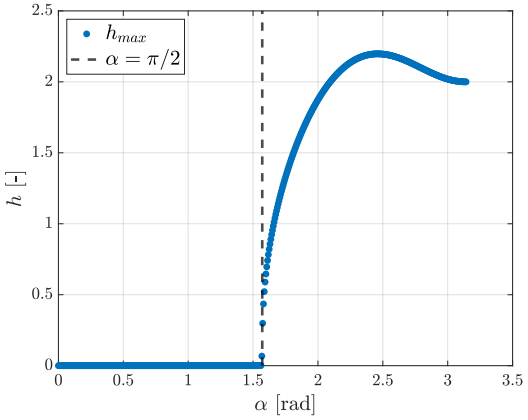


Figure 13: h_{max} vs α for RK2

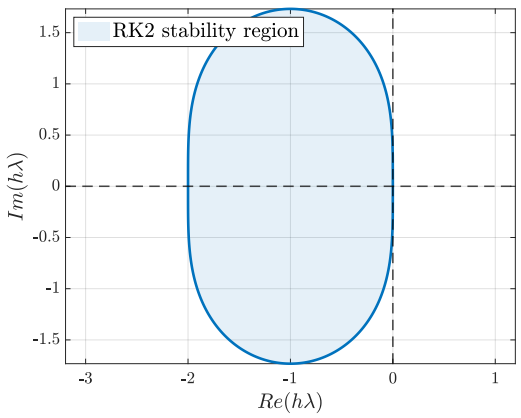


Figure 14: RK2 stability region

can be observed that for $\alpha \in [0, \pi/2]$ [rad], the solver always converges to zero. This behavior is expected because, in this interval of α, the eigenvalues of $\mathbf{A}(\alpha)$ have a real part greater than zero. Consequently, there is no $h > 0$ such that RK2 maps unstable eigenvalues of the continuous-time system into stable eigenvalues of the discrete-time system. The stability region shown in Figure 14 was obtained by plotting $h_{max}(\alpha) \cdot \lambda_{1,2}(\alpha)$ on the complex plane. Since RK2 is an explicit method, its stability region corresponds to the area inside the resulting curve.

2.6.2 4^{th} order Runge-Kutta method

The same stability analysis conducted in subsubsection 2.6.1 can also be applied to RK4. The linear operator $\mathbf{F}_{RK4}$ that maps $\mathbf{x}_k$ into $\mathbf{x}_{k+1}$ can be derived by substituting the RHS of the linear ODE system into Equation 14. After some straightforward algebraic manipulations, the expression for $\mathbf{F}_{RK4}$ in Equation 19 is obtained. It is noteworthy that this expression matches the matrix exponential up to order four, which is expected since RK4 is fourth-order accurate with respect to h [3].

$$\mathbf{F}_{RK4}(h, \alpha) = \mathbf{I}_{2\times 2} + h\,\mathbf{A}(\alpha) + \frac{h^2}{2}\mathbf{A}^2(\alpha) + \frac{h^3}{6}\mathbf{A}^3(\alpha) + \frac{h^4}{24}\mathbf{A}^4(\alpha). \tag{19}$$

The same reasoning regarding the eigenvalues of $\mathbf{F}_{RK4}$ as in subsubsection 2.6.1 can be applied here. Since $\mathbf{A}(\alpha)$ is a diagonalizable matrix whose eigenvalues form a complex-conjugate pair lying on the unit circle centered at the origin, we introduce the stability function:

$$R_{RK4}(h, \alpha) = 1 + h\,e^{\mathrm{j}\alpha} + \frac{h^2}{2}e^{2\mathrm{j}\alpha} + \frac{h^3}{6}e^{3\mathrm{j}\alpha} + \frac{h^4}{24}e^{4\mathrm{j}\alpha}. \tag{20}$$

The stability of the discrete-time solution is ensured if $|R_{RK4}(h, \alpha)| < 1$. In the particular case where $\alpha = \pi$ [rad], the stability function simplifies into a fourth-order polynomial dependent only on the real parameter h. This polynomial is positive for all $h \in \mathbb{R}$, thus the stability equation $|R_{RK4}(h, \alpha)| - 1 = 0$, which is used to determine the maximum allowable h, reduces to:

$$h\left(-1 + \frac{h}{2} - \frac{h^2}{6} + \frac{h^3}{24}\right) = 0. \tag{21}$$

Equation 21 admits two real roots: the trivial solution at $h = 0$ and $h_{max}^{(\pi)} \approx 2.78529$, which can be determined analytically using the general formula for solving cubic polynomial equations. The value of h_{max} was then computed for $\alpha \in [0, \pi)$ by gradually decreasing α and solving the stability equation numerically using MATLAB's `fsolve`, with the initial guess taken as the previously computed h_{max}.

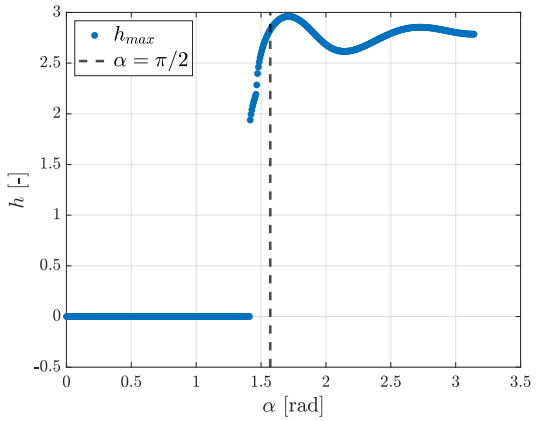


Figure 15: h_{max} vs α for RK4

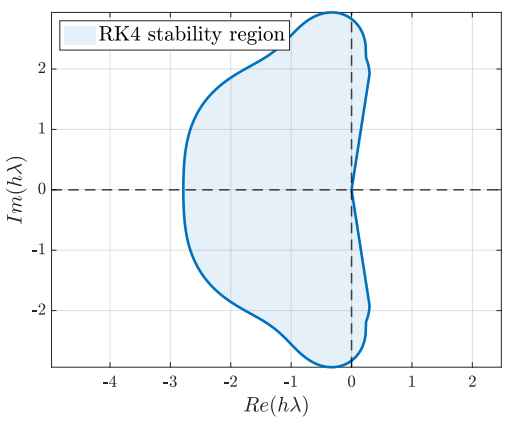


Figure 16: RK4 stability region

Figure 15 and Figure 16 illustrate the computed values of h_{max} for each value of α and the corresponding stability region for RK4. It can be observed that within the region $\alpha \in [0, \pi/2]$, unstable eigenvalues of the continuous-time system may be mapped into the stability region of RK4, particularly for values of α sufficiently close to $\pi/2$ rad. Additionally, it can be noted that beyond a certain threshold, approximately 1.4858 rad, MATLAB's `fsolve` struggles to find solutions below the specified tolerance due to the shape of the objective function. This results in an inaccurate representation of the stability region.

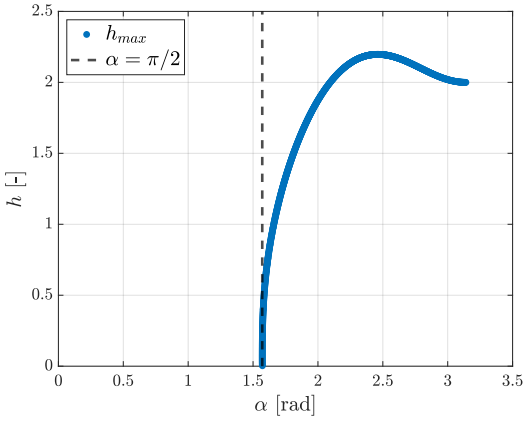


Figure 17: h_{max} vs α for RK2

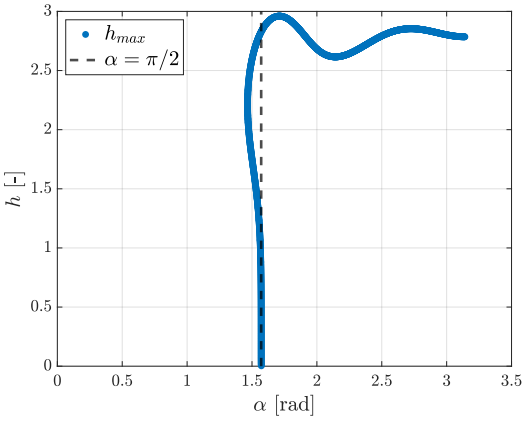


Figure 18: h_{max} vs α for RK4

To address this issue and accurately depict the stability region of RK4 without requiring an excessively fine grid of α values, an alternative approach was employed. Instead of solving for h_{max} directly, the contour line of $|R_{RK4}(h, \alpha)|$ at level 1 was plotted in the complex plane, representing the locus of solutions to the stability equation. For each point on the obtained curve, the modulus and phase were extracted, corresponding to h_{max} and α, respectively.

This procedure was also applied to RK2, enabling a more accurate representation of the stability regions of both methods along with the maximum allowable step size for each value of α. The obtained results are illustrated in Figure 17 and Figure 18. In Figure 19, a more accurate representation of the stability regions of RK2 and RK4 is illustrated. It can still be observed that for specific values of α, slightly greater than $\pi/2$ rad, unstable eigenvalues of the continuous-time system are mapped into the stability region of RK4. Additionally, it is evident that the stability domain of RK4 is larger than that of RK2, allowing for numerical stability to be maintained even with larger step sizes. Finally, it can be noted that the stability region of RK4 more closely follows the imaginary axis, enabling a better representation of marginally stable systems [3].

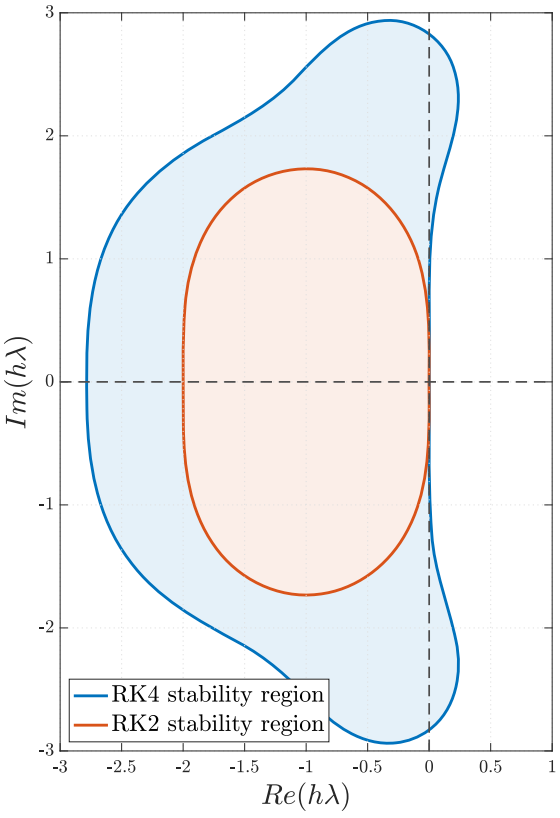


Figure 19: Accurate representation of RK2 and RK4 stability region

2.7 Backinterpolation methods

Second order backinterpolation methods ($BI2_\theta$) form a class of ODE integrators that leverage a combination of the explicit RK2 and implicit Backward Heun's (BRK2) methods. The key idea is that the state vector $\mathbf{x}_{k+\theta}$ at $t_{k+\theta} = t_k + \theta h$, with $\theta \in [0, 1]$, can be obtained either by integrating forward in time with a positive timestep of θh or by integrating backward with a negative timestep of $-(1-\theta)\,h$ [3]:

$$\begin{aligned}
\mathbf{x}_{k+\theta}^{\text{expl}} &= \mathbf{x}_k + \frac{\theta h}{2}\left[\mathbf{f}\left(t_k, \mathbf{x}_k\right) + \mathbf{f}\left(t_{k+\theta}, \mathbf{x}_k + \theta h \mathbf{f}\left(t_k, \mathbf{x}_k\right)\right)\right], \\
\mathbf{x}_{k+\theta}^{\text{impl}} &= \mathbf{x}_{k+1} - \frac{(1-\theta)\,h}{2}\left[\mathbf{f}\left(t_{k+1}, \mathbf{x}_{k+1}\right) + \mathbf{f}\left(t_{k+\theta}, \mathbf{x}_{k+1} - (1-\theta)\,h\mathbf{f}\left(t_{k+1}, \mathbf{x}_{k+1}\right)\right)\right].
\end{aligned} \tag{22}$$

By adjusting the parameter θ, the stability domain of the method can be modified. In particular, for $\theta < 0.5$, a family of second-order accurate A-stable methods can be obtained. The linear operator $\mathbf{F}_{BI2_\theta}(h, \alpha, \theta)$, which governs the dynamics of the discrete-time system, can be derived by substituting the right-hand side of the linear time-invariant (LTI) system into Equation 22, setting $\mathbf{x}_{k+\theta}^{\text{impl}} = \mathbf{x}_{k+\theta}^{\text{expl}}$, and performing algebraic manipulations.

$$\mathbf{F}_{BI2_\theta}(h, \alpha, \theta) = \left[\mathbf{I}_{2\times 2} - (1-\theta)\; h\,\mathbf{A}(\alpha) + \frac{(1-\theta)^2}{2} h^2\,\mathbf{A}^2(\alpha)\right]^{-1} \left[\mathbf{I}_{2\times 2} + \theta\, h\,\mathbf{A}(\alpha) + \frac{\theta^2}{2} h^2 \mathbf{A}^2(\alpha)\right] \tag{23}$$

The same reasoning regarding the relationship between the eigenvalues of $\mathbf{F}_{BI2_\theta}(h, \alpha)$, which govern the discrete-time dynamics, and those of $\mathbf{A}(\alpha)$, which describe the continuous-time dynamics, as discussed in subsection 2.6, also applies in this case. Consequently, a stability function for $BI2_\theta$ can be introduced. The numerical solution remains stable if the stability function satisfies the condition $|R_{BI2_\theta}(h, \alpha)| < 1$:

$$R_{BI2_\theta}(h, \alpha, \theta) = \frac{1 + \theta\, h\, e^{\mathrm{j}\alpha} + \dfrac{\theta^2}{2} h^2 e^{2\mathrm{j}\alpha}}{1 - (1-\theta)\; h\, e^{\mathrm{j}\alpha} + \dfrac{(1-\theta)^2}{2} h^2 e^{2\mathrm{j}\alpha}} \tag{24}$$

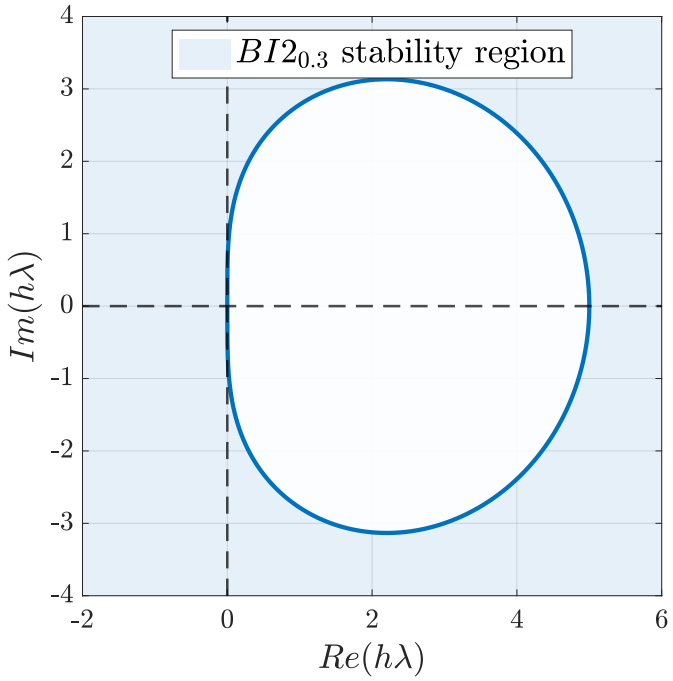


Figure 20: Stability region of $BI2_{0.3}$

In Figure 20, which was obtained by numerically solving the equation $|R_{BI2_{0.3}}(h,\alpha)| - 1 = 0$ for $\alpha \in [0, \pi]$, the stability region of $\text{BI2}_{0.3}$ is illustrated. It can be observed that the method is stable outside the obtained curve. This behavior is expected since $\theta < 0.5$, making the method closer to a fully implicit one, which would be obtained by setting $\theta = 0$. Furthermore, it can be noted that the entire left half-plane is part of the stability region of the method. Indeed, as previously mentioned, $BI2_\theta$ methods are A-stable for $\theta < 0.5$ [3]. This result makes $\text{BI2}_{0.3}$ particularly suitable for integrating stiff systems, where the eigenvalues have significantly different absolute values.

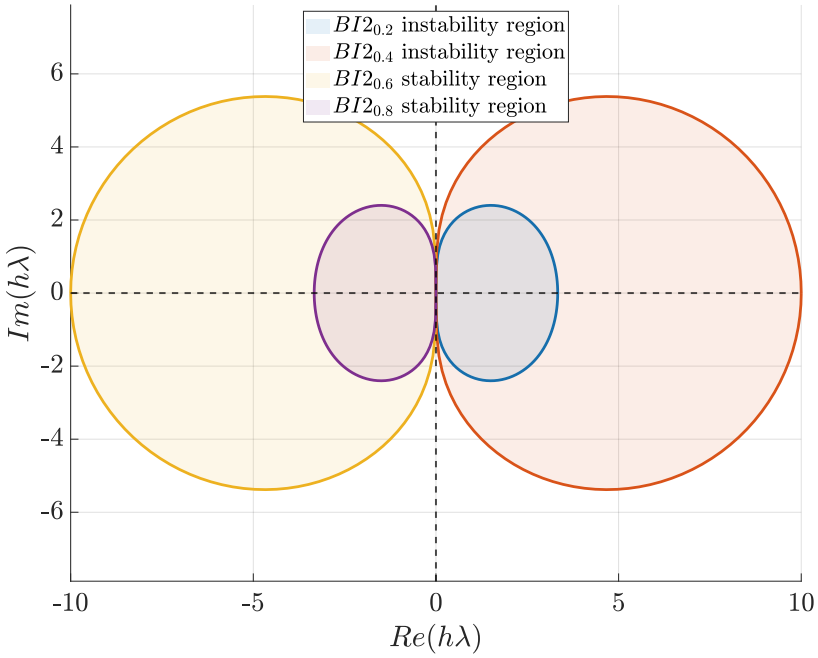


Figure 21: Stability regions of $BI2_\theta$ for different values of θ

Figure 21 illustrates the stability domains for $\theta = 0.2,\ 0.4,\ 0.6,$ and 0.8. It can be observed that the entire left half-plane lies within the stability region of $\text{BI2}_{0.2}$ and $\text{BI2}_{0.4}$, confirming the A-stability property of $BI2_\theta$ methods for $\theta < 0.5$, as previously discussed. Additionally, it can be noted that the stability curves for $\theta = 0.4$ and $\theta = 0.6$ are larger than those for $\theta = 0.2$ and $\theta = 0.8$. This occurs because $\theta = 0.4$ and $\theta = 0.6$ are closer to the critical value $\theta = 0.5$, which results in a stability region that encompasses the entire left half-plane while the right half-plane remains unstable [2]. Furthermore, $\text{BI2}_{0.2}$ and $\text{BI2}_{0.8}$ exhibit stability regions more similar to those of RK2 and BRK2, highlighting their closer resemblance to these methods [2].

Exercise 4

The cooling problem of a high-temperature mass is shown in Fig. 22. 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).

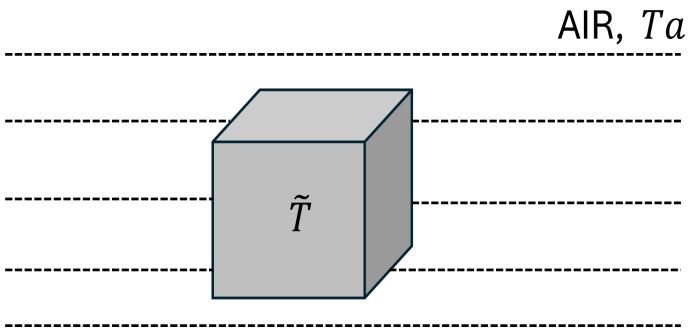


Figure 22: The cooling of high-temperature mass.

(4 points)

2.8 Mathematical model

The cooling problem under analysis was modeled using the lumped parameters approach (LPA), i.e., assuming that spatial thermal gradients inside the mass are negligible. No internal energy sources are present within the mass, meaning that energy exchange occurs only through convective and radiative heat transfer. Furthermore, the thermal capacity of the surrounding air is assumed to be significantly larger than that of the mass, ensuring that the air temperature remains constant throughout the entire process. Under these assumptions, the governing equation is obtained by applying the energy balance to the mass:

$$\begin{cases} \dot{\widetilde{T}}\left(t, \widetilde{T}\right) = -\dfrac{K_c}{C}\,\widetilde{T}\,(t) - \dfrac{K_r}{C}\,\widetilde{T}^4\,(t) + \dfrac{1}{C}\left(K_c\,T_a + K_r\,T_a^4\right), \\ \widetilde{T}\,(t_0 = 0\ \text{s}) = 555\ \text{K}. \end{cases} \tag{25}$$

2.9 Numerical solution

Equation 25 was solved numerically using RK2 with a time step of $h_1 = 720\ s$ and with RK4 with $h_2 = 1440\ s$. Both the solutions were compared with the one obtained MATLAB's `ode89`, which, being an higher order integrator, is expected to better capture the real system dynamics.

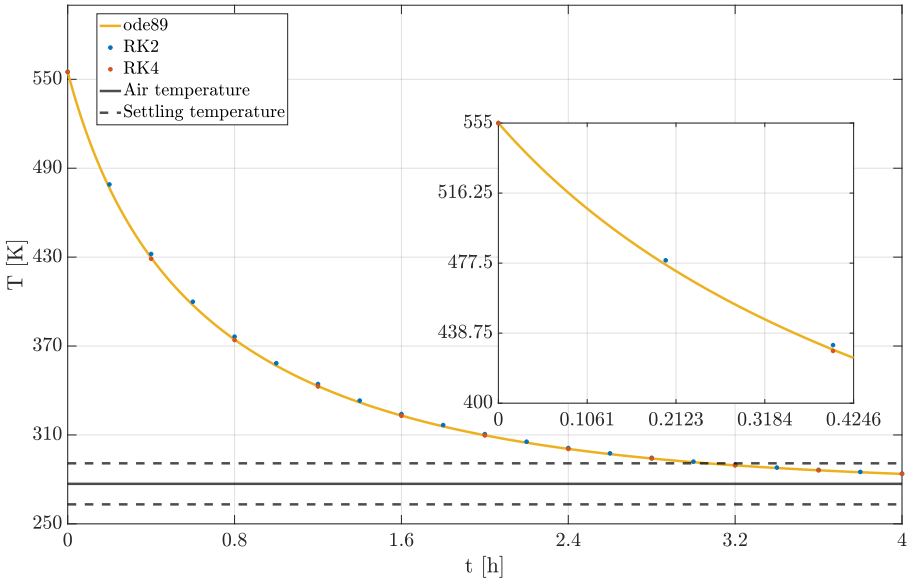


Figure 23: Numerical solution obtained with RK2, RK4 and `ode89`

As shown in Figure 23, the temperature of the mass decreases over time, eventually converging to the equilibrium temperature, which is equal to the temperature of the surrounding air. The settling time, defined as the time required for the temperature to remain within a ±5% range around the equilibrium, is approximately 3.2 h.

Figure 24 illustrates the time evolution of the relative error with respect to the `ode89` solution. It can be observed that the RK4 error is consistently lower than that of RK2. This behavior is expected, as RK4 is a higher-order method and generally provides greater accuracy than RK2. Additionally, it can be noted that the error initially increases, reaches a maximum, and then begins to decrease over time. This behavior is attributed to the temperature approaching steady-state conditions. Since the truncation error is proportional to the higher-order derivatives [1], which tend to zero near the steady state, the discrepancy between `ode89`, RK2, and RK4 diminishes as time progresses.

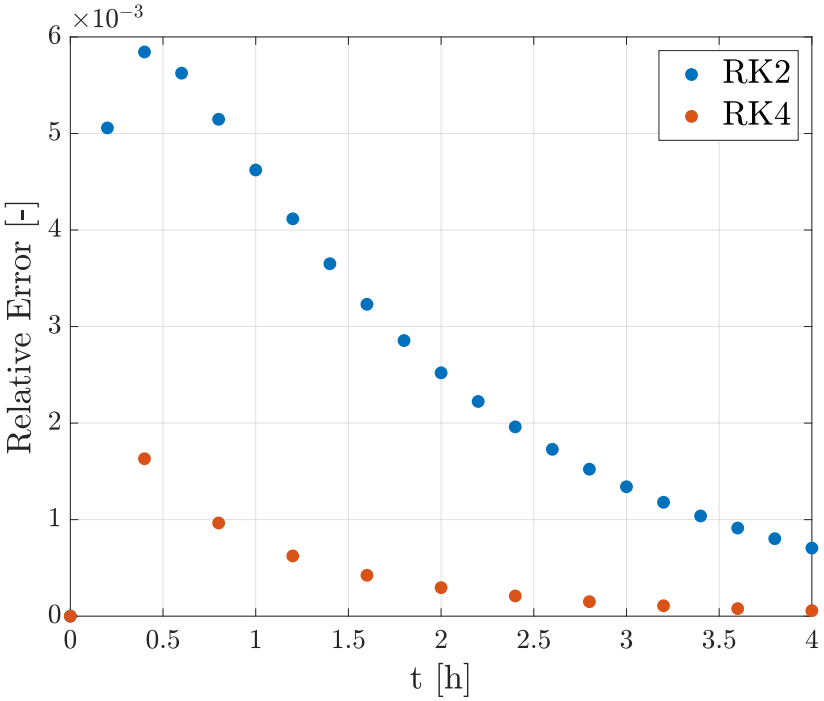


Figure 24: Time history of the relative error with RK2 and RK4

In Table 4, the results are summarized in terms of the maximum discrepancy with respect to the `ode89` solution. It can be observed that RK4 with a time step of $h = 1440\ s$ is the best choice between the two analyzed methods (RK4 with $h = 1440\ s$ and RK2 with $h = 720\ s$), as it achieves a lower error with the same number of function evaluations. Additionally, it is noted that an error two orders of magnitude lower could be obtained by integrating Equation 25 using RK4 with a time step of $h = 720\ s$, at the cost of doubling the number of function evaluations. A good compromise between accuracy and computational effort, which could be implemented in future studies, is an adaptive time-step strategy. In this approach, the time step would be kept smaller in regions where larger gradients are present and gradually increased as the system approaches steady-state conditions.

Table 4: Maximum relative error and function evaluations for RK2 and RK4

	RK2 (h=720 s)	**RK2 (h=1440 s)**	**RK4 (h=720 s)**	**RK4 (h=1440 s)**
Relative Error	$5.8443 \cdot 10^{-3}$	$1.7055 \cdot 10^{-2}$	$4.0744 \cdot 10^{-5}$	$1.6307 \cdot 10^{-3}$
Function Evaluations	40	20	80	40

Exercise 5

Consider the electrical circuit in Fig. 25. 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.

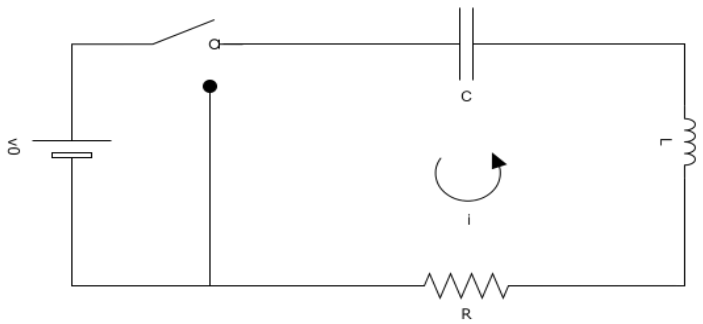


Figure 25: Electric circuit model.

(4 points)

2.10 Point 1

The linear ODE governing the system's dynamics in state-space form can be derived by analyzing the system at $t > 0$. In this condition, the switch is open, meaning the battery is disconnected and does not influence the system's dynamics. The system equation can be obtained by applying the Kirchhoff's Voltage Law (KVL) to the external loop:

$$v_C + v_L + v_R = 0 \tag{26}$$

Substituting the constitutive relations of the capacitor, inductor, and resistor, and considering that the current in the circuit is given by $i = \frac{dq}{dt}$, a second-order ODE governing the system's dynamics is obtained:

$$\frac{q\,(t)}{C} + L\,\frac{d^2q\,(t)}{dt^2} + R\,\frac{dq\,(t)}{dt} = 0 \tag{27}$$

The state-space representation of the system can be derived by dividing Equation 27 by L and introducing the state variables $x_1 = q$ and $x_2 = \frac{dq}{dt}$.
The initial conditions can be determined by enforcing continuity at $t = 0^-$. At $t = 0^-$, the system is in steady-state conditions, meaning that the capacitor behaves as an open circuit. Consequently, the current is $i(0^-) = 0$ A, and the capacitor voltage counterbalances the voltage imposed by the battery.
Thus, the Cauchy problem can be formulated as:

$$\begin{cases} \dfrac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\dfrac{1}{LC} & -\dfrac{R}{L} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \\ \\ \begin{bmatrix} x_1\,(0) \\ x_2\,(0) \end{bmatrix} = \begin{bmatrix} v_0 C \\ 0 \end{bmatrix} \end{cases} \tag{28}$$

2.11 Point 2

To evaluate whether the system is stiff, the eigenvalues of the matrix $\mathbf{A}$, given in Equation 28, which governs the system dynamics, were analyzed. Solving the characteristic equation of $\mathbf{A}$, the eigenvalues can be computed analytically, yielding an expression that depends on the three parameters R, L, and C:

$$\lambda_{1,2} = -\frac{R}{2L} \pm \frac{\sqrt{\dfrac{CR^2 - 4L}{C}}}{2L} \tag{29}$$

Substituting $R = 25\ \Omega$, $L = 20 \cdot 10^{-3}\ H$, and $C = 200 \cdot 10^{-3}\ F$ into Equation 29, the eigenvalues are obtained as $\lambda_1 \approx -0.2$ and $\lambda_2 \approx -1249.8$.
Both of these values are negative and lie on the real axis, indicating that the system is asymptotically stable. As a result, the continuous-time solution is expected to decay to zero after a transient phase, without oscillations. Moreover, since λ_1 and λ_2 differ by more than three orders of magnitude, one of the system's eigenmodes evolves significantly faster than the other. This confirms that the system is stiff [4].
The stability regions of RK2 and Implicit Extrapolation Method of order 4 (IEX4) were computed using the same technique of subsubsection 2.6.1, thus the linear operator $\mathbf{F}$ such that $\mathbf{x}_{k+1} = \mathbf{F}\mathbf{x}_k$ needs to be retrieved both for RK2 and IEX4. The stability of the numerical solution is ensured if all the eigenvalues of F have absolute value smaller than one. $\mathbf{F}_{RK2}$, corresponding to RK2, was derived using the same procedure already discussed in subsection 2.6,

leading to the expression in Equation 30, which can be noted that matches the exponential matrix up to the second order, as RK2 is a second order method as well [3].

$$\mathbf{F}_{RK2}(h,\alpha) = \mathbf{I}_{2\times 2} + h\,\mathbf{A}(\alpha) + \frac{h^2}{2}\,\mathbf{A}^2(\alpha)\,. \tag{30}$$

IEX4 is a fourth-order accurate method in which the solution $\mathbf{x}_{k+1}$ is computed as a linear combination of four predictors obtained using Backward Euler (BE), an implicit first-order accurate method [3]. In these predictors, the *macrostep* of length h is subdivided into intermediate steps of lengths $\eta_1 = h$, $\eta_2 = h/2$, $\eta_3 = h/3$, and $\eta_4 = h/4$. The four predictors and the corrector for the linear system case are given in Equation 31:

$$\begin{aligned}
\mathbf{x}_{k+1}^{P_1} &= (\mathbf{I} + h\,\mathbf{A})^{-1}\,\mathbf{x}_k \\
\mathbf{x}_{k+1}^{P_2} &= \left(\mathbf{I} + \frac{h}{2}\mathbf{A}\right)^{-2}\,\mathbf{x}_k \\
\mathbf{x}_{k+1}^{P_3} &= \left(\mathbf{I} + \frac{h}{3}\mathbf{A}\right)^{-3}\,\mathbf{x}_k \\
\mathbf{x}_{k+1}^{P_4} &= \left(\mathbf{I} + \frac{h}{4}\mathbf{A}\right)^{-4}\,\mathbf{x}_k \\
\mathbf{x}_{k+1}^{C} &= \alpha_1\mathbf{x}_{k+1}^{P_1} + \alpha_2\mathbf{x}_{k+1}^{P_2} + \alpha_3\mathbf{x}_{k+1}^{P_3} + \alpha_4\mathbf{x}_{k+1}^{P_4}
\end{aligned} \tag{31}$$

The weights of the linear combination are obtained by matching the corrector expression in Equation 31 to the Taylor series expansion up to fourth order [3]. Solving the resulting linear system gives the values of the α_i coefficients:

$$\alpha_1 = -\frac{1}{6}, \quad \alpha_2 = 4, \quad \alpha_3 = -\frac{27}{2}, \quad \alpha_4 = \frac{32}{3}.$$

By substituting these computed weights into the corrector expression, the linear operator associated with IEX4 is obtained:

$$\mathbf{F}_{IEX4} = -\frac{1}{6}(\mathbf{I} - h\mathbf{A})^{-1} + 4\left(\mathbf{I} - \frac{h}{2}\mathbf{A}\right)^{-2} - \frac{27}{2}\left(\mathbf{I} - \frac{h}{3}\mathbf{A}\right)^{-3} + \frac{32}{3}\left(\mathbf{I} - \frac{h}{4}\mathbf{A}\right)^{-4}. \tag{32}$$

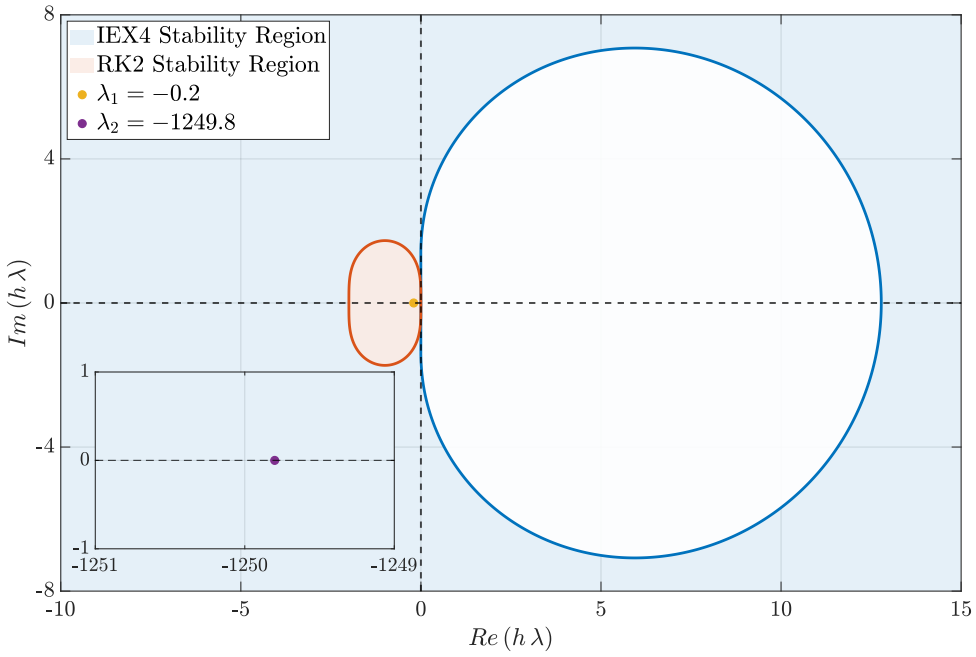


Figure 26: RK2 and IEX4 stsbility domains

$$\mathbf{A}(\alpha) = \begin{bmatrix} 0 & 1 \\ -1 & 2\cos\alpha \end{bmatrix} \tag{33}$$

The stability regions shown in Figure 26 were obtained following the same procedure described in subsubsection 2.6.1. Specifically, the matrix $\mathbf{A}(\alpha)$ in Equation 33 was substituted into Equation 30 and Equation 32. For each value of $\alpha \in [0, \pi]$, the maximum allowable time step h ensuring numerical stability was determined by solving:

$$\max\left(|\text{eig}(\mathbf{F})|\right) - 1 = 0 \tag{34}$$

for both RK2 and IEX4. Once $h_{\max}(\alpha)$ was computed for $\alpha \in [0, \pi]$ for both methods, the product of $h_{\max}(\alpha)$ and the eigenvalues of $\mathbf{A}(\alpha)$, given by

$$\lambda_{1,2}(\alpha) = e^{\pm \mathrm{j}\alpha}, \tag{35}$$

was plotted in the $h\lambda$ complex plane. This resulted in the contours of the stability regions for RK2 and IEX4, depitched in Figure 26, together with the eigenvalues of the linear system under analysis, mapped with a discretization step of $h = 1\ s$.

It can be observed that λ_2 falls outside the stability region of RK2, implying that an excessively small time step would be required to integrate the system while guaranteeing numerical stability of the solution, resulting in excessive computational effort. It can also be observed that, since IEX4 is an implicit method, its stability region extends outside the previously computed stability curve. As a consequence, the entire left half-plane is part of the IEX4 stability region, ensuring the stability of the analyzed system for any choice of h. This allows for the selection of larger time steps for system integration, thereby reducing the computational cost.

2.12 Point 3

The LTI system under analysis was simulated using IEX4 because, as discussed in subsection 2.11, both the eigenvalues of the system fall inside its stability region independently from the time step h, allowing for a reduction of the computational cost. Exploiting the linearity and the time invariance of the system under analysis, it is possible to recover $\mathbf{x}_{k+1}$ of the problem simply pre-multiplying $\mathbf{x}_k$ fer the linear operator characterizing IEX4:

$$\begin{cases} \mathbf{x}_{k+1} = \mathbf{F}_{IEX4}\,\mathbf{x}_k \\ \mathbf{x}(0) = \mathbf{x}_0 = \begin{bmatrix} v_0\,C & 0 \end{bmatrix}^T \end{cases} \tag{36}$$

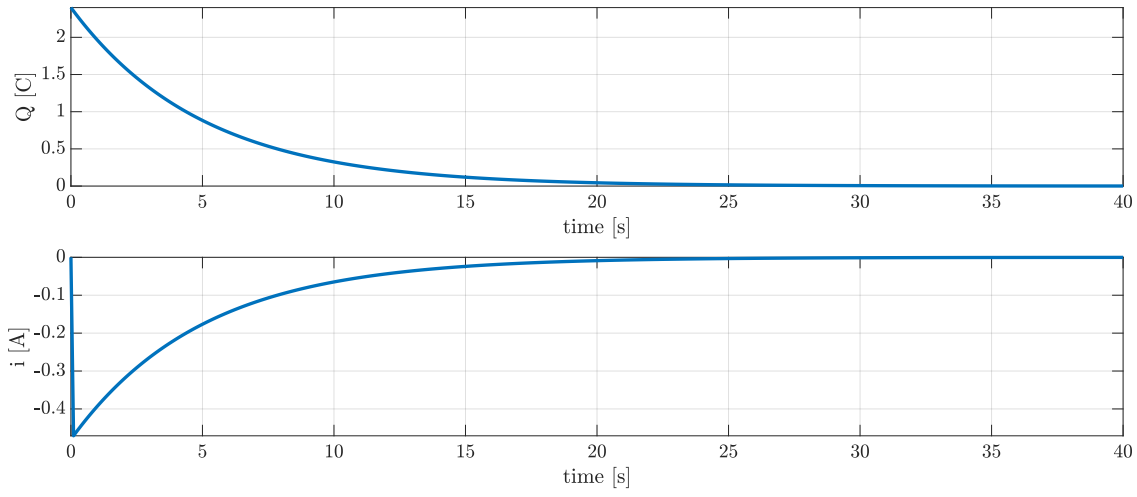


Figure 27: Transient response of the system, computed with IEX4

This implementation choice reduces computational effort because the matrix $\mathbf{F}_{IEX4}$ remains constant over time. As a result, it only needs to be computed once at the beginning of the integration process. Consequently, each time step of the IEX4 integration requires only a matrix-vector product, rather than solving ten nonlinear equations per time instant as in the general-purpose IEX4 formulation [2].

The system's transient response, computed using IEX4 with a time step of $h = 0.1$ s, is illustrated in Figure 27. As expected, the charge on the capacitor gradually decreases over time, converging to zero. When the switch is opened, the current exhibits an initial spike before decreasing, following the same trend as the capacitor charge. This behavior is anticipated, as the current is proportional to the time derivative of the capacitor charge.

The maximum allowable time step ensuring the stability of the RK2 solution can be determined by noting that the stability region of RK2 intersects the real axis of the $h\lambda$ complex plane at -2. Numerical stability is guaranteed if the product of h and λ_2, the eigenvalue of $\mathbf{A}$ with the largest absolute value, remains within the stability region. This yields:

$$h_{max} = \frac{-2}{\lambda_2} \approx 0.0016 \text{ s}, \quad \text{where} \quad \lambda_2 \approx -1249.8.$$

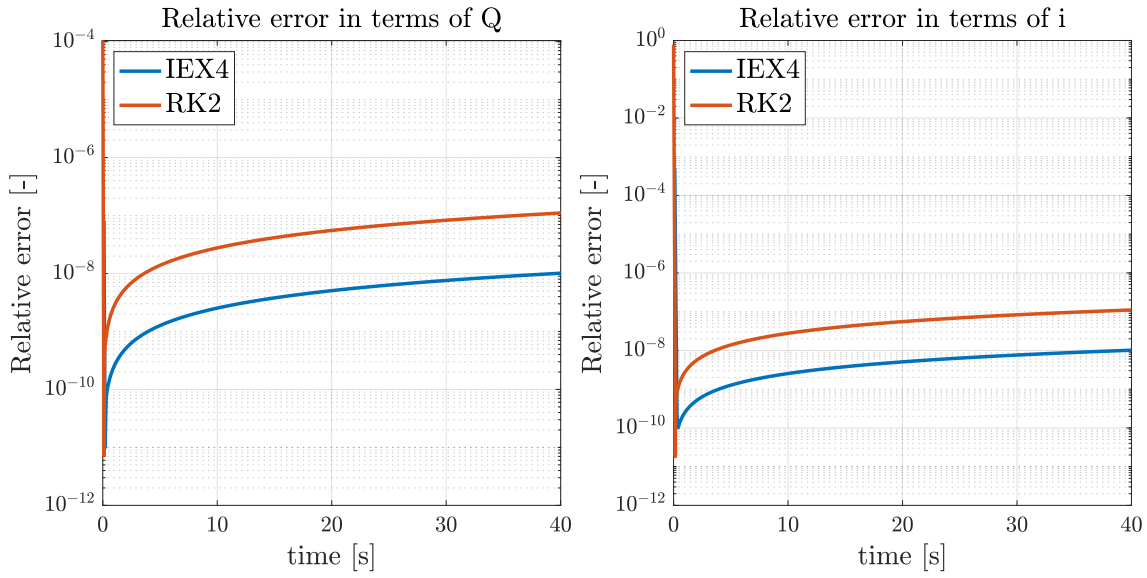


Figure 28: Relative error both in the capacitor charge and in the current

The system response was also simulated using a RK2 method specifically for LTI systems. In this approach, $\mathbf{F}_{RK2}$ is computed once at the start of the integration, and the solution is propagated using the recurrence relation $\mathbf{x}_{k+1} = \mathbf{F}_{RK2}\,\mathbf{x}_k$, with a time step of $h = 0.9\,h_{max}$. The exact solution, given by $e^{\mathbf{A}t}\,\mathbf{x}_0$ for LTI systems, was computed using the MATLAB function `expmv`, which efficiently evaluates the matrix exponential. This allowed for the assessment of the relative error between the numerical solutions obtained by RK2 and IEX4 with respect to the exact solution.

Figure 28 illustrates that, following an initial peak, caused by a rapid transient in the current, the relative error stabilizes below 10^{-6} for the entire simulation duration in both methods. However, the error associated with IEX4 consistently remains lower than that of RK2. Given that IEX4 achieves this accuracy with a larger time step, thereby reducing computational cost, it proves to be a more efficient choice for integrating the stiff system under study.

Exercise 6

Consider the bouncing ball physical model in Fig. 29. The mathematical model of the system is represented by the following equations:

$$\frac{dv}{dt} = -g + \frac{F_D}{m_b} + \frac{F_A}{m_b} \tag{37}$$

$$\frac{dx}{dt} = v \tag{38}$$

$$v^+ = -k \cdot v^- \tag{39}$$

$$F_D = -\ 0.5 \cdot \rho \cdot C_d \cdot A_b \cdot v \cdot |v| \tag{40}$$

$$F_A = \rho \cdot V_b \cdot g \tag{41}$$

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.

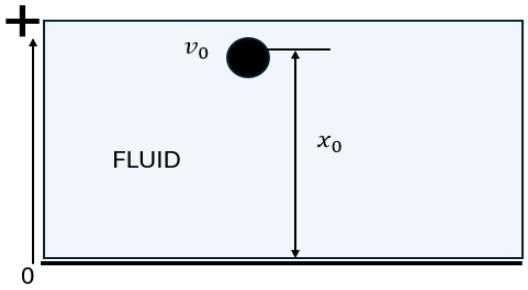


Figure 29: Bouncing ball system.

(5 points)

2.13 Point 1

The mathematical model under analysis consists of a set of nonlinear ODEs describing the dynamics of a bouncing ball subjected to gravity, fluid dynamic drag, and buoyancy force. The state-space representation of the system is given in Equation 42, along with the corresponding initial conditions:

$$\begin{cases} \dfrac{d}{dt}\begin{bmatrix} v \\ x \end{bmatrix} = \begin{bmatrix} -g - \dfrac{\rho\, C_D\, A_b\, v\, |v|}{2\, m_b} + \dfrac{\rho\, V_b\, g}{m_b} \\ v \end{bmatrix} \\ \begin{bmatrix} v\,(0) \\ x\,(0) \end{bmatrix} = \begin{bmatrix} 0 \\ 10 \end{bmatrix} \end{cases} \tag{42}$$

To accurately model the bounces on the ground, which introduce a discontinuity in velocity, an event function was added to the ODE solver options. This function stops the integration process at each ground impact. After each impact, the integration restarts with new initial conditions, as defined in Equation 43, where t_{imp}^{-} and t_{imp}^{+} refer to the time instants immediately before and after the impact, respectively:

$$\begin{bmatrix} v\left(t_{imp}^{+}\right) \\ x\left(t_{imp}^{+}\right) \end{bmatrix} = \begin{bmatrix} -k\, v\left(t_{imp}^{-}\right) \\ 0 \end{bmatrix} \tag{43}$$

The simulations were conducted using MATLAB's built-in integrator `ode45`, with both `RelTol` and `AbsTol` set to 10^{-12}. Figure 30 displays the obtained results for a fluid density of $\rho = 1.225$ kg/m^3. In this scenario, each bounce results in a loss of kinetic energy due to the impact with the ground, leading to a gradual decrease in the peak height reached by the ball. The drag force F_D, which is proportional to the square of the velocity, opposes the motion of the ball, causing it to decelerate faster as it falls and reducing its speed after each bounce.

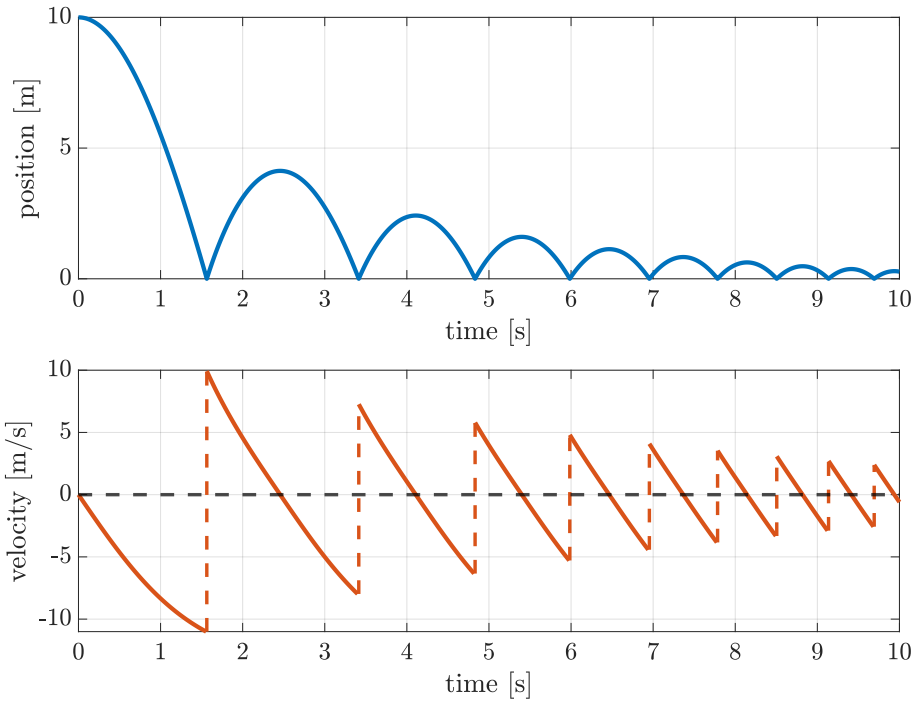


Figure 30: Position and velocity for $\rho = 1.225$ kg/m^3

2.14 Point 2

In Figure 31, the simulation results are illustrated for a case where the fluid density is increased to $\rho = 15$ kg/m^3. As expected, the fluid dynamic drag is significantly higher than in the $\rho = 1.225$ kg/m^3 case, since the drag force F_D is proportional to the fluid density. As a result, the ball takes longer to reach the ground.

Furthermore, as observed in Figure 31, before the first bounce, the fluid dynamic drag, proportional to the square of the velocity, balances the combined effect of buoyancy and gravity, causing the ball to reach the ground with a constant velocity. After the first bounce, the strong air resistance considerably dampens the vertical motion, leading to a faster cessation of the bouncing process compared to the $\rho = 1.225$ kg/m^3 case.

From a computational perspective, a condition was implemented to terminate the integration process when the peak height between two successive impacts fell below a predefined threshold of 10^{-3} m. This prevented unnecessary computations once the ball had effectively come to rest.

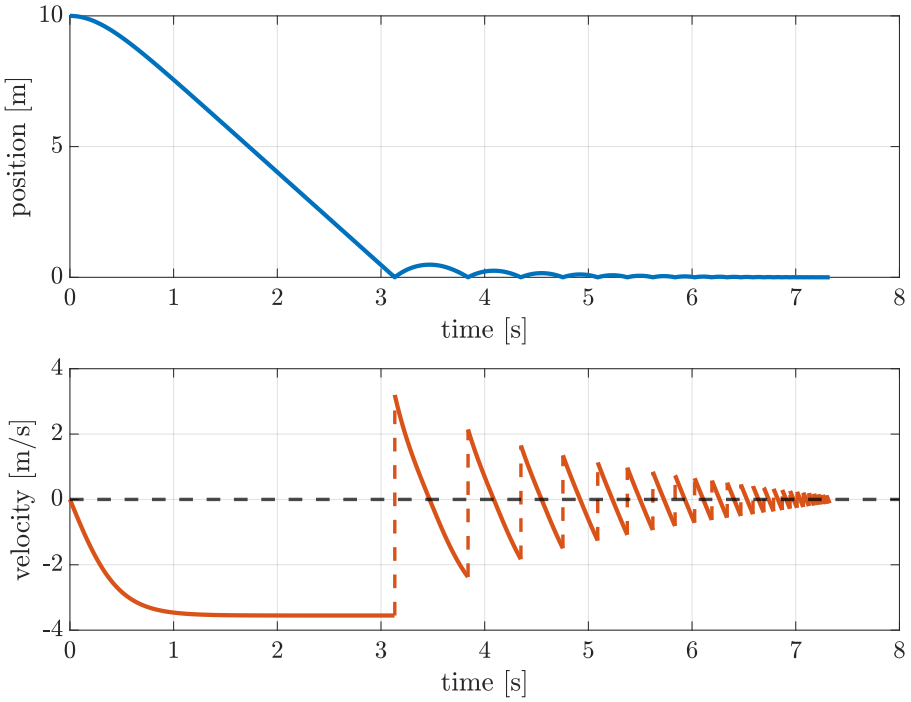


Figure 31: Position and velocity for $\rho = 15$ kg/m^3

2.15 Point 3

With a further increase in fluid density to $\rho = 60$ kg/m^3, the drag force becomes so dominant that the ball does not reach the ground within the analyzed time span of $\Delta t = 10$ s, as shown in Figure 32. It can be observed that the terminal velocity is lower than in the $\rho = 15$ kg/m^3 case. This effect arises because the buoyancy force, which is proportional to the fluid density, plays a significant role in the system dynamics when $\rho = 60$ kg/m^3. Since buoyancy is constant and always directed upward, it reduces the net downward force, which is ultimately balanced by the fluid dynamic drag. As a result, the steady-state velocity is lower than in the $\rho = 15$ kg/m^3 case.

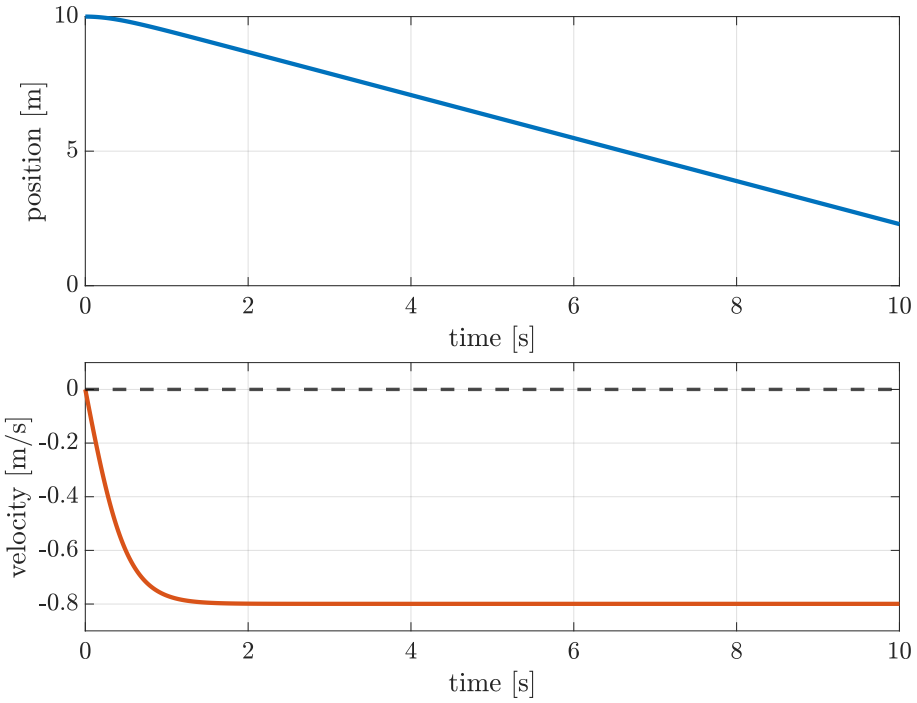


Figure 32: Position and velocity for $\rho = 60$ kg/m^3

2.16 Point 4

Finally, the system was integrated using RK4 with a fixed step size of $h = 3$ s. However, as shown in Figure 33, the solver fails to produce reasonable results. This instability arises because such a large step size maps the eigenvalues of the Jacobian of the system's RHS outside the stability region of RK4, leading to a divergent behaviour.

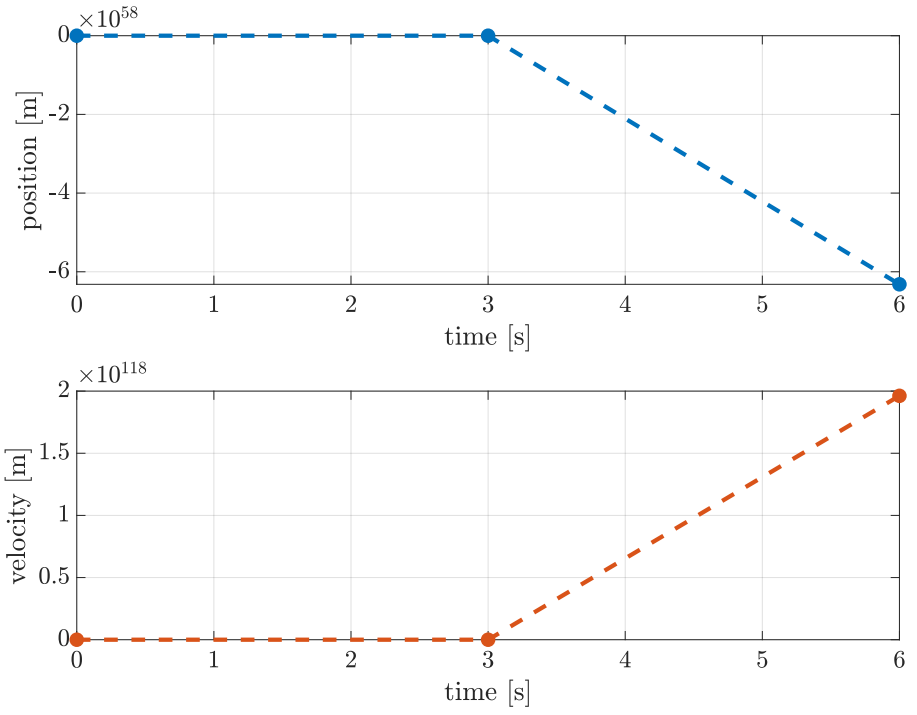


Figure 33: Position and velocity for $\rho = 60$ kg/m^3 obtained using RK4 with $h = 3$ s

To obtain a stable solution with RK4, it is necessary to reduce the step size h so that the eigenvalues of the system remain within the stability region of RK4 throughout the entire simulation. Figure 34 presents the results obtained with a reduced step size of $h = 0.5$ s. While minor accuracy issues remain, the stability is restored, and the solution closely follows the reference trajectory computed using `ode45`, illustrated in Figure 32.

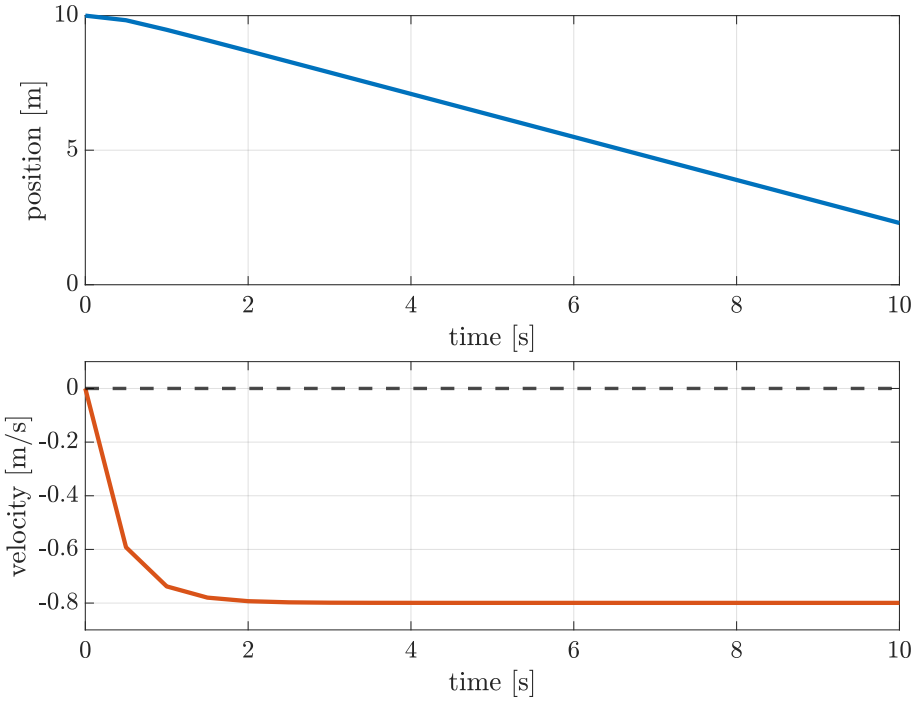


Figure 34: Position and velocity for $\rho = 60$ kg/m^3 obtained using RK4 with $h = 0.5$ s

References

[1] Gervasio P. Quarteroni A. Saleri F. *Scientific Computing with MATLAB and Octave*. 4th. Springer, 2014. ISBN: 978-3-642-45366-3. DOI: 10.1007/978-3-642-45367-0.

[2] William H. Press et al. *Numerical Recipes: The Art of Scientific Computing*. 3rd. Cambridge University Press, 2007. ISBN: 978-0-521-88068-8. DOI: 10.1017/CBO9780511809261.

[3] François E. Cellier and Ernesto Kofman. *Continuous System Simulation*. Springer, 2006. ISBN: 978-0-387-26103-0. DOI: 10.1007/0-387-30260-3.

[4] Karl Johan Åström and Richard M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2008. ISBN: 978-0-691-13576-2. DOI: 10.1515/9781400828739.