MSAS – Assignment #1: Simulation

Davide Zamblera, 994071

1 Implicit equations

Exercise 1

Given the function $f(x) = \sin x + x - 1$, guess a and b such that $f(a)f(b) < 0$. Find the zero(s) of f in $[a, b]$ with 1) the bisection method, 2) the secant method, 3) the regula falsi method. All solutions must be calculated with 8-digit accuracy. Which method requires the least number of function evaluations? Report the computational time required by each method.

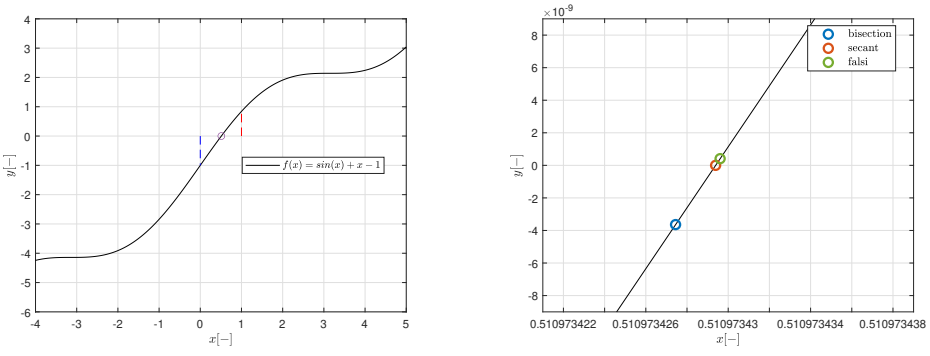
A plot of the function is visible in Fig. 1a and the range [0,1] is identified as feasible for the search since it is characterized by a sign change. There cannot be other zeros since the function can be seen as a superposition of a monotonically increasing $x - 1$ function and a marginally stable oscillating $\sin x$ function.
The three solution methods are implemented as Matlab functions and the results are presented in Table 1, while the accuracy of the methods is visible in Fig. 1b
The request to achieve a 8 digit accuracy was interpreted as a condition on the relative error with tol 10^{-8} in Eq. (1), this solution is considered appropriate because it targets directly the number of significant digits and the solution is not near zero, so a relative error measure does not generate numerical problems.

$$\frac{|x_{k+1} - x_k|}{|x_{k+1}|} < tol \tag{1}$$

The functions also return a variable k which contains the number of function evaluations, the computational time is obtained via the built-in Matlab function timeit.

Table 1: Table of Results.

Method	Result	Time [s]	Evaluations
Bisection method:	0.510973427445	0.000001363493	30
Secant method:	0.510973429389	0.000000273903	7
Falsi method:	0.510973429608	0.000000332574	9



(a) Identify initial guesses **(b)** Comparison between solutions

Figure 1

Bisection is by far the slowest, while Secant and Falsi method are very similar. It's important to note though that since the Secant method is not a two-point bracketing method, meaning it lacks a constraint on the area of search in a certain range, in other situations it could diverge, while Falsi method cannot.

Exercise 2

Let $\mathbf{f}$ be a two-dimensional vector-valued function $\mathbf{f}(\mathbf{x}) = (x_1^2 + x_2 - 5, x_2^2 - x_1)^\top$, where $\mathbf{x} = (x_1, x_2)^\top$. Find the zero(s) of $\mathbf{f}$ by using Newton's method with $\partial\mathbf{f}/\partial\mathbf{x}$ 1) computed analytically, and 2) estimated through finite differences. Which version is more accurate?

The function is regarded as a system of 2 two-dimensional function. Then a contour plot at height 0 of both is used in the search of the zeros, two points can be seen from Fig. 2. There is no need to search far away from this region for other zeros since these functions have parabolic contour lines with a maximum of two intersections. From this plot two guess solutions are proposed: $x_{guess1} = [2, 1.5]$ and $x_{guess2} = [2, -2]$

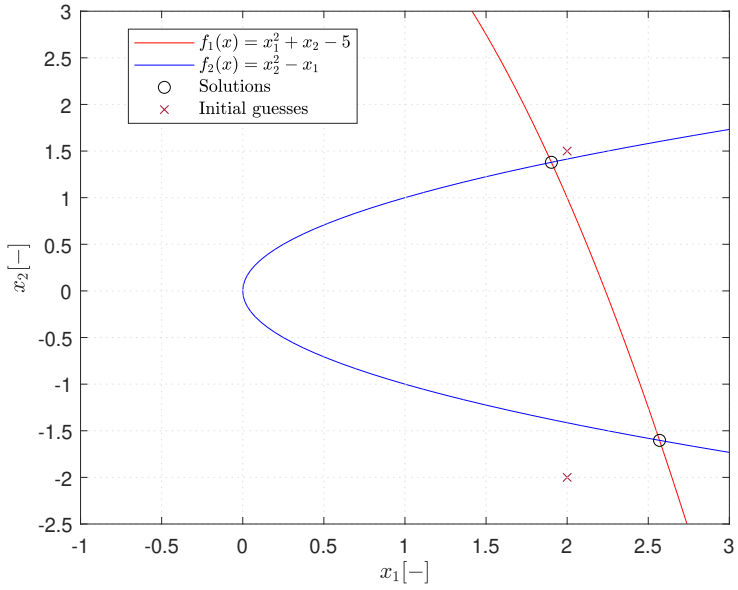


Figure 2: Contour lines at height equal to zero

Newton's method is a root search method which uses first derivatives as a search direction, in the case of multiple equations and multiple variables this means that the computation of the Jacobian at each step is needed. The true Jacobian is derived analytically in Eq. (2).

$$\frac{\partial f_i}{\partial x_j} = \begin{pmatrix} 2\,x_1 & 1 \\ -1 & 2\,x_2 \end{pmatrix} \tag{2}$$

While for the finite differences approximation a forward finite difference is considered because of the simplicity of the problem and the desire of keeping the computation load as low as possible. The results are displayed in Table 2:

Table 2: Newton search results.

Point	x1 [-]	x2 [-]
First Point:	1.902783683509	1.379414253772
Second Point:	2.569631701411	-1.603007080899

In this case numerical and analytical results closely match. This can be explained by the simplicity of the problem, the Jacobian is linear, and by the small step chosen for the forward difference, $\sqrt{eps}$, with *eps* machine number.

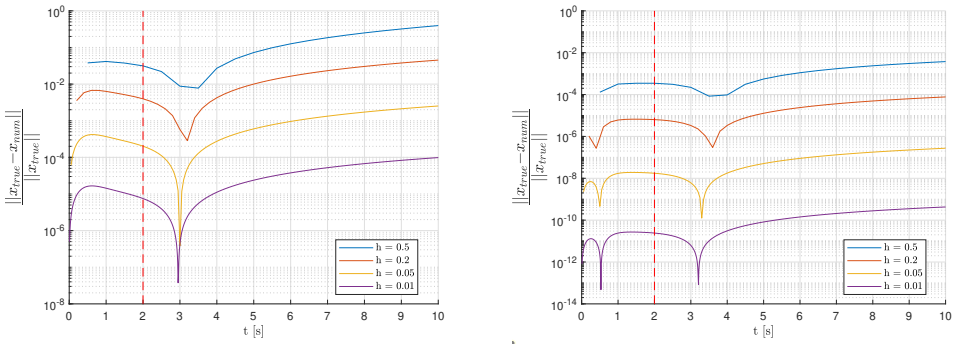
2 Numerical solution of ODE

Exercise 3

The Initial Value Problem $\dot{x} = -\frac{t^2-1}{t^2+1}x$, $x(0) = 1$, has analytic solution $x(t) = e^{2\arctan(t)-t}$. 1) Implement a general-purpose, fixed-step Heun's method (RK2); 2) Solve the IVP in $t \in [0, 2]$ for $h_1 = 0.5$, $h_2 = 0.2$, $h_3 = 0.05$, $h_4 = 0.01$ and compare the numerical vs the analytical solution; 3) Repeat points 1)–2) with RK4; 4) Trade off between CPU time & integration error.

Heun's method and in general the Runge Kutta family can be understood as an extension of the predictor corrector method.
The relative error of the numerical solution with respect to the exact one was chosen as an indicator of the accuracy of the methods.

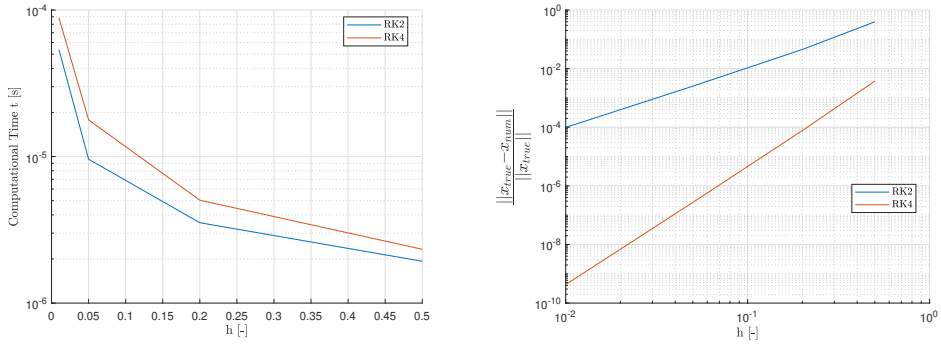
Their performance was initially evaluated separately for multiple h, and against integration time, the results are Fig. 3a and Fig. 3b. The sudden drop and localized error decrease with time must be understood as results of numerical noise. By increasing the time of integration the function stabilizes and both methods suffer from the accumulation error and their accuracy decreases.



(a) Heun method **(b)** RK4 method

Figure 3: Accuracy vs Integration Time.

Confronting their behaviour in accuracy and computational time shows that the slight gain of RK2 in speed of computation visible in Fig. 4a is not enough to balance its poor accuracy performance depicted in Fig. 4b.



(a) Computational Time **(b)** Accuracy of the methods

Figure 4: Method comparison.

Exercise 4

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 Re$\{\lambda\}$-axis. 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 and represent the points $\{h_i\lambda\}$ of Exercise 3. What can you say?

For simple systems as linear ones it is possible to write an operator matrix that update the state at each iteration, for RK2 and RK4 the operators are:

$$F_{RK2}(\alpha, h) = \mathrm{I} + A(\alpha)\, h + \frac{h^2}{2} A(\alpha)^2 \tag{3}$$

$$F_{RK4}(\alpha, h) = \mathrm{I} + A(\alpha)\, h + \frac{h^2}{2} A(\alpha)^2 + \frac{h^3}{6} A(\alpha)^3 + \frac{h^4}{24} A(\alpha)^4 \tag{4}$$

The problem of finding $h \geq 0$ s.t.max $(|\mathrm{eig}(F(h, \alpha))|) = 1$ is equivalent in finding h such that the eigenvalues of the discrete time numerical problem lie in the circle of radius 1 and center origin. Once α is fixed the problem can be rewritten as a zero finding problem with $f(h) = \max(|\mathrm{eig}(F(h, \alpha))|) - 1$, and solved with the appropriate methods, in this implementation the fzero Matlab function was chosen. The results are displayed in Table 3 This functionality

Table 3: Marginal h for $\alpha = \pi$

Method	Value [-]
RK2	1.99999999
RK4	2.78529356

can then be used in a iterative manner to find h_{max} for every α, exploiting the fact that eigenvalues have complex conjugate a scan of only $[0, \pi]$ is actually needed. The results are the stability regions in Fig. 5. Increasing the order of the method increases the region of stability, unfortunately for RK4 it makes numerically stable region which are not stable analytically. By plotting the eigenvalue of Exercise 3 for various h_i and for t = 0 it is clear that they are all outside the region of stability. Nevertheless the eigenvalues at a specific time determine the stability only locally, in this case the global behaviour of the system is to converge asymptotically.

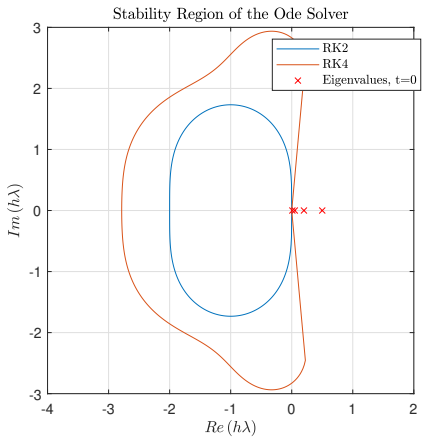
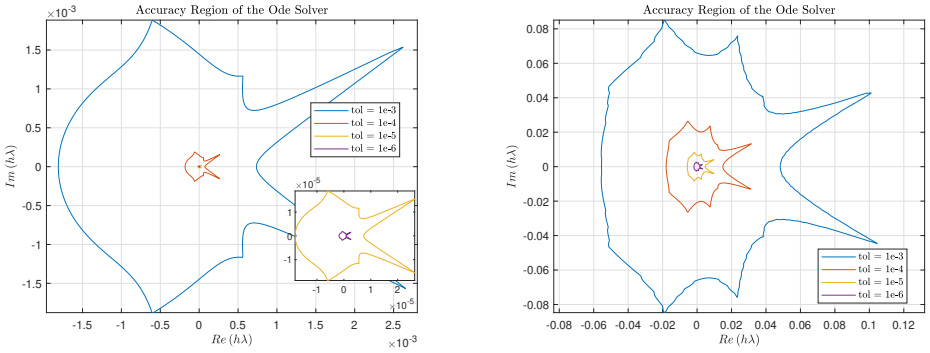


Figure 5: Region of stability and eigenvalues

Exercise 5

Consider the IVP $\dot{\mathbf{x}} = A(\alpha)\mathbf{x}$, $\mathbf{x}(0) = [1,1]^T$, to be integrated in $t \in [0,1]$. 1) Take $\alpha \in [0, 2\pi]$ and solve the problem "Find $h \geq 0$ s.t. $\|\mathbf{x}_{\text{an}}(1) - \mathbf{x}_{\text{RK1}}(1)\|_\infty = \text{tol}$", where $\mathbf{x}_{\text{an}}(1)$ and $\mathbf{x}_{\text{RK1}}(1)$ are the analytical and the numerical solution (with RK1) at the final time, respectively, and $\text{tol} = \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$. 2) Plot the five locus of solutions in the $(h\lambda)$-plane; plot also the function evaluations vs tol for $\alpha = \pi$. 3) Repeat points 1)–2) for RK2 and RK4.

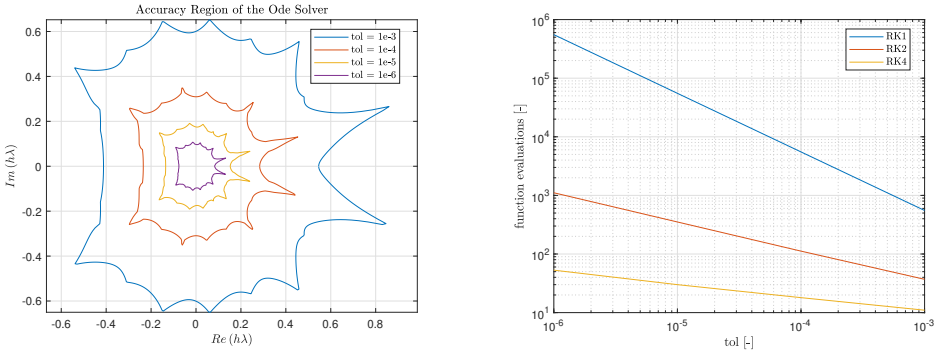
This problem requires plotting accuracy regions on the $h\lambda$ plane, so regions in which it's assured a certain level of accuracy measured by a tolerance on the final value in infinite norm. The implementation deals with h values which fail to match exactly on t_f by performing a last step of appropriate length. The fzero Matlab function is chosen as the nonlinear equation solver and the range of h in which to search is chosen by trial and error for all tolerances and methods. The results for RK1, RK2 and RK4 are visible respectively in Fig. 6a, Fig. 6b, and Fig. 7a.



(a) Regions of Accuracy RK1 **(b)** Regions of Accuracy RK2

Figure 6

It is clear how an increase of the method order, leads to an increase of the regions of accuracy, permitting a use of greater h to achieve a certain level of accuracy. Fig. 7b shows that the increase of function evaluations due to higher orders is a minimal factor with respect to the increase due to smaller steps typical of lower order methods. This difference is increased for stricter tolerances and RK1, RK2 rapidly loose computational efficiency with respect to RK4 for $tol = 10^{-6}$.



(a) Regions of Accuracy RK4 **(b)** Function evaluation vs Tolerance

Figure 7

Exercise 6

Consider the backinterpolation method $\mathrm{BI2}_{0.4}$. 1) Derive the expression of the linear operator $B_{\mathrm{BI2}_{0.4}}(h,\alpha)$ such that $\mathbf{x}_{k+1} = B_{\mathrm{BI2}_{0.4}}(h,\alpha)\mathbf{x}_k$. 2) Following the approach of point 3) in Exercise 5, draw the stability domain of $\mathrm{BI2}_{0.4}$ in the $(h\lambda)$-plane. 3) Derive the domain of numerical stability of $\mathrm{BI2}_\theta$ for the values of $\theta = [0.1, 0.3, 0.7, 0.9]$.

Assuming that the RK2 update operator is known, the derivation of the BI2 operator is straightforward. The derivation can be divided in a forward step, from x_k to $x_{k+\theta}$ of length $h\,\theta$, shown in Eq. (5), and a backward step of length $h\,(1-\theta)$ from x_{k+1} to $x_{k+\theta}$, shown in Eq. (6). These are combined and the result is Eq. (7).

$$x_{k+\theta} = \left(\mathrm{I} + \theta\, h\, A\,(\alpha) + \frac{(\theta\, h)^2}{2} A\,(\alpha)^2\right) x_k \tag{5}$$

$$x_{k+1} = \left(\mathrm{I} - (1-\theta)\, h\, A\,(\alpha) + \frac{((1-\theta)\, h)^2}{2} A\,(\alpha)^2\right)^{-1} x_{k+\theta} \tag{6}$$

$$B_{BI2_\theta} = \left(\mathrm{I} - (1-\theta)\, h\, A\,(\alpha) + \frac{((1-\theta)\, h)^2}{2} A\,(\alpha)^2\right)^{-1} \left(\mathrm{I} + \theta\, h\, A\,(\alpha) + \frac{(\theta\, h)^2}{2} A\,(\alpha)^2\right) \tag{7}$$

The region of stability of the method for various θ is visible in Fig. 8.

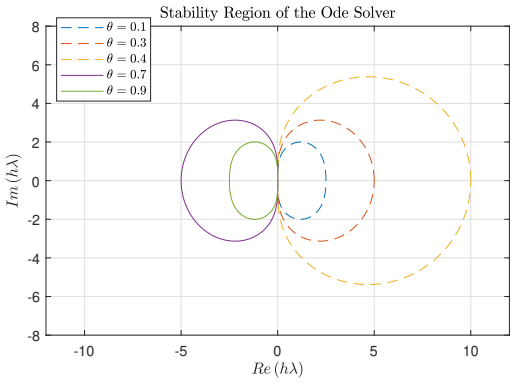


Figure 8: Regions of stability for the BI method *

This shows that the region of stability of the BI method can be easily manipulated via the θ parameter. In particular two main families are visible:

1. When $\theta > 0.5$, the forward step is dominant and the methods regions are similar to forward Euler region of stability.

2. When $\theta < 0.5$, the backward step is dominant and the methods regions are similar to backward Euler region of stability. They have A-stability and can be used for Stiff problems.

As θ tends to 0.5 from both the families the regions tend to match the left half semi plane, then the method becomes F-stable and it's best used for marginally stable problems, this tendency is visible from the progression $\theta = 0.1\,..\,0.3\,..0.4$

*In this figure and in the rest of this work, regions of stability which are outside the curve are displayed with a dashed line

Exercise 7

Consider the IVP $\dot{\mathbf{x}} = B\mathbf{x}$ with $B = [-180.5, 219.5; 179.5, -220.5]$ and $\mathbf{x}(0) = [1, 1]^T$ to be integrated in $t \in [0, 5]$. Notice that $\mathbf{x}(t) = e^{Bt}\mathbf{x}(0)$. 1) Solve the IVP using RK4 with $h = 0.1$; 2) Repeat point 1) using implicit extrapolation technique IEX4; 3) Compare the numerical results in points 1) and 2) against the analytic solution; 4) Compute the eigenvalues associated to the IVP and represent them on the $(h\lambda)$-plane both for RK4 and IEX4; 5) Discuss the results.

The results of the integration via RK4 and IEX4 and their comparison with the analytical solution are shown in Fig. 9. While IEX4 manage to maintain the integration errors contained, the relative error for RK4 grows exponentially.

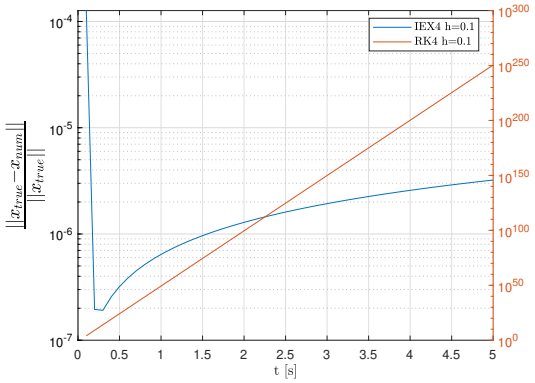


Figure 9: Accuracy error of RK4 and IEX4

The explanation of the result is understood by analysis of the problem eigenvalues which are respectively $\lambda_1 = -1$ and $\lambda_2 = -400$. Then the problem can be categorized as a stiff problem, due to the presence of very fast and slow dynamics. The integration by h=0.1 is then not enough to map both the eigenvalues in the region of stability of RK4 as shown in Fig. 10.

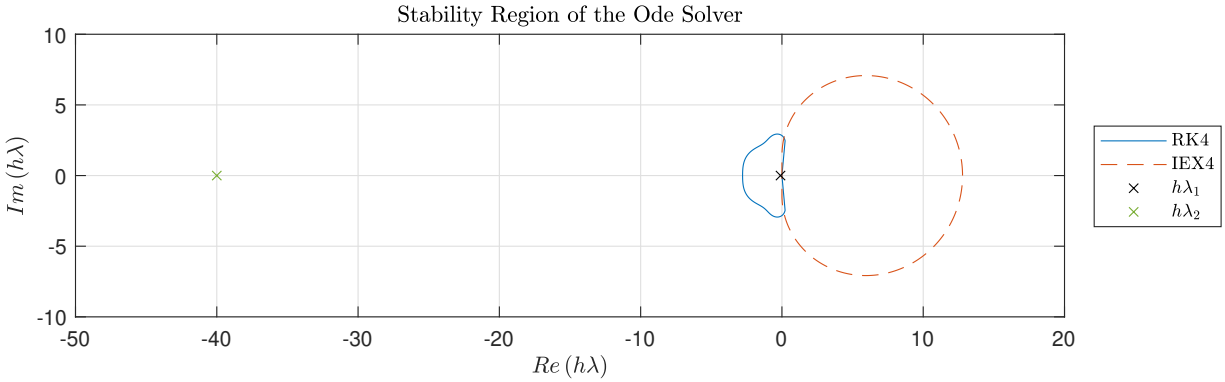


Figure 10: Region of stability and eigenvalues

IEX4 on the other hand is A-stable, which relieves the method from a choice of h to obtain asymptotic stability, h will then be governed only by accuracy requirements.

Exercise 8

Consider the two-dimensional IVP

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}-\frac{5}{2}\left[1+8\sin(t)\right]x\\ (1-x)y+x\end{bmatrix}, \qquad \begin{bmatrix}x(t_0)\\ y(t_0)\end{bmatrix} = \begin{bmatrix}1\\ 1\end{bmatrix}$$

1) Solve the IVP using AB3 in $t \in [0,3]$ for $h = 0.1$; 2) Repeat point 1) using AM3, ABM3, and BDF3; 3) Discuss the results.

It is useful to start the resolution by analysis of the analytical problem. Unfortunately the IVP is neither linear neither time invariant. Then a quick treatment using the results of the theory of LTI systems is precluded.
It can be seen that equation 1 is uncoupled from y, then it can be solved by separation of variable in the following equations.

$$\frac{dx}{x} = -\frac{5}{2}(1+8\sin t)dt \tag{8}$$

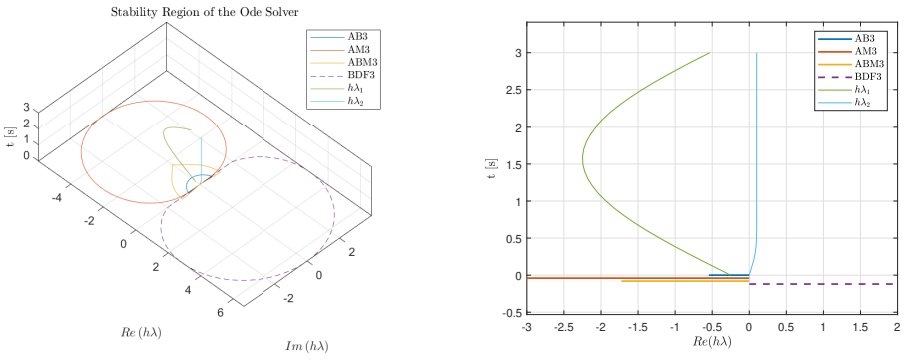
$$\ln|x| + C = -\frac{5}{2}t + 20\cos t \quad \text{with} \quad x(0) = 1 \quad C = 20 \tag{9}$$

$$x(t) = e^{-20}e^{-\frac{5}{2}t+20\cos t} \tag{10}$$

By substituting this solution into the system we can obtain a time variant linear ode system in Eq. (11).

$$\begin{bmatrix}\dot{x}\\ \dot{y}\end{bmatrix} = \begin{bmatrix}-\frac{5}{2}\left(1+8\sin(t)\right) & 0\\ 1 & (1-x(t))\end{bmatrix}\begin{bmatrix}x\\ y\end{bmatrix} \tag{11}$$

This system can then be analyzed as a time invariant ode system for various time instants, since it's an LU matrix the eigenvalues are on the diagonal. The results are visible in Fig. 11a and Fig. 11b. The second eigenvalue is always unstable for all the methods, as it should be since it is unstable also for the analytical system. The first eigenvalue instead is always stable for AM3 and BDF3, while for AB3 this eigenvalue rapidly becomes unstable.



(a) Multi-Step stability regions **(b)** Time evolution of $Re(h\lambda)$

Figure 11

At last for ABM3 it travels through an unstable region, this leads to a decrease of accuracy for ABM3 visible in Fig. 12, when compared to the other methods its relative error, even if in a mean value analysis is similar to the others, shows local spikes. All the methods were implemented with initial steps generated by lower order methods of the same family. In absence of an easily derivable analytic solution ode113 of the Matlab ode suite was used as a reference of the error. Since the accuracy order is the same for all methods the error is of the same magnitude, except for AB3 which exhibit a quick growth due to the early numerical instability.

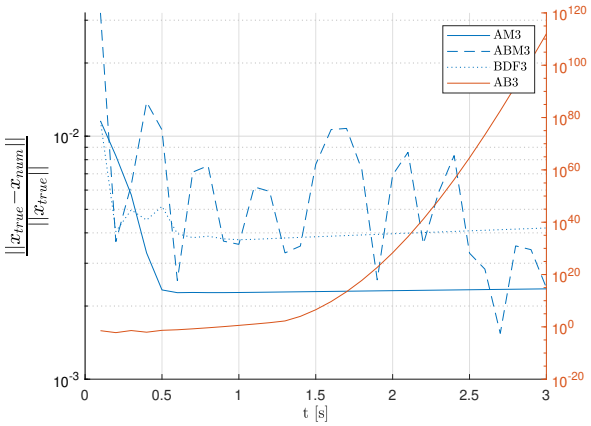


Figure 12: Accuracy Comparison of Multi-Step Methods

Since all methods are part of the multi-step family they all need one function evaluation per step, the implicit methods AM3 and BDF3 must also go through a zero finding problem which requires extra resources.