CHAPTER 113

Efficiency of fourth-order Runge-Kutta methods, Dormand Prince and Bulirsch-Stoer in solving initial value problems

10.56238/pacfdnsv1-113

Marco Aurelio Amarante Ribeiro
Master in Engineering and Management of Processes and Systems Institution: Institute of Technological Education – IETEC, Belo Horizonte, MG
E-mail: marcoaurelioamaranteribeiro@gmail.com

José Helvecio Martins
Ph.D. in Agricultural Engineering from Purdue University, United States Institution: Institute of Technological Education – IETEC, Belo Horizonte, MG
Email: j.helvecio.martins@gmail.com

Wanyr Romero Ferreira
PhD in Energy from Université Paul Sabatier, Toulouse, France Institution: Institution: Institute of Technological Education – IETEC, MG
Email: wanyr@terra.com.br

ABSTRACT
This article discusses the efficiency of the Runge-Kutta classical of fourth-order, Dormand-Prince and Bulirsch-Stoer numerical methods in solving initial value problems.

The three methods were compared by solving a problem of the suspension dynamics of a vehicle when passing over a speed bump in the lane. The problem is described by a second order ordinary differential equation. Results were obtained by varying the initial step size equally for the three methods, and the behavior of the vehicle suspension in response to the speed bump height was analyzed for each step size. It was concluded that it is essential to know the nature of the problem to be solved, to properly choose the numerical method and the size of the integration step to be used. The higher the order of the integration method, the greater the possibility of using a larger step size with the desired precision.
Therefore, knowledge of the nature of the problem is essential for choosing the solution method and the size of the integration step to obtain adequate results.

Keywords: numerical methods, ordinary differential equations, runge-kutta, dormand prince, bulirsch-stoer.

1 INTRODUCTION

Several problems encountered in the sciences, particularly in engineering, can be solved by mathematical modeling that results, for the most part, in differential equations. Ordinary differential equations (ODEs) are a subset of this universe, which constitute the formulation of initial value problems for a variety of processes and systems. The solution of these problems can be obtained by solving the ODEs that describe the dynamics of the system (GEAR, 1971, NAGLE; SAFF; SNIDER, 2018, DUTTA et al., 2020). One of the fundamental tools to obtain these solutions are numerical methods (LAPIDUS; SEINFELD, 1971, HULL et al., 2006, BISWAS; CHATTERJEE; MUKHERJEE, 2013, BRONSON; COSTA, 2014).

One of the advantages of modeling processes using ODEs is that they can be solved using numerical integration methods, without the need to focus deeply on the theoretical aspects of the mathematical models used (STOER; BULIRSCH, 2002, QUARTERONI; SACCO; SALERI, 2007, PRESS et al., 2007).

Studies of methods of solving ODEs showed over time that most of them could not be solved by simple integration or analytical solutions alone (BRONSON; COSTA, 2014, NAGLE; SAFF; SNIDER, 2018). In many or most cases, it became necessary to use numerical methods to obtain an approximate

solution, with desired accuracy, using efficient computational algorithms, enabling the solution of complex problems.

Methods for solving ODEs can be seen as originating from different models, which facilitates the understanding of the problem, and computation is closely linked to the origin of the problem and the use that will be made of the answers. Therefore, modeling and computational methods are not steps to be taken separately from reality and is embedded in a broader context, which involves numerical analysis. In this context, a numerical method is a mathematical tool designed to solve numerical problems and its implementation, with an appropriate convergence check, consists of the creation of a numerical algorithm implemented in a programming language or on a computational platform (QUARTERONI; SACCO; SALERI, 2007, ZILL, 2018).

Among the first numerical methods developed, Euler's methods stand out (BOYER, 1996), which, due to their simplicity, are used to facilitate the understanding of the concepts involved in numerical integration methods, besides providing satisfactory results for some problems, under certain conditions and constraint (CORLESS, 2000, BISWAS et al., 2013, SHAMPINE).

Several other methods have been developed, which provide more accurate results for more complex problems, although they require more sophisticated algorithms and longer computational time. Among the most modern classical numerical methods are the Runge-Kutta methods, and its improvements, and the Bulirsch-Stoer method (BUTCHER, 2000, PRESS et al., 2007).

Several works with applications of numerical methods for the solution of various types of problems are found in the literature. However, details on the development and theoretical foundation of these methods are found in classic textbooks (GEAR, 1971, NAGLE; SAFF; SNIDER, 2012, ZILL, 2018).

Technical-scientific articles on the subject are generally found and presented in dissertations and theses and do not contain theoretical details about the methods used to obtain the results. However, these articles serve as a reference for comparison of results and validation of new methodologies or approach techniques to solve new problems (VANANI; AMINATAEI, 2011, OBERLEITHNERA; PASCHEREITA; SORIAB, 2015, DUTTA et al., 2020; GHANBARI; BALEANU, 2020, HOSSEINI et al., 2020, BOSCH NETO et al., 2020).

In this context, this paper presents a comparison of the efficiency of the classical fourth-order Runge-Kutta method, Dormand-Prince method, and Bulirsch-Stoer method in solving the problem of suspension dynamics of a vehicle.

2 LITERATURE REVIEW

A problem involving ordinary differential equations, generically, is reduced to the study of a set of N coupled first-order differential equations for a function y_i, $i=1,2,\cdots,N$, expressed by the following general form (PRESS et al., 2007):

$$\frac{dy_i(x)}{dx} = f_i'(x, y_1, \cdots, y_N), \quad i = 1, \cdots, N$$

The functions f_i^' on the right-hand side in Equation (1) are known. The quotation mark (') here does not mean derivative, but just a notation to remember that the functions f_i^' are the derivatives of y_i (x).

A problem involving ordinary differential equations is not completely defined by its equations, because it depends on the nature of the initial conditions, which is crucial in determining the best way to approach it. Initial conditions are algebraic conditions on the values of the function y_i (x) in Equation (1) that can generally be satisfied at specific discrete points, but are not automatically preserved by the differential equations (NAGLE; SAFF; SNIDER, 2012, PRESS et al., 2007, ZILL, 2018).

Instead of being given initial conditions, boundary conditions can be presented along with the equation, and they can be as simple as requiring certain variables to have certain numerical values, or as complex as a set of nonlinear algebraic equations in their variables. Usually, it is the boundary conditions that determine which numerical methods are feasible, and are divided into two broad categories (PRESS et al., 2007, NAGLE; SAFF; SNIDER, 2012, ZILL, 2018):

Initial value problems-in this case, all y_i (x) are given for some initial value x_0, and one wishes to obtain the y_i at some point for some final value of the independent variable, x_f, or at some list of discrete points;

Two-point boundary value problems - here the boundary conditions are specified at more than one value of x. Typically, some of the conditions are specified at x_0 and the rest at x_f.

2.1 FOURTH-ORDER RUNGE-KUTTA METHOD

The fourth-order Runge-Kutta method is the most popular of the explicit one-step methods, described by Equation (2). Without going into details, a schematic illustration of this method is presented in Figure 1 (PRESS et al., 2007, NAGLE; SAFF; SNIDER, 2012, ZILL, 2018).

$$k_1 = h \cdot f(x_n,\ y_n)$$

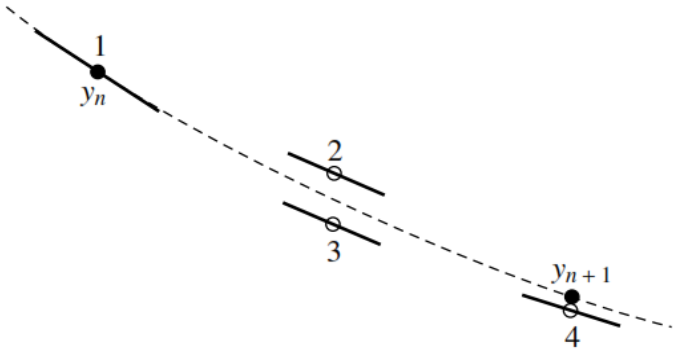
$$k_2 = h \cdot f(x_n + 0{,}5 \cdot h,\ y_n + 0{,}5 \cdot k_1)$$

$$k_3 = h \cdot f(x_n + 0{,}5 \cdot h,\ y_n + 0{,}5 \cdot k_2)$$

$$k_4 = h \cdot f(x_n + h,\ y_n + k_3)$$

$$y_{n+1} = y_n + \frac{(k_1 + 2 \cdot k_2 + 2 \cdot k_3 + k_4)}{6}$$

Figure 1. Schematic illustration of the fourth-order Runge-Kutta method.



Source: PRESS et al., 2007 (adapted).

2.2 DORMAND-PRINCE METHOD

The seven-stage Dormand-Prince method with local error estimation and interpolator is presented in a Butcher framework (Table 1). The solution is advanced with y_(n+1) of order five, a procedure called local extrapolation, and the solution yˆ_(n+1) of order four is used to obtain the local error estimate via the difference y_(n+1)-yˆ_(n+1). In fact, yˆ_(n+1) is not calculated, instead, the coefficients in the b^T - bˆ^T row in Butcher's framework are used to obtain the local error estimate. The sixth constant (C_6) in the higher-order error term is minimized, maintaining stability. Six stages are needed for the order 5 method and for the seventh stage it is necessary to have an interpolator, which is the last row in Table 1 (ASHINO; NAGASE; VAILLANCOURT, 2000).

This seven-stage method, in practice, reduces to six stages because k_1^[n+1] =k_7^[n] , and the row vector b^T is equal to the seventh row corresponding to k_7. The absolute stability range of this method is approximately (-3,3,0), as shown in Figure 2. This method is implemented in computational subroutines in several platforms, such as MATLAB (ASHINO; NAGASE; VAILLANCOURT, 2000), OCTAVE and SCILAB.

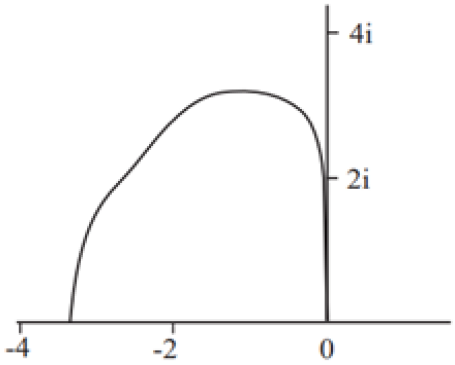
Table 1. Butcher's framework for the Seven Stage Dormand-Prince method with Interpellor

	C	A					
k_1	0	0					
k_2	$\frac{1}{5}$	$\frac{1}{5}$	0				
k_3	$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$	0			
k_4	$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$	0		
k_5	$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$	0	
k_6	1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$	0

k_7	1	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$
$\hat{y}_{n+1}$	$\hat{b}^T$	$\frac{5179}{57600}$	0	$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{1}{40}$
y_{n+1}	b^T	$\frac{35}{384}$	0	$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	0
$b^T - b^T$		$\frac{71}{57600}$	0	$-\frac{71}{16695}$	$\frac{71}{1920}$	$-\frac{17253}{339200}$	$-\frac{1}{40}$
$y_{n+0,5}$		$\frac{5783653}{57600000}$	0	$\frac{466123}{1192500}$	$-\frac{41347}{1920000}$	$\frac{16122321}{339200000}$	$\frac{183}{10000}$

Source: ASHINO; NAGASE; VAILLANCOURT, 2000 (Adapted).

Figure 2. Absolute stability region of the Dormand-Prince method.



Source: ASHINO; NAGASE; VAILLANCOURT, 2000.

2.3 BULIRSCH-STOER METHOD

To understand this method it is necessary to discuss the modified midpoint method, which has its most important application in the Bulirsch-Stoer Method. Another technique that can be combined with the Bulirsch-Stoer method is the Richardson Extrapolation. A brief discussion of these methods is presented below (STOER; BULIRSCH, 2002, PRESS et al., 2007).

2.3.1 Modified Midpoint Method

The modified midpoint method consists of advancing a vector of dependent variables, y(x), from a point whose abscissa is x to x+H by a sequence of n sub-steps of size h each, given by Equation (3):

$$h = \frac{H}{n}$$

The number of right-hand side evaluations required by the modified midpoint method is n+1. The formulas for this method are:

$$z_0 = y(x)$$

$$z_1 = z_0 \cdot h \cdot f(x, z_0)$$

$$z_{m+1} = z_{m-1} + 2h \cdot f(x + m \cdot h,\ z_m); \qquad para\ m = 1, 2, \cdots\ n-1$$

$$y(x + H) \approx y_n \equiv \frac{1}{2}\left[z_n + z_{n-1} + h \cdot f(x + H, z_n)\right]$$

The variables z are intermediate approximations that march along the h steps, while y_n is the final approximation of y(x+H). This is basically a centralized difference or midpoint method, except for the first and last points, which give the method the modicated qualifier.

The modified midpoint method is second-order with the advantage that it asymptotically requires, for large n, only one evaluation of the first derivative per step h, instead of the two evaluations required by the second-order Runge-Kutta method. In general, the use of the modified midpoint method alone is outperformed by the Runge-Kutta method with adaptive step-size control built in.

The usefulness of the modified midpoint method for the Bulirsch-Stoer technique derives from the fact that it obtains, from Equation (4), a detailed result (STOER; BULIRSCH, 2002). It turns out that the error in Equation (4), expressed as a series of powers of h, contains only even powers of h:

$$y_n - y(x+H) = \sum_{i=1}^{\infty} \alpha_i \cdot h^{2i} \quad (5)$$

The value of H is held constant, but the value of h changes when n varies in Equation (3). The importance of these even power series is that if the usual tricks of combining steps and eliminating higher-order error terms are applied, two more orders of accuracy can be obtained at a time. This process allows higher-order accuracy to be obtained with fewer evaluations of the derivative function, in general half as many as with the fourth-order Runge-Kutta method (PRESS et al., 2007).

2.3.2 Richardson Extrapolation

Richardson extrapolation uses the powerful idea of extrapolating a computed result to the value that would have been obtained if the step size had been much smaller than it actually was. In particular, extrapolation to step size zero is the desired goal. The first practical ODE integrator that implemented this idea was developed by Bulirsch and Stoer, and thus the extrapolation methods are often called Bulirsch-Stoer Methods (PRESS et al., 2007).

2.4 FORMULATION OF THE BULIRSCH-STOER METHOD

The sequence of separate attempts to cross the H-interval is made with increasing number of sub-steps, n. Bulirsch and Stoer originally proposed the following sequence (PRESS et al., 2007):

$$n = 2,4,6,8,12,16,24,32,48,64,96,\cdots,[n_j = 2n_{(j-2)}],\cdots$$

According to Deuflhard (1983, 1985) the following sequence was more efficient:

$$n = 2,4,6,8,10,12,14,\cdots,[n_j = 2j],\cdots$$

In this sequence, for each step, it is not known in advance where it will end up. After each successive intermediate step, n, a polynomial extrapolation is performed. This extrapolation gives extrapolated values

and error estimates. If the error values are not satisfactory, the value of n is increased. If they are satisfactory, one proceeds to the next step and starts again with n=2 (PRESS et al., 2007).

There is some upper bound, beyond which one concludes that there is some obstacle in the path along the interval H, so one should reduce H instead of just subdividing it more finely, because loss of accuracy occurs if too fine a subdivision is chosen. In algorithm implementations using this method, n is usually taken to be equal to eight. The eighth value in the sequence expressed by Equation (7) is 16, which is the maximum number of subdivisions of H allowed (PRESS et al., 2007).

Error control is enforced by monitoring internal consistency and adapting the step size to match a prescribed limit on the local truncation error. Error control is represented in Table 2 by a kind of genetic algorithm, expressed in the form of Equation (8), where P represents a "daughter" and x represents its "parents".

Table 2. Symbolic representation of Neville's algorithm.

x_0:	$y_0 = P_0$				(8)
		P_{01}			
x_1:	$y_1 = P_1$		P_{012}		
		P_{12}		P_{0123}	
x_2:	$y_2 = P_2$		P_{123}		
		P_{23}			
x_3:	$y_3 = P_3$				

Neville's algorithm is a recursive way to fill in the numbers in Table 2, one column at a time, from left to right. It is based on the relationship between a "daughter" P and its "parents" x, as per Equation (9), that this recursion works, because the two "parents" already agree on the points x_(i+1) ⋯ x_(i+m-1).

$$P_{i(i+1)\ldots(i+m)} = \frac{(x - x_{i+m})P_{i(i+1)\ldots(i+m-1)} + (x_i - x)P_{(i+1)(i+2)\ldots(i+m)}}{(x_i - x_{i+m})}$$

Each new result of the modified midpoint integration sequence allows a frame, such as Table 2, to be extended by an additional set of diagonals, written as a lower triangular matrix, expressed by Equation (10):

$$T_{00} \quad T_{10}\, T_{11} \quad T_{20} \quad T_{21}\, T_{22}$$
$$\ldots \quad \ldots \quad \ldots$$

Where T_k0=y_k, and y_k=y(x_n+H) calculated with step size h_k=H/n_k . By substituting P=T, x_i=h_i^2, and x=0 into Equation (10), Neville's algorithm can be rewritten as:

$$T_{k,j+1} = T_{kj} + \frac{T_{kj} - T_{k-1,j}}{\left(n_k / n_{k-j}\right)^2 - 1}, \quad \forall\ j = 0, 1, \cdots, k-1$$

At each new step size h_i a new row in the table is started, and then the polynomial extrapolation fills the rest of the row. Each new element in the table comes from the two closest elements in the previous column. The elements in the same column have the same order, and T_kk, the last element in each row, is the highest order approximation of accuracy at that step size. The difference between the last two consecutive elements is taken as the (conservative) error estimate. This error estimate can be used to adjust the step size. A good strategy was originally proposed by Deuflhard (1983, 1985), and later modified, details of which can be obtained from the literature (LAPIDUS; SEINFELD, 1971, HAIRER; WANNER; NØRSETT, 1993, STOER; BULIRSCH, 2002, PRESS et al., 2007).

3 METHODOLOGY

3.1 VEHICLE SUSPENSION SYSTEM MODEL

The modeling of the suspension of a vehicle can be done by considering the vehicle driving over a hump. The force that the bump exerts on the suspension is described by the expression:

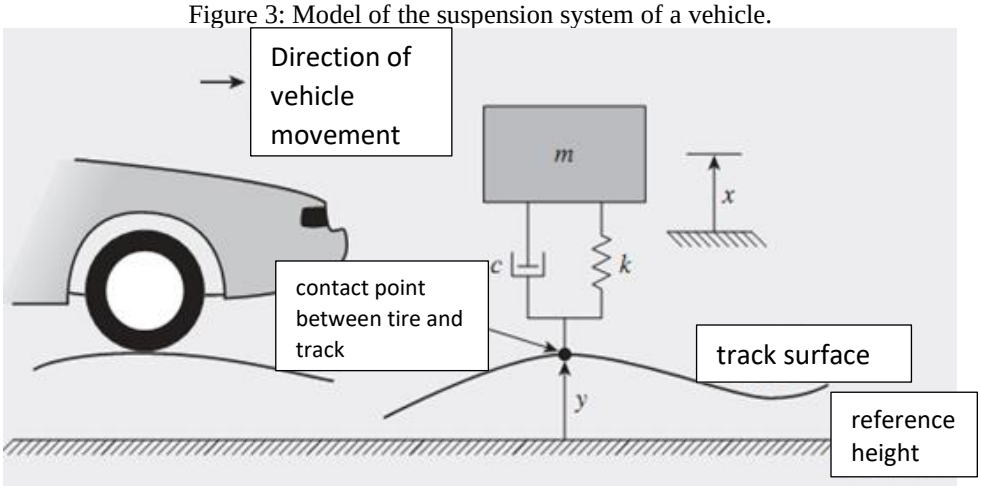
$$F_s(t) = a \cdot y(t) + b \cdot \frac{dy(t)}{dt}$$

Where:

F_s (t) = Force on the suspension, [N];
y(t) = Displacement caused by the track surface, [m];
a = Constant;
b = Constant.

The suspension model of the vehicle, considering a quarter of its size for symmetry, is represented in Figure 3.

Figure 3: Model of the suspension system of a vehicle.



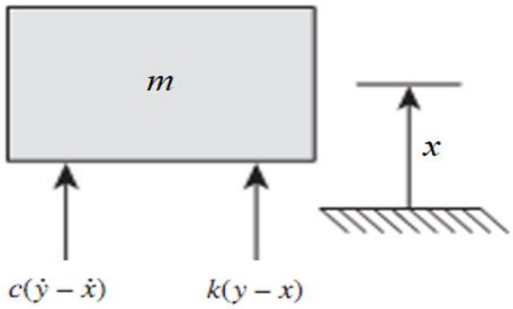
Source: ÇENGEL; PALM III, 2014 (Adapted).

In the simplified model, the masses of the wheel, tire, and axle are neglected and, for symmetry, the mass m represents a quarter of the vehicle mass. The spring constant k represents a combination of the effects of the tire and the suspension spring. The constant c represents the damping constant of the shock absorber. The equilibrium position of mass m, when y=0, is x=0.

The displacement caused by the track surface, y(t), can be obtained from the profile of the track surface and the vehicle speed. The equation of motion of the vehicle is obtained by means of the free body diagram, shown in Figure 4, considering only the dynamic force, because the static force is cancelled out by the gravitational force:

$$m \cdot \frac{d^2x(t)}{dt^2} + c \cdot \frac{dx(t)}{dt} + k \cdot x(t) = k \cdot y(t) + c \cdot \frac{dy(t)}{dt}$$

Figure 4 - Free-body diagram, assuming that $\frac{dy}{dt} > \frac{dx}{dt}$ e $y > x$.



Source: ÇENGEL; PALM III, 2014 (Adapted).

3.2 SIMULATING THE VEHICLE RESPONSE TO THE RISE OF A SPEED BUMP

The response of a vehicle to the lift of a given hump with height h and length L, moving at a speed v, can be simulated by solving the differential equation of motion, Equation (13). This requires the elevation profile of the hump, in this case given by (ÇENGEL; PALM III, 2012):

$$y\big(z(t)\big) = 5{,}4366 \cdot z(t) \cdot exp\big(-4 \cdot z(t)\big)$$

Whereby:

$y\big(z(t)\big)$ = Spine elevation profile, $[m]$;

z = Horizontal distance traveled by the vehicle over the elevation, $[m]$;

The displacement y(t) felt by the suspension is related to y(z(t)), by means of the velocity, where z(t)=v(t)·t. Substituting z(t) into Equation (14) gives:

$$y(t) = 97{,}859 \cdot t \cdot exp(-72 \cdot t)$$

For this work we used the data contained in Table 1. Combining Equation (14) with Equation (15) and substituting the data contained in Table 1, we obtain the equation of motion for this problem:

$$\frac{d^2x(t)}{dt^2} + \frac{5.000}{240} \cdot \frac{dx(t)}{dt} + \frac{16.000}{240} \cdot x(t) = \left[\frac{489.290 - (33.663.152) \cdot t}{240} e^{-72 \cdot t}\right]$$

Table 1. Data used in the solution to the problem.

Parameter	Description	Value	Unit
m	Mass of the vehicle	240	$[kg]$
c	Damping constant	5.000	$\left[\frac{N \cdot s}{m}\right]$
k	Elastic constant of the spring	16.000	$\left[\frac{N}{m}\right]$
v	Vehicle speed	18	$[m/s]$
h	Spine Height	0,5	$[m]$
L	Spine length	1,0	$[m]$

This is a second order initial value problem, so the solution of Equation (16) was performed using the fourth order Runge-Kutta and Dormand-Prince methods (BUTCHER, 2000, ASHINO; NAGASE; VAILLANCOURT, 2000) and Bulirsch-Stoer (STOER; BULIRSCH, 2002, PRESS et al., 2007), transforming it into a system of two first order equations of the form:

$$\{\frac{dx_1}{dt} = x_2 \quad \frac{dx_2}{dt} = C_1 \cdot x_2 + C_2 \cdot x_1 + f(x)$$

4 RESULTS AND DISCUSSION

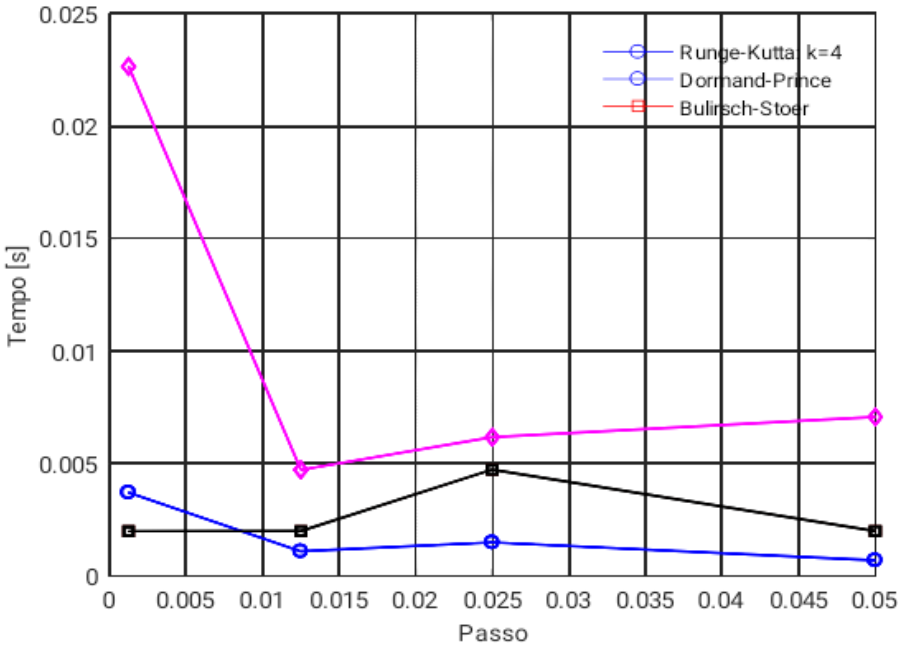
The computational time results and final solution estimates are given in Table 2 for various values of the integration step for the Runge-Kutta methods of order 4, Dormand-Prince and Bulirsch-Stoer. The variation of computational time as a function of step size is shown in Figure 5.

Table 2. Computational time and final estimates of the solution of the one vehicle suspension system problem.

Step	Computational Time, $[s]^*$		
	$RK4$	DP	BS
$h = 0{,}05000$	0,00070	0,00200	0,00707
$h = 0{,}02500$	0,00149	0,00473	0,00617
$h = 0{,}01250$	0,00110	0,00200	0,00472
$h = 0{,}00125$	0,00371	0,00199	0,02264
Step	**Final Estimate**		
	$RK4$	DP	BS
$h = 0{,}05000$	1.0355×10^{-1}	$9{,}3881 \times 10^{-4}$	$9{,}3304 \times 10^{-4}$
$h = 0{,}02500$	$9{,}7124 \times 10^{-3}$	$9{,}3881 \times 10^{-4}$	$9{,}3304 \times 10^{-4}$
$h = 0{,}01250$	$1{,}5333 \times 10^{-3}$	$9{,}3881 \times 10^{-4}$	$9{,}3304 \times 10^{-4}$
$h = 0{,}00125$	$9{,}3310 \times 10^{-4}$	$9{,}3881 \times 10^{-4}$	$9{,}3304 \times 10^{-4}$

* ***RK4:*** *Runge-Kutta, ordem 4;* ***DP****: Dormand-Prince;* ***BS****: Bulirsch-Stoer*

Figure 5: Computational time for the solution of a vehicle suspension system problem as a function of integration step size.



It can be observed in Figure 5 that the behavior of the computational time in relation to the step size did not present a well defined functional trend. It was not possible, in this work, to find a reasonable explanation for this randomness, although it is thought that it may be an effect of the transient state of the computer processor used, influenced by some electromagnetic event. What was effectively verified was that different processing time values were obtained for the same problem when the processing was repeated.

It can be observed, however, that the computational time for the fourth order Runge-Kutta method, in general, was the lowest and for the Bulirsh-Stoer method was the highest, with the Dormand-Prince method in an intermediate position, but all presented coherent results. On the other hand, these results depend on the nature of the problem in relation to the method used. Therefore, for a different problem these results can be completely opposite. The final solutions are represented graphically in Figures 6, 7, 8 and 9.

Figure 6. Numerical solutions of the one vehicle suspension system problem using the fourth order Runge-Kutta method (k = 4), with fixed step, and the adaptive step methods of Dormand-Prince and Bulirsch-Stoer, with h = 0.05.

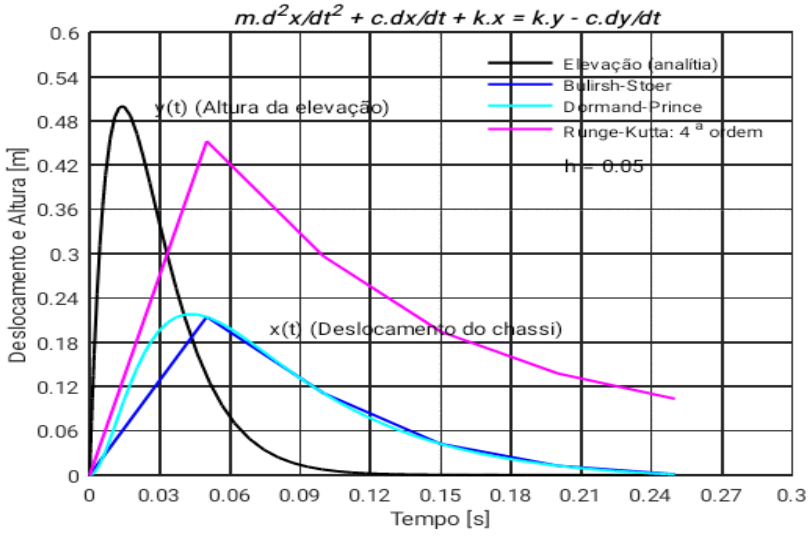


Figure 7. Numerical solutions of the one-vehicle suspension system problem using the fourth-order Runge-Kutta method (k = 4), with fixed pitch, and the adaptive pitch methods of Dormand-Prince and Bulirsch-Stoer, with h = 0.025.

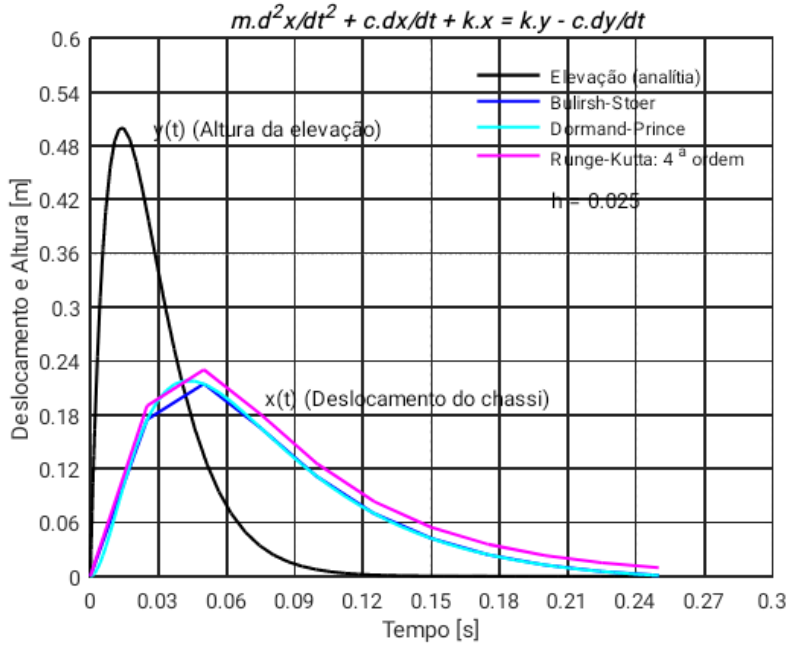


Figure 8. Numerical solutions of the one-vehicle suspension system problem using the fourth-order Runge-Kutta method (k = 4), with fixed pitch, and the adaptive pitch methods of Dormand-Prince and Bulirsch-Stoer, with h = 0.0125.

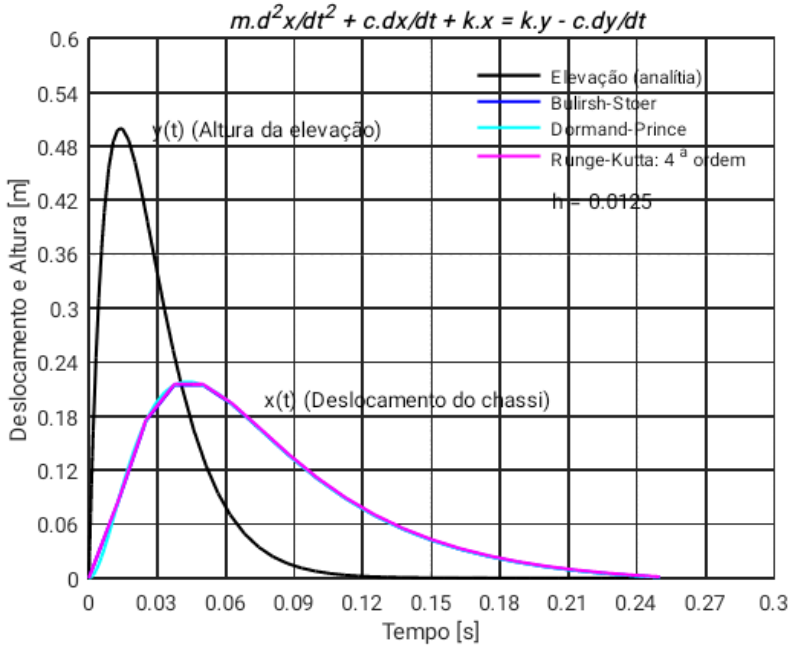
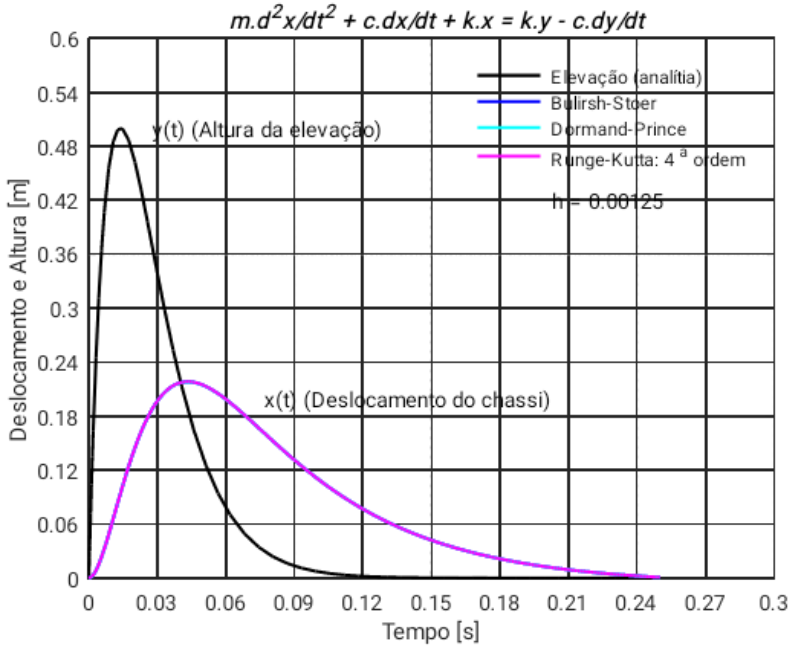


Figure 9. Numerical solutions of the one-vehicle suspension system problem using the fourth-order Runge-Kutta method (k = 4), with fixed pitch, and the adaptive pitch methods of Dormand-Prince and Bulirsch-Stoer, with h = 0.00125.



It can be seen in Figures 6 to 9 that, for all methods, there was an improvement in the accuracy of the responses of the vehicle chassis displacement as a function of time, with the reduction of the integration step. However, it can be observed that the Dormand-Prince method presented, practically, the same response, regardless of the size of the integration step. Therefore, this method depends very little on the initial integration step, because of the step size adaptation process. In this problem, the fourth-order Runge-Kutta and Bulirsch-Stoer methods presented feasible solutions only for $h<0.01$, in the domain $0 \leq t \leq 0.25$.

5 CONCLUSIONS

It is essential to know the nature of the problem to be solved, to properly choose the numerical method and the size of the integration step to be used. The higher the order of the integration method, the greater the possibility of using a larger step size with desired accuracy.

The Dormand-Prince method showed virtually the same response regardless of the integration step size. Therefore, this method depends very little on the initial integration step, because of the step size adaptation process.

Methods with a step adaptation algorithm, such as the Bulisrch-Stoer and Dormand-Prince methods, are not very sensitive to the initial step size because they are modified throughout the process to maintain accuracy.

In general, all types of methods can be applied to any initial value problem, each with its own set of pros and cons, which should be understood before they are used.

In all numerical methods, generally, accuracy increases with decreasing step size, especially in fixed-step methods. Therefore, adaptive pitch methods such as the Dormand-Prince and Bulirsch-Stoer methods generally provide more accurate estimates, provided they are used properly.

Therefore, it is concluded that knowledge of the nature of the problem is critical in choosing the solution method and the size of the integration step to obtain adequate results.

REFERENCES

Ashino, r.; nagase, m.; vaillancourt, r. Behind and beyond the matlab ode suite. Crm-2651. January, 2000.
Biswas, b. N.; chatterjee, s.; mukherjee, s. P.; pal, s. A discussion on euler method: a review. Electronic journal of mathematical analysis and applications vol. 1(2) july 2013, pp. 294-317. Issn: 2090-792x (online) http://ejmaa.6te.net/.

Bosch neto, j. C.; barros, j. E. M.; artur, a. P. S.; abreu, b. M. P. N.; parreira, c. C. A. Emissions simulation by coupling chemical equilibrium and reduced kinetics for gasoline/ethanol mixture in ic engines. Braz. J. Of develop., curitiba, v. 6, n. 10, p.76926-76946, oct. 2020. Issn 2525-8761.

Boyer, c. B. História da matemática. Tradução: elza f. Gomide. 2a ed. São paulo: editora edgard blücher, 1996.

Bronson, r.; costa, g. B. Differential equations. Fourth edition, schaum's outline series, mcgraw-hill education, 2014. 385p.

Butcher, j. C. Numerical methods for ordinary differential equations in the 20th century. Journal of computational and applied mathematics, v. 25, issues 1–2, 15 december, 2000, p. 1-29.

Çengel, y. A.; palm iii, w. J. Differential equations for engineers and scientists. Mcgraw-hill education, 2012. 611p.

Deuflhard, p. Order and stepsize control in extrapolation methods. Numerische mathematik, 41(3): p.399–422. 1983. Doi:10.1007/bf01418332, issn 0029-599x.
Deuflhard, p. Siam review, v. 27, pp. 505–535. 1985.

Dutta, h; günerhan, h.; ali, k. K.; yilmazer, r. Exact soliton solutions to the cubic-quartic non-linear schrödinger equation with conformable derivative. Mathematical and statistical physics, frontiers in physics, march 2020. Https://doi.org/10.3389/fphy.2020.00062.

Gear, c. W. Numerical initial value problems in ordinary differential equations. New york: prentice hall, 1971.

Ghanbari, b.; baleanu, d. New optical solutions of the fractional gerdjikov-ivanov equation with conformable derivative. Mathematical and statistical physics, frontiers in physics, may 2020. Https://doi.org/10.3389/fphy.2020.00167.

Hairer, e.; wanner, g.; nørsett, s. P. Solving ordinary differential equations i: nonstiff problems. Springer series in computational mathematics, berlin, heidelberg, 1993. Doi: https://doi.org/10.1007/978-3-540-78862-1.

Hosseini, k.; mirzazadeh, m.; osman, m. S.; al qurashi, m.; baleanu, d. Solitons and jacobi elliptic function solutions to the complex ginzburg–landau equation. Mathematical and statistical physics, frontiers in physics, june 2020. Https://doi.org/10.3389/fphy.2020.00225.

Hull, t. E., enright, w. H., fellen, b. M., sedgwick, a. E. Comparing numerical methods for ordinary differential equations. Journal on numerical analysis, 9(4). Siam, p.603–637, 2006. Https://doi.org/10.1137/0709052.

Lapidus, l.; seinfeld, j. H. Numerical solution of ordinary differential equations. New york: academic press. 1971.

Nagle, r. K.; saff, e. B.; snider, a. D. Fundamentals of differential equations and boundary value problems. Sixth edition, addison-wesley, pearson education, boston, ma, usa, 2012. 810p.

Oberleithnera, k.; paschereita, c. O.; soriab, j. Stability analysis of time-averaged jet flows: fundamentals and application. Elsevier. Iutam-abcm symposium on laminar turbulent transition, procedia iutam, vol. 1, 2015. P.141–146.

Press, w. H.; teukolsky, s. A.; vetterling, w. T.; flannery, b. P. Section 17.3. Richardson extrapolation and the bulirsch-stoer method. Numerical recipes: the art of scientific computing (3rd ed.). New york: cambridge university press. 2007. Isbn 978-0-521-88068-8.

Quarteroni, a.; sacco, r.; saleri, f. Numerical mathematics. Texts in applied mathematics, springer, 2007. 654p.

Shampine, l. F.; corless, r. M. Initial value problems for odes in problem solving environments. Journal of computational and applied mathematics 125, p.31–40, elsevier, 2000.

Stoer, j.; bulirsch, r. Introduction to numerical analysis, 3rd ed. New york: springer, chapter 7, 2002.
Vanani, s. K.; aminataei, a. Numerical solution of differential algebraic equations using a multiquadric approximation scheme. Elsevier. Mathematical and computer modelling, vol. 53, n. 5–6, p. 659-666. 2011. Https://doi.org/10.1016/ j.mcm.2010.10.002.

Zill, d. G. Differential equations with boundary value problems. Ninth edition, cengage learning, boston, ma, usa, 2018. 522p.