

Numerical Methods in Classical Mechanics: Differential Equations

Author: Germán F. Molins Marconi

Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.

Advisor: Dr. Jesús M. González Miranda

(Dated: June 12, 2018)

Abstract: A revision of different first order ODE numerical integration schemes is presented in the ambit of classical mechanics. Their performance is tested on a rescaled SHO, and their traits and efficiency discussed. From these, an RK4 method is chosen to study a Duffing-Holmes oscillator. Its nonlinearity is shown to cause a period-doubling route to chaos through the exploration of a particular range of the forcing amplitude parameter using a bifurcation diagram.

I. INTRODUCTION

Many simply posed problems in classical mechanics are just rendered inaccessible through pure analytical resources, as most realistic models are. Numerical resolutions remediate such situations, and since the advent of computers they have become an essential tool in absolutely any branch of science.

Today's computational technology opens manifold possibilities on the quest for refined and more efficient numerical algorithms, whose mathematical foundation is indeed analytical. Any new worth discovery in this field is bound to have broad positive impact. With this motivation we review some popular low and middle order ODE integration methods, addressed to efficiently solve the motion of any system typically found in undergraduate courses. We then test their performance on an well-known system, comparing results with the exact solution, and choose one of them to solve a nonintegrable case of interest.

II. INTEGRATION METHODS

The context of the present work is a classical system with $2d$ degrees of freedom in phase space, for which we aim to provide a numerical solution to its equations of motion, namely, Hamilton's equations for canonical variables- coordinates q_i and their conjugate momenta p_i ,

$$\dot{p}_i = -\frac{\partial H}{\partial q_i}, \quad \dot{q}_i = \frac{\partial H}{\partial p_i}, \quad (1)$$

with initial conditions $\mathbf{p}(0) = \mathbf{p}_0$ and $\mathbf{q}(0) = \mathbf{q}_0$. They comprise a $2d$ coupled first-order ODE system.

In the following we summarize some elementary numerical methods for solving (1)- and any general coupled first-order ODE system, which consist of recursively updating the coordinates at a time $t_{n+1} \equiv (n+1)\Delta t = t_n + \Delta t$, where Δt is the integration step, using the values calculated for previous instants. The discretized points will be denoted as $\mathbf{p}(t_n) \equiv \mathbf{p}_n$ and $\mathbf{q}(t_n) \equiv \mathbf{q}_n$. We remark these methods can be used on any dynamical system, comprising (1) as a particular case.

A. Truncation methods

We first present some truncation methods [1] suitable for low dimensionality systems described in rectangular coordinates, $(\mathbf{p}, \mathbf{q}) \equiv (\mathbf{x}, \mathbf{v})$,

$$\dot{\mathbf{v}} = \mathbf{a}(\mathbf{v}, \mathbf{x}), \quad \dot{\mathbf{x}} = \mathbf{v}(t),$$

without any accuracy or efficiency ambition.

One popular second order method is the *last-point approximation* (LPA),

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \mathbf{a}_n \Delta t, \quad \mathbf{x}_{n+1} = \mathbf{x}_n + \mathbf{v}_{n+1} \Delta t.$$

Still further accuracy provides the *second-order Taylor approximation* (STA),

$$\begin{aligned} \mathbf{x}_{n+1} &= \mathbf{x}_n + \mathbf{v}_n \Delta t + \frac{1}{2} \mathbf{a}_n \Delta t^2, \\ \mathbf{v}_{n+1} &= \mathbf{v}_n + \frac{1}{2} (\mathbf{a}_n + \mathbf{a}_{n+1}) \Delta t. \end{aligned}$$

A detailed discussion of these and other elementary methods can be found in [1], who indicates that generally an adaptative time-step STA would be the better choice among the basic truncation methods.

B. Runge-Kutta methods

Let us now focus on the ODE system (1), now generalized to arbitrary phase space dimensionality (not necessarily even) and derivative functionality,

$$\dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y}), \quad (2)$$

As described by [2], the general solution of (2),

$$\mathbf{y}(t) = \mathbf{y}_0 + \int_0^t d\tau \mathbf{f}(\tau, \mathbf{y}(\tau)), \quad (3)$$

can be extended to one discretized integration interval as

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta t \int_0^1 d\tau \mathbf{f}(t_n + \tau \Delta t, \mathbf{y}(t_n + \tau \Delta t)),$$

where the integral is to be replaced by a quadrature formula, which may be implicit or explicit. Hence arise

Runge-Kutta (RK) methods, especially convenient for arbitrary dimensionality and generalized coordinates with low to medium accuracy requirements, but still far more efficient than any truncation method.

One widespread possibility is the fourth-order explicit method (RK4) listed by [3, §25.5],

$$\begin{aligned} \mathbf{k}_1 &= \Delta t \mathbf{f}(t_n, \mathbf{y}_n) \\ \mathbf{k}_2 &= \Delta t \mathbf{f}(t_n + \frac{1}{2}\Delta t, \mathbf{y}_n + \frac{1}{2}\mathbf{k}_1) \\ \mathbf{k}_3 &= \Delta t \mathbf{f}(t_n + \frac{1}{2}\Delta t, \mathbf{y}_n + \frac{1}{2}\mathbf{k}_2) \\ \mathbf{k}_4 &= \Delta t \mathbf{f}(t_n + \Delta t, \mathbf{y}_n + \mathbf{k}_3) \\ \mathbf{y}_{n+1} &= \mathbf{y}_n + \frac{1}{6}\mathbf{k}_1 + \frac{1}{3}\mathbf{k}_2 + \frac{1}{3}\mathbf{k}_3 + \frac{1}{6}\mathbf{k}_4 + \mathcal{O}(\Delta t^5). \end{aligned}$$

The sequence in which variables $y_i(t)$ are updated is irrelevant, and best performance is achieved when using an adaptative time-step, which is easily implemented, for only the last computed point is used at every new step.

C. Predictor-corrector methods

They are implicit iterative schemes, belonging to the more general family of multistep methods. Their underlying idea is to use a polynomial interpolation of $\mathbf{f}(t, \mathbf{y})$ in (3) from previous instants (must be equally spaced) up to t_{n+1} , resulting in an implicit formula of the form

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta t(c_0 \mathbf{f}_{n+1} + c_1 \mathbf{f}_n + c_2 \mathbf{f}_{n-1} + \dots). \quad (4)$$

A first step estimates \mathbf{y}_{n+1} using a *predictor*, a less accurate explicit formula for evaluating the integral. This is done by polynomially extrapolating $\mathbf{f}(t, \mathbf{y})$ to t_{n+1} from previous instants up to t_n ,

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \Delta t(b_1 \mathbf{f}_n + b_2 \mathbf{f}_{n-1} + b_3 \mathbf{f}_{n-2} + \dots). \quad (5)$$

Next follows the *corrector*: using the predicted value of \mathbf{y}_{n+1} from (5) as an initial estimate, solve (4) iteratively for the corrected \mathbf{y}_{n+1} , increasing the order by one unit per iteration [4, §7.4] until the implicit formula's inherent order is achieved. No more precision is possible from further iteration.

Resolution of the implicit formula in the corrector step may be carried out by either functional iteration (right to left feedback in (4) until reaching self-consistency) or by the Newton-Raphson root finding method and the alike, which is preferable in stiff problems, where different time-scales are present in the same system.

As is clear from the predictor step, an additional method is needed for starting the algorithm. Explicit RK methods usually find their place here.

A common predictor-corrector (PC) example is the third order Adams-Bashforth-Moulton (ABM3) [5, §16.7] scheme,

$$\begin{aligned} \mathbf{y}_{n+1}^{\text{pred}} &= \mathbf{y}_n + \frac{1}{12}(23\mathbf{f}_n - 16\mathbf{f}_{n-1} + 5\mathbf{f}_{n-2})\Delta t + \mathcal{O}(\Delta t^4) \\ \mathbf{y}_{n+1}^{\text{corr}} &= \mathbf{y}_n + \frac{1}{12}(5\mathbf{f}_{n+1}^{\text{pred}} + 8\mathbf{f}_n - \mathbf{f}_{n-1})\Delta t + \mathcal{O}(\Delta t^4), \end{aligned} \quad (6)$$

or the fourth-order Milne's method (A-unstable [8], [3, §25.5], [4, §8.2.1])

$$\begin{aligned} \mathbf{y}_{n+1}^{\text{pred}} &= \mathbf{y}_{n-3} + \frac{4}{3}(2\mathbf{f}_n - \mathbf{f}_{n-1} + 2\mathbf{f}_{n-2})\Delta t + \mathcal{O}(\Delta t^5) \\ \mathbf{y}_{n+1}^{\text{corr}} &= \mathbf{y}_{n-1} + \frac{1}{3}(\mathbf{f}_{n+1}^{\text{pred}} + 4\mathbf{f}_n + \mathbf{f}_{n-1})\Delta t + \mathcal{O}(\Delta t^5). \end{aligned} \quad (7)$$

In contrast to RK methods, only one evaluation of $\mathbf{f}(t, \mathbf{y})$ is required per time-step at any order; here lies the PC efficiency.

PC methods with adaptative time-step are difficult to implement, but their efficiency is naturally boosted by their intrinsic variable order.

III. TEST AND COMPARISON

By choosing a system with known behaviour and analytical solution, different integration methods can be tested by fixing a desired precision and comparing the required computational work, or the other way around, by fixing a time-step and looking at the attained precision. Of particular interest is to know the convergence rate to the exact solution, a weight factor in the scheme choice when moving up in order. Convergence is computationally probed by looking at a sequence of decreasing time-step solution executions.

Beyond the equations of motion, a symmetry in the Lagrangian yields a conservation law. An acceptable numerical solution should respect this constraint, as well as any periodicity. This must also be part of the testing procedure, and can be used as a rectifier or controller at runtime.

We chose a rescaled one-dimensional simple harmonic oscillator (SHO) as our testing system, with Hamiltonian $H = \frac{1}{2}x^2 + \frac{1}{2}y^2$ and equations of motion

$$\dot{x} = y, \quad \dot{y} = -x, \quad \text{with i.c. } x(0) = 1, \quad y(0) = 0.$$

We proceeded by fixing a position accuracy $\varepsilon = 10^{-2}$, and an integration time $t = 300$ cycles (one cycle has period $T = 2\pi$), meaning every computed point up to this instant must fall inside the position accuracy interval, $|x_n - x(t_n)| < \varepsilon$. Then we sought a lower estimate of the maximum constant Δt respecting this accuracy, which we called "optimal time-step" Δt_{opt} . Once found, we calculated for position, energy and period variables, two quantities giving rough information about the method's efficiency. Accordingly, a measure of the overall deviation from the exact solution, and the increase rate (to first order) of the position error, which is its time-slope at $t = 0$. The overall deviation is defined as

$$\varepsilon_x \equiv \left[\frac{1}{N} \sum_{n=0}^N \left(\frac{x_n - x(t_n)}{x(t_n)} \right)^2 \right]^{1/2},$$

Table I: SHO numerical solution for 300 cycles and position accuracy of 10^{-2} using a constant time-step. CPU_time is the average over 10 execution samples.

method	$\Delta t_{\text{opt}} [10^{-3}]$	CPU_time[ms]	$\varepsilon_x [10^{-5}]$	$\varepsilon_E [10^{-5}]$	$\varepsilon_T [10^{-4}]$	$s_x [10^{-9}]$	$s_E [10^{-8}]$	$s_T [10^{-9}]$
RK3	5.00	66	1.90	1.13361	3.830	-5.2	-1.03990	0.0
ABM3	2.50	105	1.15	0.14168	1.926	0.7	0.12997	0.0
RK4	8.65	48	2.36	7×10^{-5}	6.687	0.0	-6×10^{-5}	0.0
ABM4	5.00	60	1.74	1.1×10^{-4}	3.830	0.0	1.0×10^{-4}	0.0
ABM3_4	5.00	85	1.73	5×10^{-5}	3.830	0.0	4×10^{-5}	0.0

$$\varepsilon_E \equiv \left[\frac{1}{N} \sum_{n=0}^N \left(\frac{E_n - E}{E} \right)^2 \right]^{1/2},$$

$$\varepsilon_T \equiv \left[\frac{1}{N_c} \sum_{n=0}^{N_c} \left(\frac{T_n - T}{T} \right)^2 \right]^{1/2},$$

with $N \equiv \lfloor t/\Delta t \rfloor$ the number of integration steps and $N_c \equiv \lfloor t/T \rfloor$ the number of cycles. $x(t) = \cos t$ is the exact position solution, $E = 1/2$ is the conserved total energy, and T_n the numerical period of each cycle, calculated as the elapsed time from sign change to sign change of x_n . Slopes s_x , s_E and s_T were estimated graphically (2 point linear fit). For s_x and s_E , a plot of the cycle-averaged relative error was used so as to factor out in-cycle faster fluctuations and visualizing linear behaviour; see Fig. 1.

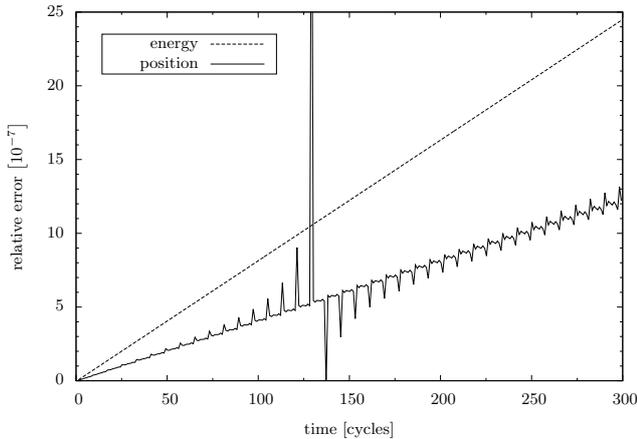


Figure 1: ABM3 cycle-averaged position and energy relative error for optimal time-step.

The code for carrying out these computations was developed in Fortran03 using an object-oriented programming paradigm, allowing high flexibility and enough generic style for this work's purpose. Organization was set in modules, providing a working environment with separated elements to describe the physical system, integrate it and analyse the solution.

Results for studied integration schemes are summarized in Table I. These solutions were computed using RAM dynamically allocated arrays from first to last point

of the trajectory, thus lowest order methods LPA and STA, listed in Section II, would not be tested successfully, demanding excessive memory (too tiny a time-step) to meet accuracy requirements. Also, (7) proved to be unstable in this problem.

We used a procedure given by [6] as a GNU Octave (or Matlab) script to generate the and ABM4 (fourth order) predictor and corrector formulae coefficients,

$$b_1 = 55/24, b_2 = -59/24, b_3 = 37/24, b_4 = -3/8,$$

$$c_0 = 3/8, c_1 = 19/24, c_2 = -5/24, c_3 = 1/24.$$

ABM3_4 uses ABM3's third order predictor and ABM4's fourth order corrector. We used a corresponding order RK method as a starter for ABMs. ABM functional iteration self-consistency criterion was set to consecutive changes smaller than the method's order plus one, which resulted in at most a single iteration for ABM3 and ABM4 (same order predictor and corrector), and at most two iterations for ABM3_4 (one order lower predictor).

According to Table I, RK4 exhibited the largest Δt_{opt} . Almost a factor 2 below are RK3, ABM4 and ABM3_4, and another factor 2 below is ABM3. This is no surprise: RK evaluates the function at several point and time values per step, against a single evaluation of ABM, so the latter will probably need a smaller Δt at the same order.

As for execution time, ABM3 performed the worst, and again RK4 was here the best. Above RK4 are RK3 and ABM4, about the same CPU_time. Note that ABM3_4 is far above ABM4, since more than one iteration were done at some steps.

Looking at position deviation, it seems RK4 now pays the price for such good numbers in Δt_{opt} and CPU_time, and scores the worst at ε_x , even doubling that of ABM3, that now gets on top. ABM4 and ABM3_4 are in between these two, with practically equal ε_x and below RK3. In spite of all methods having the same order of magnitude of ε_x , position precision is proven to be slightly better in fourth order methods, presenting all null s_x . RK3 is then the clear loser as for position precision, having a larger $|s_x|$ than ABM3; still, though, both have very small position error slopes of magnitude 10^{-9} .

Methods' numbers distribution in the period deviation is similar to that of ε_x , with RK4's ε_T being much higher and ABM3's much lower than the others', just that RK3, ABM4 and ABM3_4 all have equal ε_T . The difference lies in that the period deviation tendency s_T is null- to a fair

extent- for all studied methods. Along with $\varepsilon_T \sim 10^{-4}$ for all of them, we conclude they reproduce the SHO periodicity very well up to the chosen accuracy.

Differences between methods in measured quantities discussed so far ranged inside the same order of magnitude, thus we found energy to be the benchmark of performance in this test: at least four orders of magnitude separate third order from fourth order methods, both in ε_E as in s_E , so that energy conservation is highly enhanced in going from the former to the latter. Both in ε_E and s_E , RK3 is one order of magnitude on top of ABM3, showing poorer energy conservation. Among fourth order ones, we see RK4 and ABM3.4 exhibited pretty similar numbers in ε_E and s_E , being ABM3.4 superior, while ABM4 lied an order of magnitude above, proving the worst of the three. This energy comparison leaves ABM3.4 as clearly superior to ABM4.

As in Fig. 1 for ABM3, the position error profile exhibited fluctuations about the s_x -line for all studied methods, while the energy error did not, the reason being an opposite sign compensation in the momentum fluctuation.

Accounting for the different performance measures, ABM3.4 is probably the best choice for this test regarding solution precision. If, however, a small fraction of precision is willing to be sacrificed, then RK4 offers equally good results in much shorter execution times.

There is evidence [4, ch. 12], for certain problems, that RK methods have higher ratios of accuracy over total number of function evaluations, for low accuracies, than Adams schemes, and also have faster execution times per function evaluation. Our SHO test accuracy was low, and the function evaluation trivial; this would explain the two RK observed fast execution times.

As an overview, no doubt fourth order methods outperformed third order ones, but it is still interesting ABM3 was arguably the best in position solution and period conservation, thus leaving RK3 at the bottom of global precision. Nonetheless, ABM3 shows poor execution times.

IV. APPLICATION TO A NONLINEAR SYSTEM

One of the simplest systems exhibiting chaotic vibrations from a periodic source is that modelled by the Duffing-Holmes equation

$$\ddot{x} + \delta\dot{x} - \frac{1}{2}x(1 - x^2) = \lambda \cos \omega t, \quad (8)$$

representing a forced damped oscillation in a double-well symmetric potential, with stable minima $x_{\pm} = \pm 1$. Real systems showing such behavior include plasma oscillations, solid point defects, or the originally studied periodically forced nonlinearly elastic beam by Holmes. δ is the linear damping coefficient, the dissipative piece, λ the forcing amplitude and ω the driving force frequency.

We are interested to know which regions of $(\delta, \lambda, \omega)$ parameters admit stable periodic solutions (interspike in-

terval equal to the driving force period) and which ones present chaos. Following [7, Appendix B.6], we solved the autonomous system

$$\dot{x} = y, \quad \dot{y} = -\delta y + \frac{1}{2}x(1 - x^2) + \lambda \cos z, \quad \dot{z} = \omega,$$

equivalent to (8), taking the form (2) in 3-dimensional phase space, using RK4 with $\Delta t = 5 \times 10^{-3}$. The probed parameter region was $(\delta, \omega) = (0.15, 0.8)$ and $0.1 \leq \lambda \leq 0.3$. To perceive the nature of the solution, we used a bifurcation diagram of the motion's x spikes for varying λ parameter: for each λ we computed a long time trajectory and plotted a fixed number of consecutive x maxima. Using this measure of the motion, single spike periodic motion appears as a single point for that λ , for example, and chaotic motion as many spread points. Fig. 3 shows a representation of 100 consecutive spike values for 2000 uniformly distributed λ values after a fixed 50 force periods "stabilization" interval. The actual stabilization time was optimized by using the last computed point of the previous solution to start the present one, with the origin as the first trajectory initial condition. Only trajectories in nonchaotic regions get to stabilize, clearly.

From a big scope sight, the two main branches of Fig. 3 mean the system repeatedly jumps from vibrational states in one well to the other. Throughout the explored region, large and small periodic windows (branches) are observed in between chaotic regions (unstructured). Fig. 2 shows qualitatively different trajec-

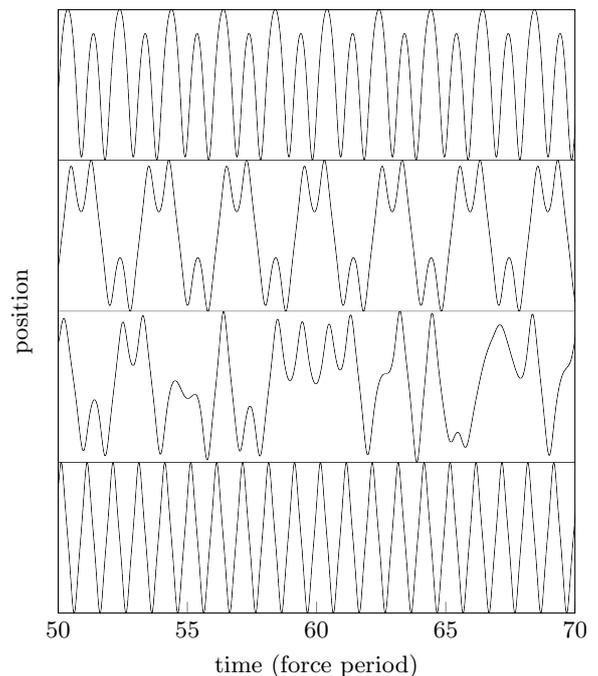


Figure 2: Long time solutions of the Duffing-Holmes oscillator for fixed $\delta = 0.15$ and $\omega = 0.8$. From top to bottom, trajectories for driving force amplitude $\lambda = 0.1, 0.2, 0.22$ and 0.3 . Position scales differ for each plot; refer to Fig. 3.

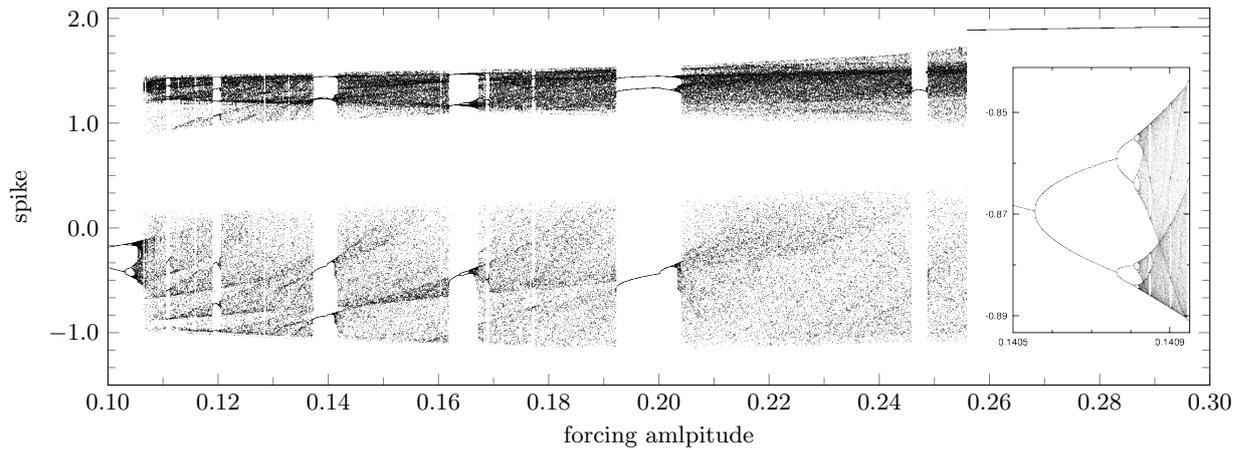


Figure 3: Duffing-Holmes bifurcation diagram for the motion's spikes against the forcing amplitude parameter. Resolution is of 100 spikes times 2000 amplitude values, for both the original and the zoomed-in region plots. The latter starts from a first bifurcation coming from a chaotic window.

tories for different λ in this region: Starting at $\lambda = 0.1$, the motion is 2-periodic around the x_- well; at $\lambda = 0.2$ it is 3-periodic, with two maxima at the x_+ well and two minima at the x_- well; it is chaotic at $\lambda = 0.22$, jumping with no apparent pattern between the two wells (strongly suggested by Fig. 3 and not by Fig. 2 alone). Finally, the behavior is restored to a period-1 oscillation around the x_+ well outside the intermitent chaotic-to-periodic middle region.

In the zoomed-in region of Fig. 3, it is observed the system follows a periodic-doubling route to chaos. From the third bifurcation, another magnification with the same resolution was carried out (enclosed λ and x -spike region, exponentially long computation) to lead a ratio $(\lambda_5 - \lambda_4)/(\lambda_6 - \lambda_5) \simeq 4.2$ between consecutive bifurcation intervals, at least similar to the asymptotic convergence prediction of 4.6692016 by Feigenbaum [7, ch. 5.3].

V. CONCLUSIONS

The computational analysis presented for the various integration methods should provide a clear distinction for choice among low to medium order methods in nonstiff

problems. We found RK4 to be the most suited among them as a solid first approach to a little known system, with short execution times compared to achieved precision.

Our study of the Duffing-Holmes system led to the observation of chaotic regions embedded with smaller periodic regions, connected by a renormalizable period-doubling structure. Two iterative fine-grained magnifications of the original force amplitude parameter region of the bifurcation diagram finally allowed us to check the Feigenbaum scaling law up to the leading term.

As exemplified in this solved case, numerical methods for the integration of dynamical systems constitute a basic and powerful tool, allowing to transcend inherent analytical difficulties and accessing a system's behavior for its understanding: only through numerical computation have we been able to do so and proceed to its analysis.

Acknowledgments

Grateful thanks to Jesús, my advisor, for his guidance and involvement; he provided me insight and helped me revise and improve every piece of this work.

-
- [1] R. W. Stanley, *American Journal of Physics* **52**, 499 (1984).
 - [2] A. Iserles, *A first course in the numerical analysis of differential equations* (Cambridge university press, 2009).
 - [3] M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables* (National Bureau of Standards Applied Mathematics Series, 1964).
 - [4] W. C. Gear, *Numerical Initial Value Problems in Ordinary Differential Equations* (Prentice-Hall, New Jersey, 1971), 1st ed.
 - [5] W. Press et al., *Numerical Recipes in Fortran 77: The Art of Scientific Computing* (Cambridge University Press, 1992), 2nd ed.
 - [6] B. Seidu, *International Journal of Computational and Applied Mathematics* **6**, 215 (2011).
 - [7] F. C. Moon, *Chaotic Vibrations: An Introduction for Applied Scientists and Engineers* (Wiley-Interscience, New York, 2004), 1st ed.
 - [8] Absolute stability is defined as a perturbation in a single y_n of the solution of $y' = \lambda y$ producing a change in subsequent y_m s not increasing from step to step.