

Article

Estimating the Highest Time-Step in Numerical Methods to Enhance the Optimization of Chaotic Oscillators

Martín Alejandro Valencia-Ponce ^{1,†} , Esteban Tlelo-Cuautle ^{1,*,†}  and Luis Gerardo de la Fraga ^{2,†} ¹ Department of Electronics, INAOE, Tonantzintla, Puebla 72840, Mexico; alejandroponce@inaoe.mx² Computer Science Department, CINVESTAV, Av. IPN 2508, Mexico City 07360, Mexico; fraga@cs.cinvestav.mx

* Correspondence: etlelo@inaoe.mx; Tel.: +52-222-2470-517

† These authors contributed equally to this work.

Abstract: The execution time that takes to perform numerical simulation of a chaotic oscillator mainly depends on the time-step h . This paper shows that the optimization of chaotic oscillators can be enhanced by estimating the highest h in either one-step or multi-step methods. Four chaotic oscillators are used as a case study, and the optimization of their Kaplan-Yorke dimension (D_{KY}) is performed by applying three metaheuristics, namely: particle swarm optimization (PSO), many optimizing liaison (MOL), and differential evolution (DE) algorithms. Three representative one-step and three multi-step methods are used to solve the four chaotic oscillators, for which the estimation of the highest h is obtained from their stability analysis. The optimization results show the effectiveness of using a high h value for the six numerical methods in reducing execution time while maximizing the positive Lyapunov exponent ($LE+$) and D_{KY} of the chaotic oscillators by applying PSO, MOL, and DE algorithms.



Citation: Valencia-Ponce, M.A.; Tlelo-Cuautle, E.; de la Fraga, L.G. Estimating the Highest Time-Step in Numerical Methods to Enhance the Optimization of Chaotic Oscillators. *Mathematics* **2021**, *9*, 1938. <https://doi.org/10.3390/math9161938>

Academic Editors: Árpád Bűrmen and Tadej Tuma

Received: 9 July 2021

Accepted: 9 August 2021

Published: 13 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: chaotic oscillator; time-step; one-step method; multi-step method; particle swarm optimization (PSO); many optimizing liaison (MOL); differential evolution (DE); Kaplan-Yorke dimension; Lyapunov exponent

1. Introduction

Chaotic oscillators are dynamical systems modeled by nonlinear ordinary differential equations (ODEs), in the form of initial value problems: $\dot{x} = f(x)$, solved by the initial condition $x(t_0) = x_0$. As already known, the main characteristic of a chaotic system relies on its high sensitivity to initial conditions, meaning that a very small variation causes significant differences in the system's response, thus causing random and disorderly behavior [1]. The term “chaos” is very often associated with disorder and unpredictability, so that chaotic oscillators are good candidates to develop cryptographic applications [2,3], design random number generators [4,5], and secure communication systems [6–8], among others.

The biggest challenge when solving nonlinear ODEs that model chaotic behavior relies on choosing both an appropriate numerical method and the selection of the highest step-size h that reduces execution time. This matters when optimizing characteristics of chaotic oscillators because a small h increases the execution time exponentially. On the one hand, choosing the highest h is not a trivial task; in fact, one must analyze the stability of the numerical method being applied to guarantee convergence to the solution. On the other hand, among the available one-step and multi-step numerical methods, if they are explicit or implicit, one must also choose the method executing the lowest execution time to increase the throughput and operating frequency in a chaotic system. For instance, in linear dynamical systems, h can be directly determined from the evaluation of the eigenvalues, which are related to the natural frequencies of the system. However, in non-linear dynamic systems, the problem is more complex, and one must analyze both the eigenvalues and the stability of the numerical method.

Recently, it has been demonstrated that both stochastic nature-inspired metaheuristics and deterministic global optimization methods are competitive and surpass one another in dependence on the available budget of function evaluations [9,10]. For instance, the authors in [11] applied a DIRECT-type technique [12] to a black-box global optimization problem with expensive function evaluations, which is challenging for numerical methods due to the practical limits on computational budgets often required by intelligent systems. Both DIRECT-type techniques and metaheuristics can be applied to optimize characteristics of chaotic oscillators, such as maximizing the positive Lyapunov exponent ($LE+$) or Kaplan-Yorke dimension (D_{KY}), where the challenge is solving the mathematical model multiple times until a minimum error is accomplished or a desired number of generations is executed. Therefore, the number of calls to the model and the time required to solve it are the main issues. In this manner, to reduce the execution time, we propose estimating the maximum h of a numerical method. The case study includes the application of three representative one-step and three multi-step numerical methods in order to solve four well-known chaotic oscillators, whose $LE+$ and D_{KY} are maximized by applying three metaheuristics, namely, the particle swarm optimization (PSO), many optimizing liaison (MOL), and differential evolution (DE) algorithm. These metaheuristics were chosen because they have been successfully applied to optimize chaotic oscillators, such as in [13], where PSO is applied to solve the parameter identification problem of a fractional-order discrete chaotic system. Additionally, in [14], PSO is applied to chaotic systems formulating the problem as a multi-dimensional one, and in [15], PSO is applied again to optimize the parameters of the Rössler chaotic system. More recently, PSO has been used to explain a bifurcation parameter detection strategy, as shown in [16], and MOL and DE algorithms have also been applied to optimize integer/fractional-order chaotic systems in [17]. In this manner, PSO, MOL, and DE algorithms are applied herein using the same population and generations, in which $LE+$ and D_{KY} are evaluated by the software time-series analysis (TISEAN) [18].

The rest of the manuscript is organized as follows: Section 2 shows four well-known chaotic oscillators taken as a case study, namely: Lorenz, Rössler, Lü, and an autonomous chaotic system introduced in [19]. Section 3 describes the six numerical methods applied to solve the four chaotic systems. Section 4 details the stability region of the six numerical methods and the computation of eigenvalues to estimate the highest h . Section 5 describes PSO, MOL, and DE algorithms. Section 6 describes the maximization of $LE+$ and D_{KY} by applying PSO, MOL, and DE to the four chaotic oscillators. A discussion on the optimization results is given in Section 7. Finally, the conclusions are summarized in Section 8.

2. Chaotic Oscillators

Among the huge plethora of chaotic/hyperchaotic oscillators, which are classified according to their dynamical characteristics as self-excited and hidden attractors, this section shows four well-known models. The most well-known chaotic oscillator is the one introduced by Lorenz, whose mathematical model is given in (1) [20]. In this oscillator, chaotic behavior is observed by setting $\sigma = 10$, $\rho = 28$, and $\beta = 8/3$, with initial conditions $x_0 = y_0 = z_0 = 0.1$. Figure 1 shows the phase-space portraits of the Lorenz chaotic system.

$$\begin{aligned}\dot{x} &= \sigma(y - x) \\ \dot{y} &= x(\rho - z) - y \\ \dot{z} &= xy - \beta z\end{aligned}\tag{1}$$

The second oscillator considered herein is the Rössler one, whose mathematical model is given in (2) [21]. The chaotic behavior is observed by setting $a = b = 0.2$ and $c = 5.7$, with initial conditions $x_0 = y_0 = z_0 = 1$. The phase-space portraits among its state variables are shown in Figure 2.

$$\begin{aligned}\dot{x} &= -y - z \\ \dot{y} &= x + (ay) \\ \dot{z} &= b + z(x - c)\end{aligned}\tag{2}$$

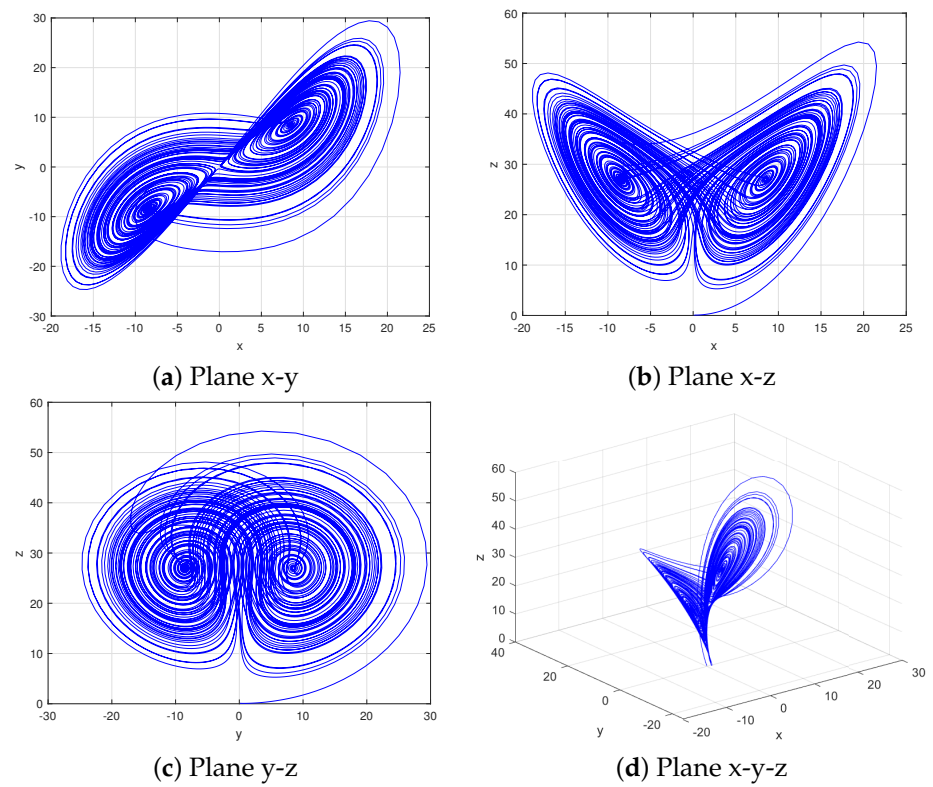


Figure 1. Phase-space portraits of the Lorenz chaotic system given in (1).

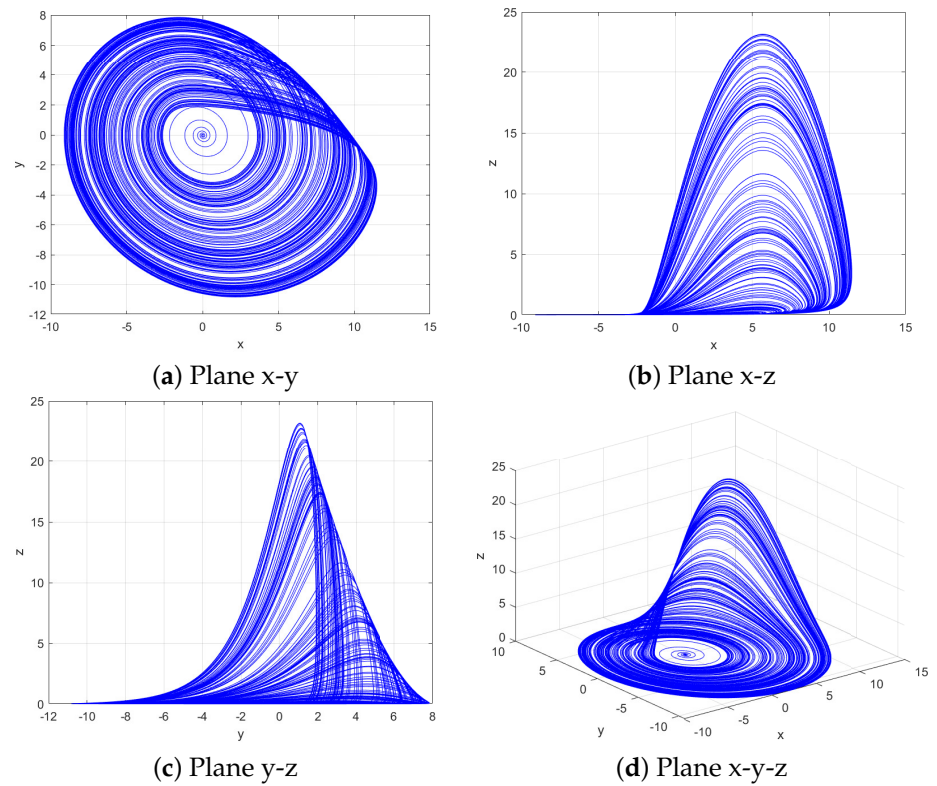


Figure 2. Phase-space portraits of the Rössler chaotic system given in (2).

The third case study is the Lü chaotic system given in (3) [22]. In this oscillator, the nonlinear function is approximated by a piecewise-linear (PWL) one, which is known as a saturated non-linear function (SNLF), as described in (4). This chaotic system has the advantage of adding more PWL functions to generate multi-directional and multi-scroll chaotic attractors [17].

$$\begin{aligned}\dot{x} &= y \\ \dot{y} &= z \\ \dot{z} &= -ax - by - cz + d_1 f(x)\end{aligned}\quad (3)$$

$$f(x) = \begin{cases} -k & \text{if } x < bp \\ mx & \text{if } -bp \leq x \leq bp \\ k & \text{if } x > bp \end{cases}\quad (4)$$

The Lü chaotic system generates chaotic behavior when the coefficients are set to $a = b = c = d_1 = 0.7$, and using initial conditions $x_0 = y_0 = z_0 = 0.1$. In this paper, the PWL functions has a saturation level of $k = 5$ and break points of $bp = 1$, so that the slope of the SNLF can be evaluated as $m = k/bp$. The phase-space portraits are shown in Figure 3.

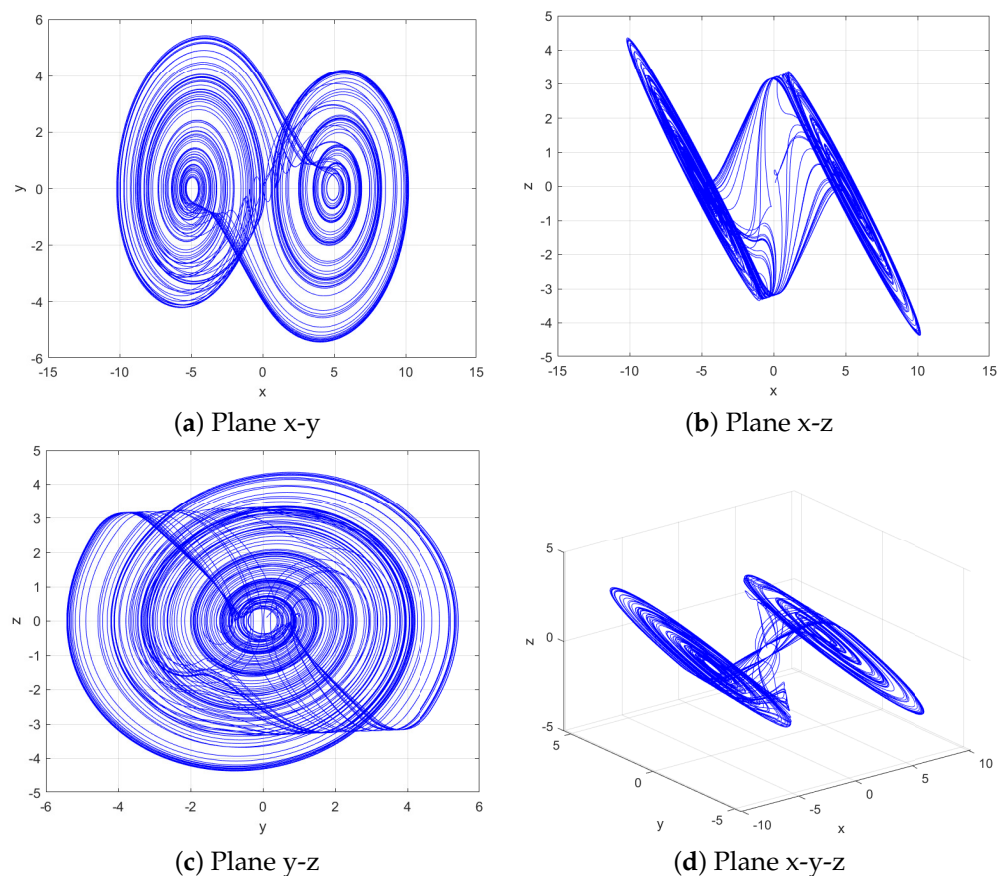


Figure 3. Phase-space portraits of the Lü chaotic system given in (3).

The fourth chaotic system is the autonomous chaotic oscillator described in (5) and introduced in [19]. This oscillator has a quadratic term x^2 , and the chaotic behavior arises by setting $a = 1$, $b = 1.1$, and $c = 0.42$, and using initial conditions $x_0 = 0.1$ and $y_0 = z_0 = 0$. The phase-space portraits are shown in Figure 4.

$$\begin{aligned}\dot{x} &= y \\ \dot{y} &= z \\ \dot{z} &= -ax - by - cz - x^2\end{aligned}\quad (5)$$

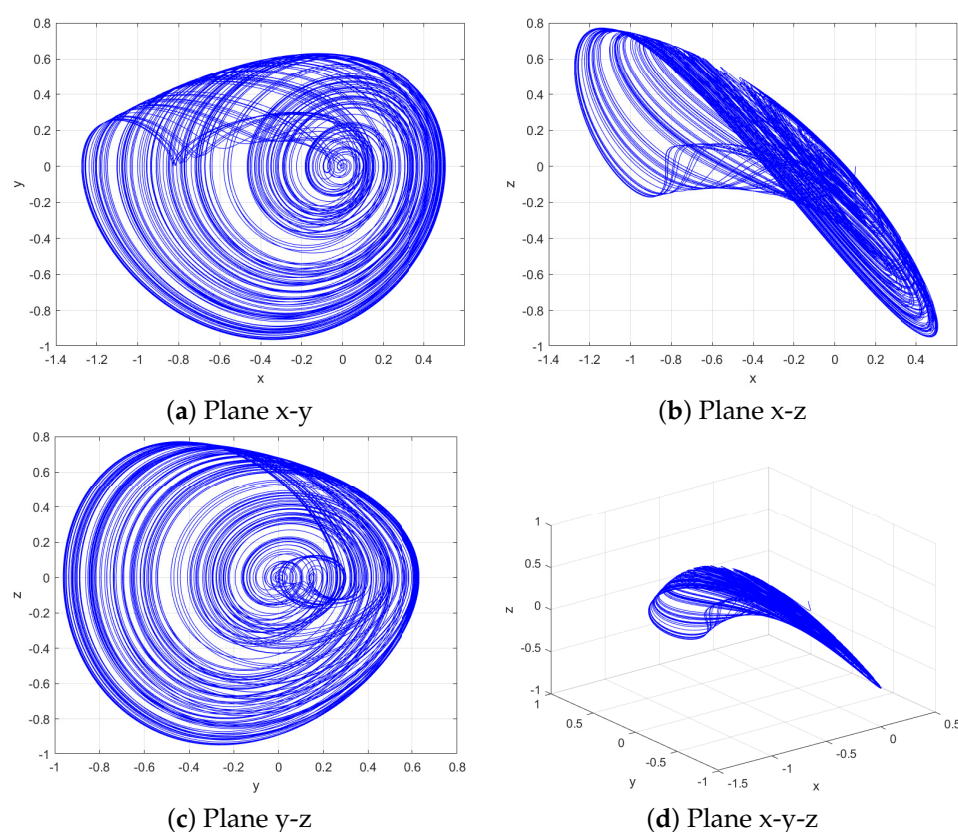


Figure 4. Phase-space portraits of the autonomous chaotic oscillator given in (5).

3. Numerical Methods

Solving an initial value problem requires the use of numerical methods, which in general can be of a one-step or multi-step type [23]. In both cases, the mathematical model is solved by choosing an appropriate h with the aim of reducing execution time. Besides, as shown in the following section, h is related to the stability of the numerical method [24].

As mentioned in [25], the main objective of a numerical method is to provide an acceptable approximation of the behavior of a dynamical system in continuous-time. However, as there exist a huge number of methods, this section shows the application of three representative one-step and three multi-step methods. Some of them are explicit, and others implicit. In the former case, the method requires only past values at each iteration n to update the current value at $n + 1$, while the implicit method updates the value $n + 1$ at the same iteration $n + 1$, so that they require an explicit method to estimate f_{n+i} .

The simplest one-step explicit method is known as the Forward Euler (FE), named in honor of Leonhard Euler. The iterative equation is given in Table 1. On the other hand, the simplest one-step implicit method is known as Backward Euler (BE), whose iterative equation is given in Table 1, and where the requirement of an additional calculation to approximate the problem solution at iteration $n + 1$ can be clearly appreciated. Therefore, the Forward Euler method can be applied first to evaluate $f(x_{n+1}, t_{n+1})$, and afterwards, one can obtain x_{n+1} . Among the one-step methods, the Runge-Kutta family is quite important, and is very often used to simulate chaotic oscillators. The fourth-order Runge-Kutta (RK4) method is given in Table 1, and it was developed around 1900 by Carl David Tolm Runge and Martin Wilhelm Kutta [26]. RK4 is the most widely used due to its high accuracy, but when it is implemented on digital hardware, it requires more resources than FE and BE methods, as already shown in [23].

The most well-known explicit multi-step methods are called Adams-Bashforth, and the implicit multi-step ones are called Adams-Moulton algorithms. Similar to the one-step methods, an implicit multi-step one finds the solution of an initial value problem using an explicit method from the Adams-Bashforth family to estimate $f(x_{n+1})$. Another type

of multi-step method is focused on solving stiff problems, and are known as Gears algorithms [24]. Table 2 shows three multi-step methods that are used herein to solve the four chaotic oscillators given in the previous section.

Table 1. One-step methods.

Method	Equation
Forward Euler (FE)	$x_{n+1} = x_n + hf(x_n, t_n)$
Backward Euler (BE)	$x_{n+1} = x_n + hf(x_{n+1}, t_{n+1})$
Runge-Kutta 4 (RK4)	$x_{n+1} = x_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$
	$k_1 = f(x_n, t_n)$
	$k_2 = f(x_n + \frac{h}{2}, t_n + \frac{hk_1}{2})$
	$k_3 = f(x_n + \frac{h}{2}, t_n + \frac{hk_2}{2})$
	$k_4 = f(x_n + h, t_n + k_3h)$

Table 2. Multi-step methods.

Method	Equation
Adams-Bashforth 6	$x_{n+1} = x_n + \frac{h}{1440}(4277f(x_n, t_n) - 7923f(x_{n-1}, t_{n-1}) + 9982f(x_{n-2}, t_{n-2}) - 7298f(x_{n-3}, t_{n-3}) + 2877f(x_{n-4}, t_{n-4}) - f(x_{n-5}, t_{n-5}))$
Adams-Moulton 6	$x_{n+1} = x_n + \frac{h}{1440}(475f(x_{n+1}) + 1427f(x_n) - 798f(x_{n-1}) + 482f(x_{n-2}) - 173f(x_{n-3}) + 27f(x_{n-4}))$
Gear 2 (G2)	$x_{n+1} = \frac{4}{3}x_n - \frac{1}{3}x_{n-1} + h(\frac{2}{3}f(x_{n+1}, t_{n+1}))$

4. Stability Regions and Estimation of h

The stability region of a numerical method can be obtained by solving a first-order linear equation of the form $y' = f(x, y)$, where $f(x, y) = -\lambda$, for which λ is an eigenvalue. From the stability analysis, a numerical method is said to be numerically stable if the numerical error is not amplified, but it decreases with the evolution of the iterations. This behavior can be identified in the one-step methods known as Backward Euler and Trapezoidal. The numerical methods that do not have this property are said to be numerically unstable, and this behavior can be observed in Forward Euler in specific cases.

Let us solve the initial value problem of the form $\dot{x} = -\lambda x$, applying Forward Euler, Backward Euler, and Trapezoidal methods. Applying Forward Euler, one gets the iterative formulae given in (6), and applying Backward Euler, one gets (7), and applying the Trapezoidal method, one gets (8).

$$x_{n+1} = (1 - h\lambda)x_n, \quad (6)$$

$$x_{n+1} = \frac{x_n}{1 + h\lambda}, \quad (7)$$

$$x_{n+1} = \frac{1 - \frac{h\lambda}{2}}{1 + \frac{h\lambda}{2}}x_n \quad (8)$$

To show that Forward Euler is numerically unstable, the iterations beginning with the initial condition x_0 become:

$$x_1 = (1 - \lambda h)x_0$$

$$\begin{aligned}
 x_2 &= (1 - \lambda h)x_1 = (1 - \lambda h)^2 x_0 \\
 &\vdots \\
 x_n &= (1 - \lambda h)^n x_0
 \end{aligned}$$

Therefore, if $(1 - \lambda h) > 1$, then $x_n \rightarrow \infty$ as $n \rightarrow \infty$; consequently, for the Forward Euler to be numerically stable, it must be necessary that $(1 - \lambda h) < 1$ or $0 < \lambda h < 2$. In this case, if λ is a real and positive number, then $h < 2/\lambda$.

The Backward Euler and Trapezoidal methods are numerically stable, because from (7) and (8), one can observe that $x_n \rightarrow 0$ as $n \rightarrow \infty$, and this ideally occurs regardless of h . The stability regions of FE, BE, and RK4 are shown in Figure 5. A similar analysis is performed for the multi-step methods, so that the stability regions of Adams-Bashforth 6, Adams-Moulton 6, and Gear 2 are shown in Figure 6. Looking at these figures, one can see that if the eigenvalues of a function increase, then h must decrease, and vice versa.

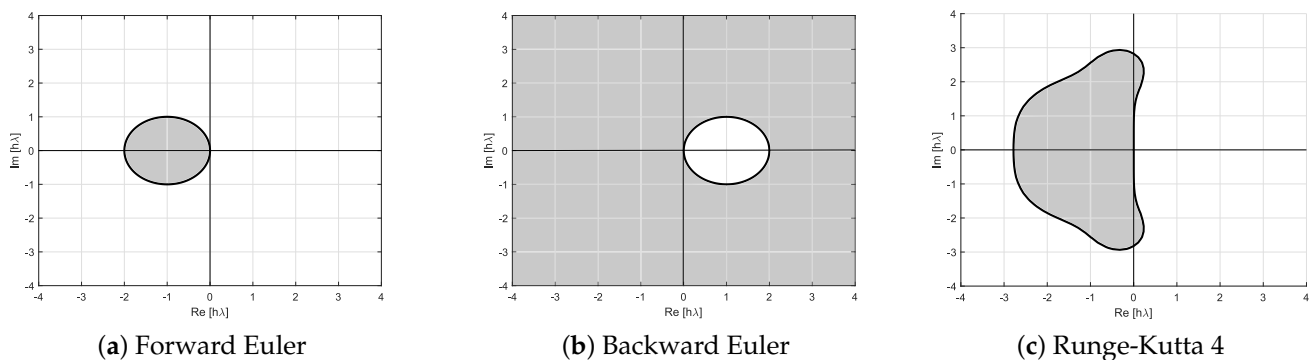


Figure 5. Stability regions of one-step methods listed in Table 1.

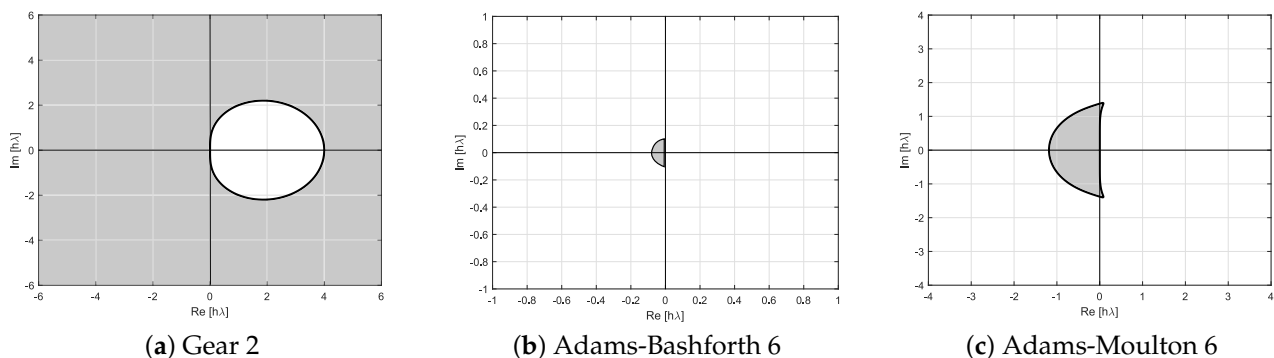


Figure 6. Stability regions of multi-step methods listed in Table 2.

Analyzing the stability regions of the numerical methods helps to verify that the eigenvalues are inversely related to the value of h . In this manner, one must compute the eigenvalues (λ) of a dynamical system to estimate an h that guarantees stability. Taking the Lorenz system given in (1) as an example, it has three equilibrium points: $EP_1 = (8.4852, 8.4852, 27)$, $EP_2 = (-8.4852, -8.4852, 27)$ and $EP_3 = (0, 0, 0)$. The Jacobian matrix is given in (9), which must be evaluated at the three equilibrium points $EP^* = (x^*, y^*, z^*)$ to obtain the eigenvalues listed in (10). Table 3 shows the Jacobians, equilibrium points, and eigenvalues of the chaotic oscillators described in Section 2.

$$J_{(x^*, y^*, z^*)} = \begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z^* & -1 & -x^* \\ y^* & x^* & -\beta \end{bmatrix} \quad (9)$$

$$\begin{aligned}
 EP_1 : \lambda_{(1,2,3)} &= (-13.8545, 0.0939 \pm -j10.1945) \\
 EP_2 : \lambda_{(1,2,3)} &= (-13.8545, 0.0939 \pm -j10.1945) \\
 EP_3 : \lambda_{(1,2,3)} &= (-2.6667, -22.8277, 11.8277)
 \end{aligned} \tag{10}$$

Table 3. Jacobian, Equilibrium points, and Eigenvalues of the chaotic systems given in (1)–(5).

System	Jacobian	Equilibrium Points	Eigenvalues
Lorenz	$\begin{bmatrix} -\sigma & \sigma & 0 \\ \rho - z^* & -1 & -x^* \\ y^* & x^* & -\beta \end{bmatrix}$	$(-8.4852, -8.4852, 27)$ $(8.4852, 8.4852, 27)$ $(0, 0, 0)$	$-13.8545, 0.0939 \pm -j10.1945$ $-13.8545, 0.0939 \pm -j10.1945$ $-2.6667, -22.8277, 11.8277$
Rössler	$\begin{bmatrix} 0 & -1 & -1 \\ 1 & a & 0 \\ z^* & 0 & x^* - c \end{bmatrix}$	$(0.0070, -0.0351, 0.0351)$ $(5.6929, -28.4648, 28.4648)$	$-5.6869, 0.0970 \pm -j0.9951$ $0.1929, 0.0000045 \pm -j5.4280$
Lü	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -a & -b & -c \end{bmatrix}$	$(0, 0, 0)$ $(1, 0, 0)$	$1.0743, -0.8872 \pm -j1.3488$ $-0.8480, 0.0740 \pm -j0.9055$
Autonomous Chaotic System	$\begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -2x^* - a & -b & -c \end{bmatrix}$	$(0, 0, 0)$ $(-1, 0, 0)$	$0.7451, 0.1625 \pm -j1.1471$ $0.5897, -0.5048 \pm -j1.2002$

After computing the eigenvalues, one can evaluate the stability conditions, so that FE's stability is guaranteed when $0 < h < 2/\lambda$, where λ is the largest eigenvalue of the system, that is, $\lambda = 22.8277$. Substituting λ in the inequality, one finds that FE is stable for $h < 0.0876$. However, in practice, this h value varies due to local and round-off errors of the methods. For instance, the local error is defined as the error that occurs at $t = t_{n+1}$, assuming that x_n is the initial condition; whereas a total error is defined as the current accumulated error from $t = 0$ to $t = t_{n+1}$, with initial condition x_0 . In [27], these errors are described as local and global, where the local error by truncation (LTE) is first calculated, and then some form of stability is used to show that the global error can be limited in LTE terms. The global error refers to the approximated solution minus the exact solution error, and LTE refers to the error produced by the finite difference derivative approximation, and is therefore something that can be easily estimated using Taylor series expansions. Obviously, these are not the unique errors in numerical methods: according to [28,29], there are various sources of errors, and some of them are errors in the input data, rounding errors during calculations, simplifications in mathematical models, and even human errors. Besides, the stability analysis helps to estimate the highest h from which one can test lower values after observing the desired chaotic behavior.

5. PSO, MOL, and DE Algorithms

5.1. Particle Swarm Optimization Algorithm

The particle swarm optimization (PSO) algorithm is a sub-field of computational intelligence belonging to the field of swarm and collective intelligence. Similar swarm algorithms are the Ant Colony Optimization [30] and Artificial Bee Colony [31]. PSO is based on a mathematical model developed by Kennedy and Eberhart in 1995 [32], which describes the social behavior of birds and/or fish through a model that is based on the basic principles of self-organization that can be used to describe complex systems, as for the chaotic oscillators. The population is based on particles forming a search group for the purpose of finding food, and generally, each individual continues his search according to his own experience and the experience of the whole group.

The main idea in PSO begins by initializing a set of particles in a search space, given a favorable initial position, assigning an initial velocity vector, and allowing the particles to change their position and velocity at each iteration according to some random parameters. The algorithm updates the position and velocity of the particles that follow the particle with the best result associated to their social behavior. Each particle remembers its best position

and recognizes if its current position is the best among the other particles, that is, the global best. Mathematically, the update equations are given in (11) to describe the velocity and (12) to describe the position, at iteration i th, respectively. $\text{rand}()$ is a function that returns some random real numeric values between 0 and 1; p_{best} is the best position of the particle and g_{best} represents the best global position among all the particles. c_1 and c_2 are two parameters that represent the confidence of the particle itself, named “cognition” and “swarm”.

$$v_i(t+1) = v_i(t) + c_1 \text{rand}() (p_{best}(t) - p_i(t)) + c_2 \text{rand}() (g_{best}(t) - p_i(t)) \quad (11)$$

$$p_i(t+1) = p_i(t) + v_i(t+1) \quad (12)$$

5.2. Many Optimizing Liaisons Algorithm

MOL is a simplified version of PSO proposed by Kenedy [33], but according to [34], after some research, it was named Many Optimizing Liaisons. Basically, MOL is based on eliminating the best-known position of the particle p_{best} in (11), which updates to (13):

$$v_i(t+1) = wv_i(t) + c_2 \text{rand}() (g_{best}(t) - p_i(t)). \quad (13)$$

As shown in [35], MOL is faster and shorter than PSO, so that the selection of parameters is simpler compared to PSO. In addition, MOL is a purely social algorithm tending to follow the best swarm's particle (g_{best}). The w in (13) is the inertia weight that maintains a balance between global and local search, so that the exploration process of MOL quickly finds an optimum solution with a lesser number of iterations. In this paper, w and c_2 are set to -0.31 and 2 , respectively.

5.3. Differential Evolution Algorithm

DE is a simple and effective evolutionary algorithm inspired by the theory of the evolution of species proposed by Darwin. DE is used to solve global optimization problems in a continuous domain. According to [36], DE works in two phases: initialization and evolution. In the first phase, the population is randomly generated, and afterwards, it goes through mutation, crossover, and selection processes, which are repeated until a stop criterion is met. During the initialization, the population is saved into a D -dimensional vector $\mathbf{x}_j^G = \{x_{1,j}^G, x_{2,j}^G, \dots, x_{D,j}^G\}$ for $j = \{1, 2, \dots, Np\}$, where Np is the population's size and G denotes the maximum number of generations. In the evolution phase, new individuals are generated by performing mutation (14), crossover (15), and selection (16) operations. In the mutation process, a mutant vector $\mathbf{V}_j^G$ is generated for each target vector $\mathbf{x}_j^G$ in generation G with the following form:

$$\mathbf{V}_j^G = \mathbf{x}_{r1}^G + F * (\mathbf{x}_{r2}^G - \mathbf{x}_{r3}^G), \quad (14)$$

where F is the scale factor with a value between 0 and 1; and $r1, r2, r3 \in \{1, 2, \dots, Np\}$ are three different random scalars. The crossover process is performed using the target vector, mutant vector, and a crossover probability Cr , whose value is set between 0 and 1 in order to generate a trial vector $\mathbf{U}_j^G = \{u_{1,j}^G, u_{2,j}^G, \dots, u_{D,j}^G\}$, which is generated as:

$$u_{i,j}^G = \begin{cases} v_{i,j}^G & \text{if } \text{rand}_j \leq Cr \\ x_{i,j}^G & \text{otherwise} \end{cases} \quad (15)$$

where $i \in \{1, 2, \dots, D\}$. Finally, in the selection process, the target and trial vectors are compared according to their fitness value, and the best of them survives for the next generation. The selection process is performed by (16).

$$\mathbf{x}_j^{G+1} = \begin{cases} \mathbf{U}_j^G & \text{if } f(\mathbf{U}_j^G) \leq f(\mathbf{x}_j^G) \\ \mathbf{x}_j^G & \text{otherwise} \end{cases} \quad (16)$$

6. Optimization of $LE+$ and Kaplan-Yorke Dimension

6.1. Lyapunov Exponents and Kaplan-Yorke dimension

The high sensitivity to initial conditions of a chaotic oscillator is appreciated by analyzing two orbits produced by two quite close and different initial values. The orbits separate exponentially over time, causing the orbits to separate exponentially too. This phenomenon is quantified by evaluating the positive Lyapunov exponent ($LE+$) [37], which describes local instability in a chaotic motion. The existence of an $LE+$ means that there exists a high probability that the system has chaotic behavior [38].

For a chaotic oscillator that has three ODEs, like the systems described in Section 2, it has three Lyapunov exponents, where one must be positive, one zero, and one negative. The ordering of the Lyapunov exponents (LE) makes it possible to evaluate the Kaplan-Yorke dimension through (17), where k is an integer such that the sum of (LE_i) is not negative. The $LE+$ and D_{KY} shown in Table 4 for the four chaotic oscillators were evaluated using TISEAN [18], whose chaotic time series were generated by applying the six numerical methods described in Tables 1 and 2, for the state variable x . One can also see the execution time of the numerical method (TimeNumMet) and the one taken by TISEAN:

$$D_{KY} = k + \frac{\sum_{i=1}^k LE_i}{LE_{k+1}} \quad (17)$$

6.2. Maximizing $LE+$ and D_{KY} by PSO, MOL and DE Using the Highest h

The optimization of chaotic oscillators takes a long execution time, mainly associated to the numerical method and TISEAN. In this paper, the six methods described in Section 3 are applied to the four chaotic oscillators presented in Section 2. As already shown in Section 4, the selection of an appropriate h depends on the method's stability region. In this manner, the highest h for each method and for each chaotic oscillator is given in Table 5. As one can see, both TISEAN and the numerical method take less execution time. In all cases, the highest h is allowed by Runge Kutta 4. However, RK4 requires more additional calculations compared to the Adams-Moulton method, which allows an acceptable high h value. In optimizing $LE+$ and D_{KY} , the method that better satisfies the trade-off between the number of calculations and the highest h is Gear of order 2, while the Backward Euler, in general, is the method that accepts the smallest h . Another issue is that, as h increases, D_{KY} decreases, so that we solve this challenge by optimizing D_{KY} by varying the design parameters of the four chaotic systems by applying the PSO, MOL, and DE algorithms. Thus, the pseudo-codes of PSO and DE are given in Algorithms 1 and 2, respectively. In all cases, the constraint is guaranteeing the highest h .

As one can observe from Tables 4 and 5, the longest execution time is required by TISEAN. Due to this limitation, some restrictions were established to determine that the system has unstable behavior. One of the characteristics that determines this instability is based on analyzing the eigenvalues of the chaotic oscillator. According to [39], if there are i and j such that $Re[\lambda_i] < 0$ and $Re[\lambda_j] > 0$, and if the system has two complex eigenvalues and one real, then the system is said to be unstable.

Table 4. $LE+$ and D_{KY} of (1)–(5) using a small h .

		One-Step Method			Multi-Step Method		
		FE	BE	RK4	G2	AB6	AM6
Lorenz	h	0.01	0.001	0.01	0.01	0.001	0.01
	$LE+$	0.0707	0.0604	0.0503	0.0501	0.0486	0.0460
	D_{KY}	3	3	3	3	3	3
	TimeNumMet(s)	1.16	16.57	3.06	2.69	45.38	4.36
	TimeTISEAN(s)	879.68	36.953	789.79	819.92	101.204	788.99
Rössler	h	0.01	0.01	0.01	0.01	0.01	0.01
	$LE+$	0.3112	0.2443	0.2405	0.2391	0.1900	0.2409
	D_{KY}	2.7786	2.7927	2.7588	2.7518	2.7362	2.7374
	TimeNumMet(s)	1.17	1.52	3.11	1.96	4.45	4.15
	TimeTISEAN(s)	626.17	700.56	658.64	640.1	648.62	646.63
Lü	h	0.01	0.01	0.01	0.01	0.01	0.01
	$LE+$	0.0800	0.0939	0.1062	0.0862	0.1020	0.0738
	D_{KY}	2.8915	2.7568	2.8744	2.9039	2.8879	2.9289
	TimeNumMet(s)	1.22	1.59	3.17	2.26	4.73	4.53
	TimeTISEAN(s)	596.02	638.57	600.47	614.01	585.01	597.74
Autonomous Chaotic System	h	0.001	0.01	0.01	0.01	0.01	0.01
	$LE+$	0.2688	0.0145	0.0090	0.0174	0.0122	0.0088
	D_{KY}	2.9047	2.8336	2.6598	2.7710	2.7374	2.6427
	TimeNumMet(s)	11.84	1.57	3.09	1.79	3.92	4.79
	TimeTISEAN(s)	26.405	392.06	366.67	360.04	362.61	367.65

Table 5. $LE+$ and D_{KY} of (1)–(5) using the highest h allowed by each numerical method.

		One-Step Method			Multi-Step Method		
		FE	BE	RK4	G2	AB6	AM6
Lorenz	h_{max}	0.0245	0.0020	0.1070	0.0600	0.0046	0.057
	$LE+$	0.0380	0.1232	0.0687	0.0886	0.0462	0.0727
	D_{KY}	2.1009	3.0640	2.0660	2.2256	3.7000	2.3617
	TimeTISEAN(s)	670.96	64,901.53	30.33	109.99	14,498.11	153.18
	TimeNumMet(s)	0.43	7.69	0.36	0.29	8.79	0.74
Rössler	h_{max}	0.0894	0.0800	0.2399	0.0900	0.0101	0.1300
	$LE+$	0.2346	0.2475	0.0204	0.0215	0.2133	0.0881
	D_{KY}	2.3909	2.6866	2.0439	2.0903	2.7676	2.3724
	TimeTISEAN(s)	138.15	109.95	28.34	95.59	3693.74	53.98
	TimeNumMet(s)	0.11	0.18	0.13	0.18	3.72	0.42
Lü	h_{max}	0.1300	0.0850	1.5000	0.9000	0.0770	0.8000
	$LE+$	0.0177	0.0181	0.1550	0.1621	0.0372	0.1518
	D_{KY}	2.1805	2.1839	2.0195	2.2013	2.0276	2.6127
	TimeTISEAN(s)	1675.19	3134.73	37.95	79.91	3909.75	96.65
	TimeNumMet(s)	0.83	1.97	0.23	0.21	5.55	0.61
Autonomous Chaotic System	h_{max}	0.0033	0.0260	1	0.1633	0.0981	0.1926
	$LE+$	0.1683	0.0028	0.0976	0.0104	0.0869	0.0152
	D_{KY}	2.9056	2.2115	2.2155	1.7764	1.9543	2.1544
	TimeTISEAN(s)	5385	507.13	6.9	32.31	63.41	16.8
	TimeNumMet(s)	2.95	0.88	0.05	0.16	0.62	0.24

Algorithm 1 PSO.

```

procedure PSO( $nPop, MaxIt$ )
  Generate a file including the numerical method and the chaotic oscillator (CO)
  for  $i = 1 : nPop$  do
    Randomly initialize the design variables
    Randomly replace the design variables into the file to evaluate  $LE+$  and  $D_{KY}$ 
    Simulate the  $CO_i$  in TISEAN
    Get  $D_{KY}$  of the output file according to TISEAN
    Update  $p_{best}$  and  $g_{best}$  particles checking the constraints
  end for
  for  $it = 1 : MaxIt$  do
    for  $i = 1 : nPop$  do
      Copy particle  $i$  to  $p$ 
      Update the particle  $p$  velocity according to (11)
      Update the particle  $p$  position (design variables) according to (12)
      Replace the new design variables into the file
      Simulate the  $CO_p$  in TISEAN
      Compare particles  $i$  and  $p$ 
      Update  $p_{best}$  and  $g_{best}$  particles checking the constraints.
    end for
  end for
end procedure

```

Algorithm 2 DE.

```

procedure DE( $G, Np, Cr, F, D$ )
  Generate a file including the numerical method and the chaotic oscillator (CO)
  for  $i = 1 : Np$  do
    Randomly initialize the design variables
    Randomly replace the design variables into the file to evaluate  $LE+$  and  $D_{KY}$ 
    Simulate the  $CO_i$  in TISEAN
    Get  $D_{KY}$  of the output file according to TISEAN
    Save the evaluation results for each individual
  end for
  for  $it = 1 : G$  do
    for  $i = 1 : Np$  do
      Randomly select three different individuals  $x_a, x_b$  and  $x_c$ .
      Generate the target vector according to (14)
      Perform the crossover process according to (15)
      Replace the new design variables into the file
      Simulate the CO in TISEAN
      Evaluate each individual according to (16)
      Update the population by selecting the individuals with the greatest fitness value
    end for
  end for
end procedure

```

7. Feasible Solutions Provided by PSO, MOL and DE

To perform the D_{KY} optimization by applying PSO, MOL and DE, a number of generations equal to 100 was set for each metaheuristic, with a population equal to the number of design parameters multiplied by ten. The search spaces for the design parameters of the four chaotic oscillators were set as follows: $0.01 < \sigma < 30$, $0.01 < \rho < 50$ and $0.01 < \beta < 10$ for Lorenz; $0.01 < a, b < 10$ and $0.01 < c < 30$ for Rössler; $0.01 < a, b, c, d_1 < 10$ for Lü, and $0.01 < a, b, c < 10$ for the Autonomous Chaotic System.

The optimization results are given in Table 6, where it can be appreciated that the execution time evaluating D_{KY} and the one taken by TISEAN are enhanced, while keeping the highest h allowed by the numerical method. In this manner, for the Lorenz and Lü

oscillators, the Gear of order 2 was used to perform the optimization, while for Rössler and the Autonomous Chaotic System, the Adams-Moulton of order 6 was applied. Analyzing the data in Table 5 and comparing them with those given in Figure 7, it can be seen that after the optimization process, the execution time for optimizing the Rössler and the Autonomous Chaotic System oscillators is significantly enhanced.

Table 6. Results performed by PSO, MOL and DE for the four chaotic oscillators. Time(s) includes the time of the numerical method and TISEAN.

Oscillator	PSO				MOL				DE			
	Design Parameters	LE+	D_{KY}	Time(s)	Design Parameters	LE+	D_{KY}	Time(s)	Design Parameters	LE+	D_{KY}	Time(s)
Lorenz	$\sigma = 2.1146$ $\rho = 14.7896$ $\beta = 0.2390$	0.0586	3.3108	66.45	$\sigma = 1.8941$ $\rho = 13.3546$ $\beta = 0.1357$	0.0622	3.4161	57.47	$\sigma = 2.1144$ $\rho = 12.8497$ $\beta = 0.2154$	0.0671	3.4515	69.63
Rössler	$a = 0.3233$ $b = 0.6000$ $c = 5.5711$	0.3191	2.81282	16.83	$a = 0.2000$ $b = 0.3984$ $c = 5.9857$	0.2359	2.6491	43.2	$a = 0.2615$ $b = 0.5772$ $c = 7.3303$	0.2931	2.8818	17.35
Lü	$a = 3.2570$ $b = 2.6660$ $c = 1.1400$ $d1 = 2.3578$	0.3179	5.5029	79.14	$a = 3.1472$ $b = 2.6865$ $c = 1.0686$ $d1 = 2.6899$	0.3181	5.8201	80.69	$a = 2.91$ $b = 2.3913$ $c = 1.0533$ $d1 = 3.1351$	0.3118	5.6715	83.53
Autonomous Chaotic System	$a = 1.7787$ $b = 1.8787$ $c = 0.3815$	0.0251	2.2543	21.78	$a = 1.5159$ $b = 1.6162$ $c = 0.3916$	0.0267	2.2567	21.82	$a = 1.3767$ $b = 1.4767$ $c = 0.3861$	0.0073	2.3703	19.94

Figure 7 shows the evolution of the D_{KY} for each chaotic oscillator applying the PSO, MOL and DE algorithms.

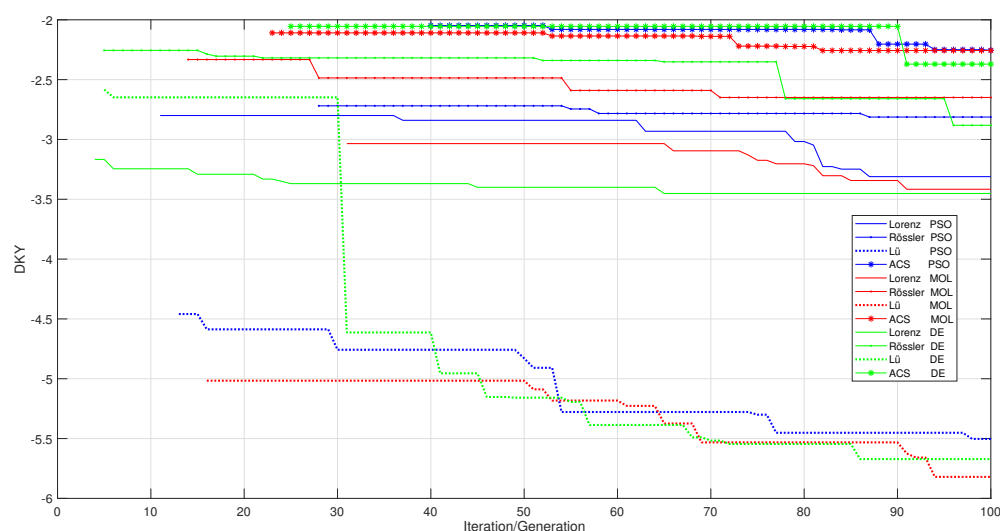


Figure 7. Best global evolution of the D_{KY} for the four chaotic oscillators applying PSO, MOL and DE.

8. Conclusions

Performing the simulation of a chaotic system is time-consuming, and reducing its execution time is very challenging because there are many considerations that must be accomplished in order to guarantee chaotic behavior. This paper showed that the execution time is enhanced by estimating the highest h for each numerical method, which has been performed by stability analysis of the methods. The case study included four well-known chaotic oscillators, three presentative one-step methods, and three multi-step methods. We demonstrated that one can maximize D_{KY} by applying metaheuristics, such as PSO, MOL and DE algorithms, while maintaining the highest h for the numerical method. As a result, the behavior of the three metaheuristics over 100 generations was shown in Figure 7, where the best global evolution of D_{KY} for the four chaotic oscillators applying PSO, MOL and DE is appreciated. Finally, the optimization results listed in Table 6 confirmed the suitability of applying metaheuristics in the optimization of chaotic systems, and the usefulness of estimating the highest h to enhance execution time.

Author Contributions: Investigation, M.A.V.-P. and E.T.-C.; Writing—review and editing, M.A.V.-P., E.T.-C. and L.G.d.l.F. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Wang, L.; Liu, J.; Lu, Z.R. Incremental response sensitivity approach for parameter identification of chaotic and hyperchaotic systems. *Nonlinear Dyn.* **2017**, *89*, 153–167. [\[CrossRef\]](#)
- Muhammad, A.S.; Özkaynak, F. SIEA: Secure Image Encryption Algorithm Based on Chaotic Systems Optimization Algorithms and PUFs. *Symmetry* **2021**, *13*, 824. [\[CrossRef\]](#)
- Sun, X.; Shao, Z.; Shang, Y.; Liang, M.; Yang, F. Multiple-image encryption based on cascaded gyrator transforms and high-dimensional chaotic system. *Multimed. Tools Appl.* **2021**, *80*, 15825–15848. [\[CrossRef\]](#)
- Stoller, S.; Campbell, K.A. Demonstration of Three True Random Number Generator Circuits Using Memristor Created Entropy and Commercial Off-the-Shelf Components. *Entropy* **2021**, *23*, 371. [\[CrossRef\]](#)
- Rezk, A.A.; Madian, A.H.; Radwan, A.G.; Soliman, A.M. Multiplierless chaotic pseudo random number generators. *AEU-Int. J. Electron. Commun.* **2020**, *113*, 152947. [\[CrossRef\]](#)
- Bendoukha, S.; Abdelmalek, S.; Ouannas, A. Secure communication systems based on the synchronization of chaotic systems. In *Mathematics Applied to Engineering, Modelling, and Social Issues*; Springer: Cham, Switzerland, 2019; pp. 281–311.
- Rusyn, V.; Mamat, M.; Azharul, F.; Sanjaya, W.M.; Sambas, A.; Dwipriyoko, E.; Sutoni, A. Computer Modelling of the Information Properties of Hyper Chaotic Lorenz System and Its Application in Secure Communication System. *J. Phys. Conf. Ser.* **2021**, *1764*, 012205. [\[CrossRef\]](#)
- Weng, H.; Zhang, C.; Chen, P.; Chen, R.; Xu, J.; Liao, Y.; Liang, Z.; Shen, D.; Zhou, L.; Ke, J. A quantum chaotic image cryptosystem and its application in IoT secure communication. *IEEE Access* **2021**, *9*, 20481–20492.
- Sergeyev, Y.D.; Kvasov, D.; Mukhametzhano, M. On the efficiency of nature-inspired metaheuristics in expensive global optimization with limited budget. *Sci. Rep.* **2018**, *8*, 1–9. [\[CrossRef\]](#)
- Cavoretto, R.; De Rossi, A.; Mukhametzhano, M.S.; Sergeyev, Y.D. On the search of the shape parameter in radial basis functions using univariate global optimization methods. *J. Glob. Optim.* **2021**, *79*, 305–327. [\[CrossRef\]](#)
- Paulavičius, R.; Sergeyev, Y.D.; Kvasov, D.E.; Žilinskas, J. Globally-biased BIRECT algorithm with local accelerators for expensive global optimization. *Expert Syst. Appl.* **2020**, *144*, 113052. [\[CrossRef\]](#)
- Jones, D.R.; Martins, J.R. The DIRECT algorithm: 25 years Later. *J. Glob. Optim.* **2021**, *79*, 521–566. [\[CrossRef\]](#)
- Peng, Y.; Sun, K.; He, S.; Peng, D. Parameter identification of fractional-order discrete chaotic systems. *Entropy* **2019**, *21*, 27. [\[CrossRef\]](#)
- He, Q.; Wang, L.; Liu, B. Parameter estimation for chaotic systems by particle swarm optimization. *Chaos Solitons Fractals* **2007**, *34*, 654–661. [\[CrossRef\]](#)
- Rosalie, M.; Kieffer, E.; Brust, M.R.; Danoy, G.; Bouvry, P. Bayesian optimisation to select Rössler system parameters used in Chaotic Ant Colony Optimisation for Coverage. *J. Comput. Sci.* **2020**, *41*, 101047. [\[CrossRef\]](#)
- Matsushita, H.; Kurokawa, H.; Kousaka, T. Bifurcation analysis by particle swarm optimization. *Nonlinear Theory Appl. IEICE* **2020**, *11*, 391–408. [\[CrossRef\]](#)
- Tlelo-Cuautle, E.; De La Fraga, L.G.; Guillén-Fernández, O.; Silva-Juárez, A. *Optimization of Integer/Fractional Order Chaotic Systems by Metaheuristics and Their Electronic Realization*; CRC Press: Boca Raton, FL, USA, 2021.
- Hegger, R.; Kantz, H.; Schreiber, T. Practical implementation of nonlinear time series methods: The TISEAN package. *Chaos Interdiscip. J. Nonlinear Sci.* **1999**, *9*, 413–435. [\[CrossRef\]](#) [\[PubMed\]](#)
- Pandey, A.; Baghel, R.; Singh, R. Analysis and circuit realization of a new autonomous chaotic system. *Int. J. Electron. Commun. Eng.* **2012**, *5*, 487–495.
- Lorenz, E.N. Deterministic nonperiodic flow. In *The Theory of Chaotic Attractors*; Springer: Cham, Switzerland, 2004; pp. 25–36.
- Letellier, C.; Rossler, O.E. Rossler attractor. *Scholarpedia* **2006**, *1*, 1721. [\[CrossRef\]](#)
- Lu, J.; Chen, G.; Yu, X.; Leung, H. Design and analysis of multiscroll chaotic attractors from saturated function series. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2004**, *51*, 2476–2490. [\[CrossRef\]](#)
- Tlelo-Cuautle, E.; Pano-Azucena, A.D.; Guillén-Fernández, O.; Silva-Juárez, A. *Analog/Digital Implementation of Fractional Order Chaotic Circuits and Applications*; Springer: Cham, Switzerland, 2020.
- Chua, L.O. Algorithms and Computational Techniques. In *Computer-Aided Analysis of Electronic Circuits*; Prentice Hall: Hoboken, NJ, USA, 1975.
- Epperson, J.F. *An Introduction to Numerical Methods and Analysis*; John Wiley & Sons: Hoboken, NJ, USA, 2021.
- Runge, C. Über die numerische Auflösung von Differentialgleichungen. *Math. Ann.* **1895**, *46*, 167–178. [\[CrossRef\]](#)
- LeVeque, R.J. *Finite Difference Methods for Ordinary and Partial Differential Equations: Steady-State and Time-Dependent Problems*; SIAM: Philadelphia, PA, USA, 2007.
- Dahlquist, G.; Björck, Å. *Numerical Methods in Scientific Computing*; SIAM: Philadelphia, PA, USA, 2008; Volume I.
- Shampine, L.F.; Allen, R.C.; Pruess, S. *Fundamentals of Numerical Computing*; Wiley: New York, NY, USA, 1997; Volume 1.

-
30. Dorigo, M.; Stützle, T. Ant colony optimization: Overview and recent advances. In *Handbook of Metaheuristics*; Springer: Cham, Switzerland, 2019; pp. 311–351.
 31. Wahid, A.; Behera, S.; Mohapatra, D. Artificial Bee Colony and its Application: An Overview. *Int. J. Adv. Res. Comput. Eng. Technol. (IJARCET)* **2015**, *4*, 1475–1480.
 32. Wang, D.; Tan, D.; Liu, L. Particle swarm optimization algorithm: An overview. *Soft Comput.* **2018**, *22*, 387–408. [[CrossRef](#)]
 33. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360), Anchorage, AK, USA, 4–9 May 1998; pp. 69–73.
 34. Saha, N.; Panda, A.; Panda, S. Speed control with torque ripple reduction of switched reluctance motor by many optimizing liaison technique. *J. Electr. Syst. Inf. Technol.* **2018**, *5*, 829–842. [[CrossRef](#)]
 35. Mohanty, M.; Kumar Sahu, R.; Panda, S. A novel hybrid many optimizing liaisons gravitational search algorithm approach for AGC of power systems. *Automatika* **2020**, *61*, 158–178. [[CrossRef](#)]
 36. Pant, M.; Zaheer, H.; Garcia-Hernandez, L.; Abraham, A. Differential evolution: A review of more than two decades of research. *Eng. Appl. Artif. Intell.* **2020**, *90*, 103479.
 37. Cui, L.; Chen, C.; Jin, J.; Yu, F. Dynamic Analysis and FPGA Implementation of New Chaotic Neural Network and Optimization of Traveling Salesman Problem. *Complexity* **2021**, *2021*, 5521192. [[CrossRef](#)]
 38. Tian, K.; Grebogi, C.; Ren, H.P. Chaos Generation With Impulse Control: Application to Non-Chaotic Systems and Circuit Design. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2021**, *68*, 3012–3022. [[CrossRef](#)]
 39. Parker, T.S.; Chua, L. *Practical Numerical Algorithms for Chaotic Systems*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2012.