Implementación en Matlab del Método Adam-Bashforth-Molton para Resolver Sistemas Caóticos de Orden Fraccional

Ramon Ulises-Almada Prieto[a], *Dr. José Cruz-Núñez Pérez*[a], *M. C. Gilberto Enrico-Vazquez Alcaraz*[b].

[a] Instituto Politécnico Nacional, IPN-CITEDI, ramon_ulises@citedi.mx, nunez@citedi.mx, Tijuana, Baja California, México.
[b] Tecnológico Nacional de México, Instituto Tecnológico de Tijuana, gilberto.vazquez@tectijuana.edu.mx, Tijuana, Baja California, México.

Resumen

En este artículo se implementa en Matlab el método numérico de Adam-Bashforth-Molton, con el objetivo de resolver sistemas caóticos de orden fraccional, abarcando sistemas caóticos desde 3 hasta *N* cantidad de dimensiones, tanto para sistemas con el mismo ó con distinto orden fraccional. Para validar la eficacia del método, se presentan los resultados obtenidos con el oscilador caótico de orden fraccional de Lorenz, explorando diversas configuraciones en el parámetro de bifurcación y variando el orden fraccional del sistema. Además, se realizan comparaciones entre los resultados obtenidos con la solución propuesta en este artículo y el código fde12 de Garrapa. La principal contribución de esta investigación radica en el desarrollo de una metodología en Matlab capaz de resolver sistemas caóticos de orden fraccional de N dimensiones.

Palabras clave— Adam-Bashforth-Molton, Caos, Matlab, Método Numérico, Orden fraccional.

Abstract

In this article, the numerical method of Adam-Bashforth-Molton is implemented in Matlab with the aim to solve fractional order chaotic systems, covering chaotic systems ranging from 3 up to N dimensions, for systems with both the same and different fractional orders. To validate the effectiveness of the method, the results obtained with the fractional-order chaotic oscillator of Lorenz are presented, exploring various configurations in the bifurcation parameter and varying the fractional order of the system. Additionally, comparisons are made between the results obtained with the proposed solution in this article and Garrapa's fde12 code. The main contribution of this research lies in the development of methodology capable of solving fractional-order chaotic systems of N dimensions.

Keywords— *Adam-Bashforth-Molton, Chaos, Fractional order, Matlab, Numerical Method.*

1. INTRODUCCIÓN

Actualmente existen gran variedad de métodos numéricos para resolver sistemas caóticos, tales como el método de Euler, Runge Kutta, entre otros. Si bien muchos de estos métodos son bastante conocidos y existen programas en diferentes lenguajes basados en dichos métodos, el problema radica en que dichos métodos no son capaces de solucionar sistemas caóticos de orden fraccional. Por lo que es necesario utilizar distintos métodos numéricos los cuales no son tan conocidos, aquí la importancia de encontrar un método capaz de resolver sistemas caóticos de orden fraccional y desarrollar una herramienta o software basado en dicho método en un sistema de cómputo popular como Matlab.

Desde el siglo XV, René Descartes aseguraba que cualquier fenómeno podría ser explicado si se conocen los suficientes datos de este. Afirmando que realmente no existe el azar, estableciendo que el universo no es más que un gran sistema caótico, el cual, pareciera que se comporta de forma aleatoria. Sin embargo, fue hasta 1963 que Lorenz, al realizar pruebas con el modelo meteorológico que desarrolló, descubrió que dicho sistema poseía una alta sensibilidad ante valores iniciales, descubriendo así el primer oscilador caótico [1].

De igual forma, el cálculo fraccional ha sido estudiado desde hace más de 300 años y si bien, dichas ecuaciones u operaciones solo representaban problemas matemáticos sin aplicaciones reales, hoy en día, con los avances tecnológicos es posible utilizar dichos conceptos para problemas y aplicaciones reales. Tal es el caso de X. Wang, Y. He y M. Wang [2] quienes diseñaron un sistema para controlar el caos de dinamos acoplados utilizando ecuaciones de orden fraccional.

En este trabajo de investigación se plantea el diseño de un software como herramienta basado en el método numérico de Adam-Bashforth-Molton para resolver sistemas caóticos de orden fraccional con el propósito de brindar una alternativa eficaz para resolver este tipo de sistemas. Actualmente existen herramientas similares a la diseñada en este trabajo, tal es el caso del código fde12 diseñado por Garrapa [3] para Matlab. Por lo que en el presente trabajo se realizan comparaciones entre ambas herramientas para comprobar y demostrar la eficacia que tiene el software desarrollado en el presente trabajo de investigación con respecto del desarrollado por Garrapa. La principal contribución del software desarrollado en este trabajo, es ser capaz de resolver sistemas caóticos de orden fraccional de N dimensiones con el mismo y distinto orden fraccional.

Este articulo está organizado de la siguiente manera, la sección 2 contiene teoría relevante para desarrollar los objetivos de este artículo, por lo que se encuentran temas relacionados con sistemas de orden fraccional, sistemas caóticos, método numérico de Adams-Bashforth-Molton, así como sus fundamentos. En la sección 3 se muestran los resultados obtenidos con el software desarrollado para el sistema de Lorenz, junto con los resultados de otro software que también resuelve sistemas caóticos de orden fraccional en Matlab diseñado por Garrapa. Por último, en la sección 4 se presentan las conclusiones finales del articulo realizando comparaciones entre los resultados obtenidos con ambos programas.

2. MARCO TEÓRICO

En esta sección se explicarán los fundamentos teóricos sobre los sistemas caóticos de orden fraccional, así como el método numérico de Adam-Bashforth-Molton.

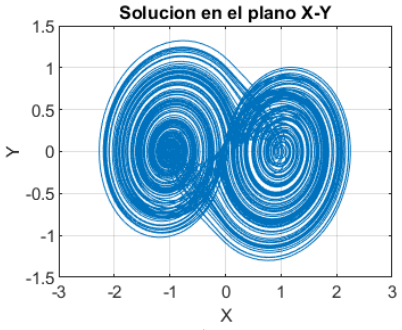
2.1. Osciladores caóticos

Según la teoría del caos, no existen los sistemas con un comportamiento aleatorio, solo que, al no conocer completamente el sistema, este sistema aparentemente posee un comportamiento aleatorio o imposible de predecir. Para que un sistema dinámico posea un comportamiento caótico se debe cumplir lo siguiente: Al menos tres variables dependientes y debe existir por lo menos un término no lineal dentro de dichas ecuaciones. Tomando en cuenta lo anterior, las características de un sistema caótico son las siguientes: 1) no linealidad, 2) determinístico, y 3) alta sensibilidad ante los valores iniciales, esto significa que, aunque parezca que el sistema se comporta de manera aleatoria provocado por su alta sensibilidad ante los valores iniciales, realmente es posible predecir el comportamiento del sistema, aunque conforme transcurre el tiempo, dicha predicción de resultados pierde fiabilidad [4].

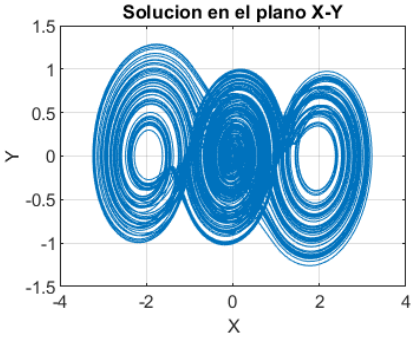
Al graficar la fase de los osciladores caóticos, como su nombre lo indica, estos poseen oscilaciones, es decir, tienden a evolucionar a ciertas zonas llamadas atractores o puntos de estabilidad. Es importante aclarar que siempre y cuando los parámetros del sistema no cambien, los puntos de equilibrio del sistema serán los mismos independientemente de el o los valores de orden fraccional. Dependiendo la cantidad de puntos de equilibrio del sistema es el número de enrollamientos que posee. Dicha estructura se rige por un mecanismo de estiramiento y doblamiento del conjunto de trayectorias, lo que provoca la alta sensibilidad ante los valores iniciales [5].

En la Fig. 1 se observan dos ejemplos de un oscilador basado en funciones saturadas no lineales con a) 2 y b) 3 enrollamientos.

Fig. 1. Oscilador caótico visto desde el plano X-Y con: a) 2 enrollamientos b) 3 enrollamientos



a)



b)

Fuente: elaboración propia a partir de [5].

2.2. Cálculo fraccional

La rama del análisis numérico conocida como cálculo fraccional se dedica a explorar la viabilidad de aplicar potencias reales a los operadores integral y derivativo, en lugar de restringir dichos operadores a órdenes de naturaleza entera. Aunque inicialmente pueda parecer desafiante resolver derivadas o integrales fraccionarias (sistemas de orden fraccional), el primer paso hacia dicha solución implica obtener una fórmula general para resolver una integral de grado *n*, tal y como se muestra en la ecuación (1),

$$\mathcal{D}^{-n}f(t) = \int_0^t \frac{f(y)(t-y)^{n-1}}{(n-1)!}dy, \qquad (1)$$

donde *n* es un numero entero positivo y *t* representa el tiempo.

También es importante tomar en cuenta la función Gamma (Γ). Esta función hace posible utilizar la notación factorial con números reales y complejos, siempre y cuando la parte real del número sea positiva [6]. Tomando en cuenta lo anterior, Cauchy decidió sustituir dicha función factorial por la función Γ, la cual se observa en la ecuación (2),

$$\Gamma(z) = \lim_{n\to\infty} \int_0^n t^{z-1}e^{-t}dt \qquad (2)$$

De esta manera se puede expresar de manera general una integral de orden real o de orden fraccional (FO por sus siglas en inglés). En la ecuación (3) se aprecia la fórmula de Cauchy donde *n* es sustituido por α, representando así el orden real [7].

$$\mathcal{D}^{\alpha}f(t) = \frac{1}{\Gamma(n)}\int_0^t f(y)(t-y)^{\alpha-1}dy \qquad (3)$$

Derivada fraccional de Riemman-Liouville

Si bien, gracias a Cauchy fue posible calcular una integral de FO, ahora el problema era obtener una derivada de FO. Tomando en consideración que la operación opuesta de una integral es una derivada se desarrolló el método de derivación

fraccional de Riemann-Liouville, el cual consiste en, si se desea obtener la derivada de FO de cualquier función. Primero se realiza una integral de FO y después se deriva enteramente [8]. Por ejemplo, si se desea obtener la derivada de grado ½, primero se debe obtener la integración fraccional cuando α = ½ y después, a dicho resultado se le aplica la derivada de primer orden. La derivación fraccional de Riemann-Liouville se expresa en la ecuación (4),

$$\mathcal{D}^{-\alpha}f(t) = \frac{1}{\Gamma(n-\alpha)}\frac{d^n}{dx^n}\int_{-\infty}^{x}\frac{f(t)}{(x-t)^{\alpha-n+1}}dt, \quad -\infty < x < \infty \tag{4}$$

Derivada fraccional de Caputo

El problema con la derivada de Riemann-Liouville es que, al realizar modelos matemáticos de fenómenos físicos reales por medio de ecuaciones diferenciales de FO, es necesario trabajar con condiciones iniciales de FO, las cuales no tienen una interpretación física clara. Es por eso que el operador diferencial de Caputo invierte el orden de la derivación en la definición de Riemann-Liouville, es decir, primero se debe realizar la derivada de grado entero y posteriormente se realiza la integración fraccional. De esta manera se utilizan las condiciones iniciales derivadas de orden entero [8]. En la ecuación (5) se observa el operador diferencial de Caputo,

$$\mathcal{D}^{-\alpha}f(t) = \frac{1}{\Gamma(n-\alpha)}\int_{\alpha}^{t}(t-s)^{n-\alpha-1}f^{(n)}(s)ds, \ t > \alpha. \tag{5}$$

2.3. Método de Adams-Bashforth-Molton para sistemas de orden fraccional

Dada la naturaleza de los sistemas de ecuaciones de FO, no es posible resolverlos de manera numérica de la misma forma que para los sistemas de ecuaciones de orden entero, es por eso que los métodos de Euler mejorada y Runge-Kutta de cuarto orden no se utilizan para describir el comportamiento de un sistema de FO. No obstante, hoy en día existen numerosas investigaciones relacionadas a dar solución a este tipo de ecuaciones. Entre los cuales destaca el método basado en el predictor-corrector Adams-Bashforth-Moulton (ABM), esto porque se puede implementar en código y por su alta precisión [9]. Las ecuaciones (6) y (7) corresponden al método basado en el predictor corrector ABM.

$$x_h^p(t_{n+1}) = \sum_{k=0}^{techo(q)-1}\frac{t_{n+1}^k}{k!}x_0^{(k)} + \frac{1}{\Gamma(q)}\sum_{j=0}^{n}b_{j,n+1}f\left(t_j, x_h(t_j)\right), \tag{6}$$

$$x_h(t_{n+1}) = \sum_{k=0}^{techo(q)-1}\frac{t_{n+1}^k}{k!}x_0^{(k)} + \frac{h^q}{\Gamma(q+2)}f\left(t_{n+1}, x_h{}^p(t_{n+1})\right) + \frac{h^q}{\Gamma(q+2)}\sum_{j=0}^{n}a_{j,n+1}f\left(t_j, x_h(t_j)\right), \tag{7}$$

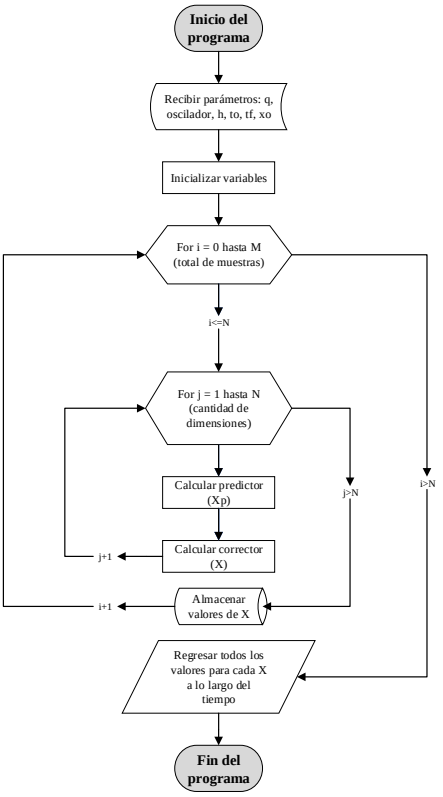
donde $x(t_{n+1})$ es el valor siguiente de x, a y b son magnitudes determinadas por el modelo predictor-corrector ABM, y dichos parámetros se obtienen utilizando las ecuaciones (8) y (9), donde $x_0^{(k)}$ es el valor inicial para la derivada de orden k.

$$b_{j,n+1} = \frac{h^q}{q}((n+1-j)^q - (n-j)^q), \tag{8}$$

$$a_{j,n+1} = \begin{cases} n^{q+1} - (n-q)(n+1)^q & j = 0 \\ (n-j+2)^{q+1} + (n-j)^{q+1} - 2(n-j+1)^{q+1} & 1 \le j \le n, \\ 1 & j = n+1 \end{cases} \tag{9}$$

donde q es el FO, h es el salto de paso entre iteraciones. Para el modelo predictor-corrector ABM el error obtenido suele ser de aproximadamente h^2, por lo que, si se desea obtener un error máximo de 1×10^{-6}, el valor máximo de h deberá ser mínimo de 1×10^{-3} [9]. En la Fig. 2 se presenta un diagrama de flujo basado en el método de ABM.

Fig. 2. Diagrama de flujo del método ABM.



Fuente: elaboración propia a partir de las ecuaciones (6-9)

En la Fig. 2 se observa que primero se inicializan algunas variables auxiliares y posteriormente se entra en un ciclo for el cual es responsable de realizar las operaciones para calcular los valores de las variables en cada iteración. Para esto, dentro de dicho ciclo for se presenta otro ciclo for en el cual se realizan las operaciones para calcular el predictor y posteriormente el corrector (el siguiente valor) de una de las variables. Dicho ciclo for es el responsable de realizar la siguiente iteración para cada una de las variables del sistema. Una vez que se calculan los valores de la siguiente iteración, estos se almacenan en una matriz X. Mientras que, cuando se realizan todas las iteraciones el programa retorna a una variable todos los valores almacenados en la matriz X. Por último, en la Fig. 3 se observa un pseudocódigo basado en el método de ABM.

Fig. 3. Pseudocódigo del método ABM.

```
X =ABM(q,f,t0,tf,xo,h)
Begin

% Inicializar variables auxiliares
% Cantidad de dimensiones
% Asignar valores iniciales
% Ciclo for para calcular valores en cada iteración
% Ciclo for para calcular valores de cada X para el tiempo
siguiente
%Inicializar valores para cada sumatoria
% Ciclo for para calcular la primer sumatoria del predictor
% Calcular la segunda sumatoria del predictor
% Valor del predictor
% Ciclo for para calcular la primer sumatoria del corrector
% Calcular la segunda sumatoria del corrector
% Valor del predictor
% Fin de ciclos for

End
```

Fuente: elaboración propia a partir de las ecuaciones (6-9)

2.4. Oscilador de Lorenz de orden fraccional

El oscilador de Lorenz es un sistema dinámico no lineal con valores iniciales tanto para un orden entero como para un FO [10]. El oscilador de Lorenz, en su versión de orden fraccional es modelado por el operador derivativo fraccional de Caputo tal y como se observa en el sistema de ecuaciones (10),

$$\begin{aligned} D_*^q x_1 &= \sigma(x_2 - x_1) \\ D_*^q x_2 &= -x_1 x_3 + p x_1 - x_2, \\ D_*^q x_3 &= x_1 x_2 - \beta x_3 \end{aligned} \qquad (10)$$

donde D_*^q es el operador diferencial de Caputo de orden $0 < q < 1$, x_1, x_2 y x_3 son las variables dependientes del sistema, σ y β son parámetros del sistema y p es el parámetro de bifurcación. Dicho sistema tiene los mismos puntos de equilibrio en su versión de FO y en su versión de orden entero, los cuales se calculan con las ecuaciones (11 – 13),

$$PE_{L1} = [0,0,0] \qquad (11)$$

$$PE_{L2} = \left[\sqrt{\beta(p-1)}, \sqrt{\beta(p-1)}, p-1\right] \qquad (12)$$

$$PE_{L3} = \left[-\sqrt{\beta(p-1)}, -\sqrt{\beta(p-1)}, p-1\right] \qquad (13)$$

Siempre y cuando p sea mayor que 1, los puntos PE_{L2} y PE_{L3} existen. Mientras que la matriz Jacobiana del sistema de Lorenz es representada en la ecuación (14),

$$J_L = \begin{pmatrix} -\sigma & \sigma & 0 \\ -x_3 + p & -1 & -x_1 \\ x_2 & x_1 & -\beta \end{pmatrix} \qquad (14)$$

Para que el oscilador de Lorenz posea un comportamiento estable los parámetros, σ, β y p deben ser mayores a 0. Por otro lado, para que el sistema tenga un comportamiento caótico estable los parámetros del sistema deben cumplir con lo siguiente: $\sigma = 10$, $\beta = 8/3$, $p \geq 28$ y el FO $q \geq 0.985$ [11].

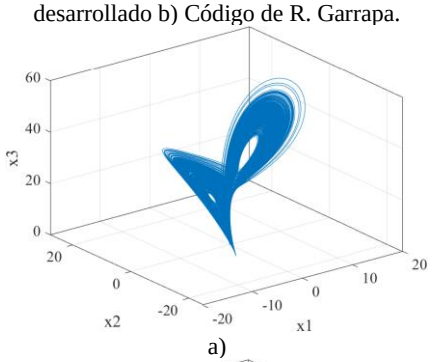
3. RESULTADOS

Una vez desarrollado el código en Matlab basado en el predictor corrector de Adams-Bashforth-Molton, se realizaron simulaciones para tres casos distintos con el sistema de Lorenz utilizando la propuesta de este trabajo de investigación y utilizando el código fde12 de Garrapa.

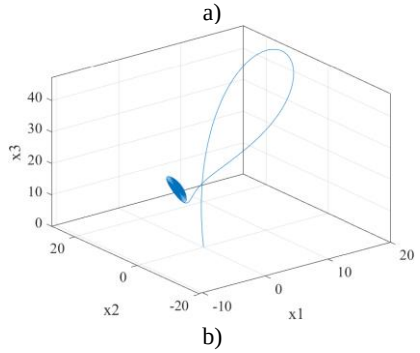
3.1. Sistema de Lorenz con p = 28

Para las primeras simulaciones se utilizó el sistema de Lorenz con $\sigma = 10$, $\beta = 8/3$, $p = 28$ y el FO $q = 0.985$, un $t_0 = 0$s $t_f =$ 500s y un muestro h de 0.005. Al realizar las simulaciones con el código diseñado y con el código de Garrapa se obtuvieron las simulaciones de las Fig 4 y 5.

Fig. 4. Sistema de Lorenz en 3D con p = 28 resuelto por: a) Código desarrollado b) Código de R. Garrapa.



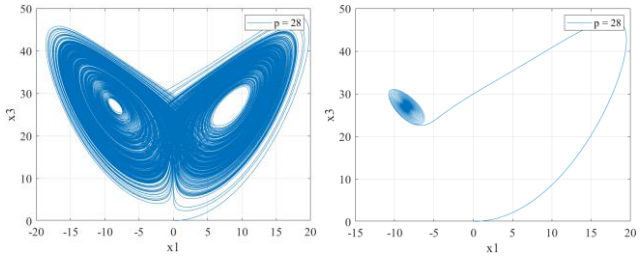
a)



b)

Fuente: elaboración propia a partir del sistema (10)

Fig. 5. Sistema de Lorenz desde los planos X1-X3 con p = 28 resuelto por: a) Código desarrollado b) Código de R. Garrapa.



a) b)

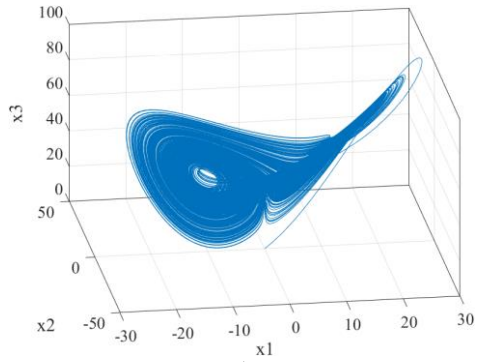
Fuente: elaboración propia a partir del sistema (10)

De igual forma, es importante destacar que, al realizar las anteriores simulaciones, el código realizado en este artículo tardo 1.57 segundos en calcular todas las muestras del sistema, mientras que el código realizado por Garrapa, en las mismas condiciones y con el mismo equipo de cómputo tardo 6.43 segundos.

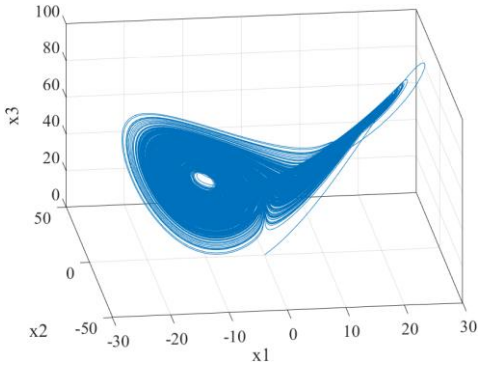
3.2. Sistema de Lorenz con p = 50

Para las primeras simulaciones se utilizó el sistema de Lorenz con $\sigma = 10$, $\beta = 8/3$, $p = 50$ y el FO $q = 0.985$, un t0 = 0s tf = 500s y un tiempo de muestro h de 0.005s. Al realizar las simulaciones con el código diseñado y con el código de R. Garrapa se obtuvieron las simulaciones de las Fig. 6 y 7.

Fig. 6. Sistema de Lorenz en 3D con p = 50 resuelto por: a) Código desarrollado b) Código de R. Garrapa.



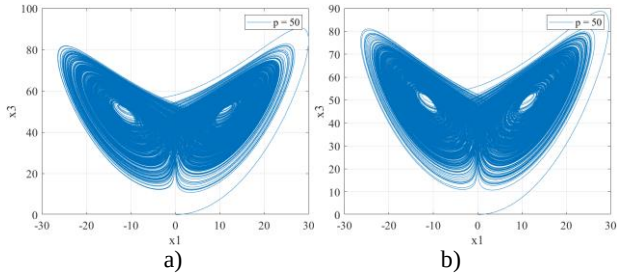
a)



b)

Fuente: elaboración propia a partir del sistema (10)

Fig. 7. Sistema de Lorenz desde los planos X1-X3 con p = 50 resuelto por: a) Código desarrollado b) Código de R. Garrapa.



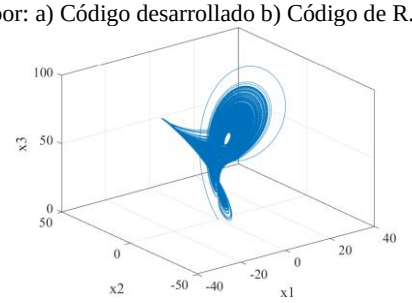
a) b)

Fuente: elaboración propia a partir del sistema (10)

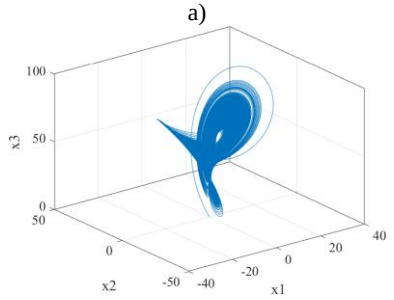
3.3. Sistema de Lorenz con q = [0.985 0.990 0.995]

En este caso se utilizó el sistema de Lorenz con $\sigma = 10$, $\beta = 8/3$, $p = 50$ y el FO $q = [0.985\ 0.990\ 0.995]$, un $t_0 = 0$s, $t_f = 500$s y un muestro h de 0.005. Los resultados con la propuesta y el código de Garrapa se muestran en las Figuras 8 y 9.

Fig. 8. Sistema de Lorenz en 3D con q = [0.985 0.990 0.995] resuelto por: a) Código desarrollado b) Código de R. Garrapa.



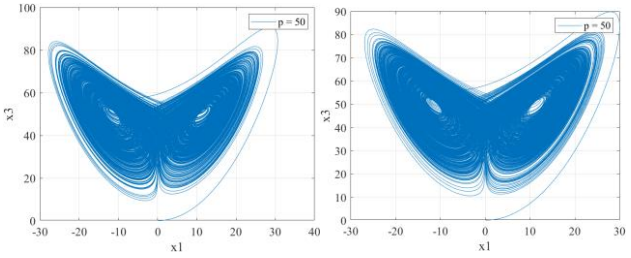
a)



b)

Fuente: elaboración propia a partir del sistema (10)

Fig. 9. Sistema de Lorenz desde los planos X_1-X_3 con q = [0.985 0.990 0.995] resuelto por: a) Propuesta b) Código de Garrapa.



a) b)

Fuente: elaboración propia a partir del sistema (10)

4. CONCLUSIONES

En este trabajo de investigación se desarrolló un software en Matlab el cual resuelve cualquier sistema caótico de orden fraccional de N dimensiones y con el mismo o distinto orden fraccional. Por otro lado, al comparar los resultados del punto se observa que las gráficas obtenidas con la propuesta para el sistema de Lorenz poseen el mismo comportamiento que las obtenidas con el código de Garrapa. Por lo que se confirma que el código trabaja correctamente y es capaz de resolver sistemas caóticos de orden fraccional con el mismo y distinto orden fraccional. De igual forma, al analizar los resultados del punto 3.1 se observa que, para un parámetro de bifurcación de 28 en el sistema de Lorenz, el código de Garrapa muestra que el sistema converge a un valor, perdiendo el comportamiento caótico del sistema, mientras que el código diseñado en este trabajo sigue mostrando un comportamiento caótico oscilatorio. Por lo que, para este caso, el código diseñado en este trabajo representa de mejor forma el comportamiento del sistema. Por último, al comparar los tiempos de ejecución se obtuvo que el código diseñado en este artículo es 4.09 veces más eficiente que el código de R. Garrapa. De manera que el código diseñado es más eficiente y eficaz que el propuesto por Garrapa para resolver el sistema de Lorenz.

5. REFERENCIAS

[1] S. Cang, A. Wu, Z. Wang, Z. Chen, “Four-dimensional autonomous dynamical systems with conservative flows: two-case study”, Nonlinear Dynamics, 89(4), pp. 2495-2508, 2017.

[2] X. Wang, Y. He, M. Wang “Chaos control of a fractional order modified coupled dynamos system”. Nonlinear Analysis: Theory, Methods & Applications, 71(12), pp. 6126-6134, 2009.

[3] R. Garrapa, “Numerical Solution of Fractional Differential Equations: A Survey and a Software Tutorial”. Mathematics, 6(2), p.16, 2018.

[4] P. Obeso, “Diseño e implementación en un FPGA de oscilador caótico para aplicaciones en sistemas de seguridad”, Tesis de maestría, IPN-CITEDI, Baja California, México, 2015.

[5] J. Cortes-Avilez, A. Bonilla-Rodríguez, R. Serrano-Andrade, G. Entrambasaguas-León, A. Calvillo-Téllez, J.C. Núñez-Pérez, “Diseño de sistemas caóticos de multi-enrrollamiento y multi-direcciones usando SNLF”, Revista Aristas 1(12), pp. 156-162. 2018.

[6] J. M. Sánchez-Muñoz, “Génesis y desarrollo del cálculo fraccional”, Pensamiento Matemático, 6(1), pp. 1-15, 2011.

[7] M. Vinagre, V. Feliu-Batlle, I. Tejado, “Control fraccionario: fundamentos y guía de uso”, Revista Iberoamericana de Automática e Información Industrial, 13(1). pp. 265-280. 2016.

[8] M. Rodríguez-Martin, “Introducción al cálculo fraccionario y a los modelos de crecimiento tumoral clásico y fraccionarios”, trabajo de obtención de grado en matemáticas, FDM, UCM, Madrid, España, 2020.

[9] K. Diethelm, N. Ford, A. Freed, “A predictor corrector Approach for the Numerical Solution of Fractional Differential Equations”, Nonlinear Dynamics, 29(1), pp. 3-22, 2002.

[10] E.N. Lorenz, “Deterministic Nonperiodic Flow”, Journal of the Atmospheric Sciences, 20(2), pp. 130-141, 1963.

[11] Y. Yu, H. X. Li, S. Wang, J. Yu, “Dynamic analysis of a fractional-order Lorenz chaotic system”. Chaos, Solitons & Fractals, 42(2), pp. 1181-1189, 2009.