

CONSTRUCCIÓN Y ESTUDIO DE CÓDIGOS
ADAPTATIVOS DE LINEALIZACIÓN LOCAL PARA
ECUACIONES DIFERENCIALES ORDINARIAS

CONSTRUCTION AND STUDY OF LOCAL
LINEARIZATION ADAPTIVE CODES FOR
ORDINARY DIFFERENTIAL EQUATIONS

ALINA SOTOLONGO AGUIAR*
JUAN CARLOS JIMÉNEZ SOBRINO†

*Received: 5/Oct/2011; Revised: 21/Jul/2013;
Accepted: 5/Nov/2013*

*Facultad de Matemática y Computación, Universidad de la Habana, La Habana, Cuba.
E-Mail: alina.sotolongo@gmail.com

†Instituto de Cibernética, Matemática y Física, Calle 15, No 551, La Habana, Cuba. E-Mail:
jcarlos@icimaf.cu

Resumen

El propósito de este trabajo es construir códigos adaptativos del método de Linealización Local para Ecuaciones Diferenciales Ordinarias (EDO) y analizar su comportamiento numérico. Además, se estudia el efecto que sobre las propiedades de los códigos produce la variación en la precisión de las aproximaciones de Padé utilizadas.

Palabras clave: Integradores numéricos, código adaptativo, método de linealización local, método de Runge Kutta, fórmula de Padé.

Abstract

The aim of this work is to construct adaptive integrators for ordinary differential equations based on the Local Linearization method. Different orders of the involved Padé approximation are considered and their effect on the adaptive integrators is studied.

Keywords: numerical integrators, adaptive codes, local linearization method, Runge–Kutta method, Padé approximation.

Mathematics Subject Classification: 65L05, 37M05, 65L06.

1 Introducción

Los esquemas de Linealización Local (LL) [15, 16] y de Linealización Local de Orden Superior (LLOS) [6, 7] forman parte de los llamados integradores exponenciales [5, 12, 13] para la solución de Ecuaciones Diferenciales Ordinarias (EDO), los cuales tienen en común el cálculo de una exponencial matricial en cada paso. En general, el desarrollo de esta clase de métodos ha sido estimulado por su capacidad para preservar numerosas propiedades dinámicas de las EDO con tamaños de pasos mucho mayores que los integradores explícitos y con menor costo computacional que los implícitos [9].

En particular el método LL clásico proporciona un integrador exponencial de paso simple que sigue la estrategia de aproximar localmente el campo vectorial de la ecuación diferencial por el término lineal de su representación en serie de Taylor, obteniendo así una EDO lineal que luego es integrada exactamente. En el método LLOS, a la aproximación anterior se le añade un término adicional que representa la solución de una ecuación diferencial auxiliar. Esta última ecuación puede ser resuelta por cualquier integrador numérico, lo que aporta una gran flexibilidad al algoritmo.

Resultados teóricos y de simulaciones muestran que los métodos LL poseen un gran número de propiedades deseables entre las que se encuentran la

A-estabilidad, ausencia de puntos de equilibrio espurios bajo condiciones bastante generales y la preservación del comportamiento dinámico de la solución alrededor de puntos de equilibrio hiperbólicos y órbitas periódicas [6, 7, 9, 15]. Adicionalmente, permite integrar ciertos tipos de ecuaciones “stiff” con menor costo computacional y mejor precisión que los integradores explícitos convencionales. Otra característica ventajosa del método LL es la flexibilidad de su aplicación a otras clases de ecuaciones diferenciales más complejas como por ejemplo las ecuaciones diferenciales estocásticas [8, 2], las aleatorias [3] y las ecuaciones con retardo [17].

Estas razones justifican la construcción de códigos de Linealización Local eficientes que permitan la integración de las complejas EDO que usualmente aparecen en situaciones prácticas. Para ello un elemento indispensable es la inclusión en el código de una estrategia adaptativa para la selección del tamaño de paso, lo cual permitiría reducir el tiempo total de cómputo mediante un control efectivo del error de integración local. Este aspecto no ha sido considerado en los códigos LL desarrollados hasta la fecha. La construcción de esquemas LL adaptativos y su análisis constituyen los primeros objetivos de este trabajo.

Otro aspecto fundamental en la construcción de códigos LL eficientes es la reducción de su costo computacional. Es conocido que la mayor parte de ese costo se debe a la evaluación de una exponencial matricial en cada paso. Típicamente, para ello se utiliza la aproximación racional de Padé combinado con el procedimiento de escalamiento y potenciación, lo cual proporciona una aproximación cuya precisión depende del orden escogido. En la segunda parte de este trabajo, se plantea como objetivo estudiar el efecto en la precisión y las propiedades dinámicas de los esquemas LL al variar el orden del algoritmo de Padé, lo cual se presenta en [16] como un problema abierto de gran interés en el análisis de los métodos LL y LLOS.

Este trabajo está organizado de la siguiente forma. En la sección 1 se expone una breve revisión de los métodos LL necesaria para la comprensión de este estudio. En la sección 2 y 3 se presentan, respectivamente, la estrategia adaptativa y de selección del orden de la aproximación de Padé que serán utilizadas para construir nuevos códigos LL. En la última sección se muestran simulaciones que ilustran las cualidades de los nuevos códigos.

2 Revisión de los métodos de linealización local

Consideremos una ecuación diferencial ordinaria

$$\begin{aligned}\frac{d\mathbf{x}(t)}{dt} &= \mathbf{f}(t, \mathbf{x}(t)) \\ \mathbf{x}(t_0) &= \mathbf{x}_0\end{aligned}\tag{1}$$

donde $\mathbf{f} : [t_0, T] \times D \rightarrow \mathbb{R}^d$ es una función diferenciable en un abierto $D \subset \mathbb{R}^d$, y asumiremos que las condiciones de existencia y unicidad de la solución de la ecuación se cumplen en D .

Consideremos, además, una partición $(t)_h = \{t_n : n = 0, 1, \dots, N\}$ del intervalo $[t_0, T]$ tal que $t_0 < t_1 < \dots < t_N = T$, y $h_n = t_{n+1} - t_n \leq h$; $\forall n = 0, \dots, N-1$.

2.1 Aproximación Lineal Local

El método clásico de Linealización Local [15, 16] se basa en aproximar en cada paso el campo vectorial de (1) mediante la expansión de Taylor de primer orden, para luego resolver exactamente la ecuación lineal resultante. Más precisamente, si $\mathbf{A}_n \mathbf{y}(t) + \mathbf{a}_n(t)$ denota la aproximación de Taylor de primer orden de $\mathbf{f}$ en una vecindad de $(t_n, \mathbf{y}_n)$, donde $\mathbf{A}_n = \mathbf{f}_x(t_n, \mathbf{y}_n)$ y $\mathbf{a}_n(t) = \mathbf{f}_t(t_n, \mathbf{y}_n)(t - t_n) + \mathbf{f}(t_n, \mathbf{y}_n) - \mathbf{A}_n \mathbf{y}_n$, entonces la solución de la ecuación

$$\begin{aligned}\frac{d\mathbf{y}(t)}{dt} &= \mathbf{A}_n \mathbf{y}(t) + \mathbf{a}_n(t) \\ \mathbf{y}(t_n) &= \mathbf{y}_n\end{aligned}\tag{2}$$

para todo $t \in [t_n, t_{n+1}]$ es una aproximación de la solución de (1) con condición inicial $\mathbf{x}(t_n) = \mathbf{y}_n$. Mediante la fórmula de variación de la constante obtenemos

$$\mathbf{y}(t) = \mathbf{y}_n + \phi(t_n, \mathbf{y}_n; t - t_n)\tag{3}$$

donde

$$\phi(t_n, \mathbf{y}_n; t - t_n) = \int_0^{t-t_n} e^{\mathbf{A}_n(t-t_n-u)} (\mathbf{A}_n \mathbf{y}_n + \mathbf{a}_n(t_n + u)) du.\tag{4}$$

Aplicando recursivamente la expresión anterior obtenemos la siguiente definición:

Definición 1 Dada una partición $(t)_h$, la discretización Lineal Local de la solución de (1) está dada por la expresión recursiva

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \phi(t_n, \mathbf{y}_n; h_n),\tag{5}$$

con $\mathbf{y}_0 = \mathbf{x}_0$ y $n = 0, 1, \dots, N-1$.

Para todo $t \in [t_0, T]$ se define:

Definición 2 Dada una partición $(t)_h$, la aproximación Lineal Local de la solución de (1) está dada por

$$\mathbf{y}(t) = \mathbf{y}_{n_t} + \phi(t_{n_t}, \mathbf{y}_{n_t}; t - t_{n_t}) \quad (6)$$

$\forall t \in [t_0, T]$, donde $\mathbf{y}_0 = \mathbf{x}_0$, $\mathbf{y}_{n_t}$ es la discretización Lineal Local (5) y $n_t = \max\{n : t_n \leq t\}$.

2.2 Aproximación Lineal Local de orden superior

La estabilidad y las propiedades dinámicas que el método LL posee son definitivas en su importancia como integrador de EDO, no obstante su bajo orden de convergencia constituye un problema en algunas circunstancias. Para solucionarlo se construyen los métodos de Linealización Local de Orden Superior [6, 7, 9], los cuales retienen las propiedades mencionadas, pero con orden de convergencia mayor. Este método se basa en agregar a la aproximación (6) un término adicional $\mathbf{r}$ de orden superior conformado por la solución numérica de una ecuación diferencial.

Dada una partición $(t)_h$, la aproximación Lineal Local de orden γ de la solución de (1) está definida por la expresión

$$\mathbf{y}(t) = \mathbf{y}_{n_t} + \phi(t_{n_t}, \mathbf{y}_{n_t}; t - t_{n_t}) + \mathbf{r}(t; t_{n_t}, \mathbf{y}_{n_t}), \quad (7)$$

donde $\mathbf{r}(t; t_n, \mathbf{y}_n)$ es la solución numérica de la ecuación auxiliar

$$\begin{aligned} \frac{d\mathbf{u}(t)}{dt} &= \mathbf{q}(t_n, \mathbf{y}_n; t, \mathbf{u}(t)) \\ \mathbf{u}(t_n) &= \mathbf{0} \end{aligned} \quad (8)$$

para todo $t \in [t_n, t_{n+1}]$, con campo vectorial

$$\begin{aligned} \mathbf{q}(t_n, \mathbf{y}_n; s, \xi) &= \mathbf{f}(s, \mathbf{y}_n + \phi(t_n, \mathbf{y}_n; s - t_n) + \xi) \\ &\quad - \mathbf{f}_{\mathbf{x}}(t_n, \mathbf{y}_n) \phi(t_n, \mathbf{y}_n; s - t_n) \\ &\quad - \mathbf{f}_t(t_n, \mathbf{y}_n)(s - t_n) - \mathbf{f}(t_n, \mathbf{y}_n), \end{aligned}$$

con $s \in [t_n, t_{n+1}]$ y $\xi \in \mathbb{R}^d$.

Cuando $\mathbf{r}$ es la aproximación obtenida mediante un esquema Runge Kutta de orden γ se llega a las definiciones:

Definición 3 Dada una partición $(t)_h$, la discretización Lineal Local-Runge Kutta de orden $\gamma > 2$ se define por la expresión recursiva:

$$\mathbf{y}_{n+1} = \mathbf{y}_n + \phi(t_n, \mathbf{y}_n; h_n) + h_n \sum_{j=1}^s b_j \mathbf{k}_j \quad (9)$$

con $\mathbf{y}_0 = \mathbf{x}_0$, donde las constantes b_j y las funciones $\mathbf{k}_j$ están definidas por el método Runge Kutta (γ, s) .

Definición 4 Dada una partición $(t)_h$, la aproximación Lineal Local-Runge Kutta de orden γ se define por

$$\mathbf{y}(t) = \mathbf{y}_{n_t} + \phi(t_{n_t}, \mathbf{y}_{n_t}; t - t_{n_t}) + (t - t_{n_t}) \sum_{j=1}^s b_j \mathbf{k}_j(t_{n_t}, \mathbf{y}_{n_t}; t - t_{n_t}) \quad (10)$$

$\forall t \in [t_0, T]$, donde $\mathbf{y}_0 = \mathbf{x}_0$, $\mathbf{y}_{n_t}$ es la discretización Lineal Local-Runge Kutta (9) y $n_t = \max\{n : t_n \leq t\}$.

2.3 Propiedades de los métodos de Linealización Local

Con el objetivo de ilustrar algunas de las ventajas teóricas y prácticas de calcular la solución de una ecuación diferencial ordinaria mediante las aproximaciones Lineales Locales, expondremos brevemente algunos de los principales resultados sobre la convergencia, estabilidad y propiedades dinámicas de los métodos de Linealización Local.

Convergencia y A-estabilidad

Teorema 1 [15] Sea $\mathbf{x}(t)$ la solución de la ecuación (1), con el campo vectorial $\mathbf{f}$ de clase C^2 en $[t_0, T] \times D$ y segundas derivadas parciales acotadas. Si $\mathbf{f}(t, \cdot)$ y su derivada parcial cumplen una condición de Lipschitz independiente de t , entonces $\forall t_n \in (t)_h$, y h suficientemente pequeño, la aproximación Lineal Local y cumple que:

$$\sup_{t \in [t_0, T]} \|\mathbf{x}(t) - \mathbf{y}(t)\| = O(h^2).$$

Para la aproximación Lineal Local-Runge Kutta de orden γ se tiene el resultado:

Teorema 2 [9] Si el campo vectorial de (1) satisface la condición

$$\mathbf{f} \in C^{\gamma+1}([t_0, T], \mathbb{R}^d),$$

entonces, $\forall t_n \in (t)_h$, y h suficientemente pequeño, la aproximación Lineal Local-Runge Kutta y cumple que:

$$\sup_{t \in [t_0, T]} \|\mathbf{x}(t) - \mathbf{y}(t)\| = O(h^\gamma).$$

Con respecto a la estabilidad para EDO lineales tenemos:

Teorema 3 [9] *Las discretizaciones LL y LLRK son A-estables.*

La A-estabilidad es una característica que distingue a las discretizaciones LL por ser estas de tipo explícito.

Propiedades dinámicas

Es conocido [4, 23, 22] que los integradores convencionales como los métodos de Runge-Kutta y predictor-corrector de Adams-Bashforth-Moulton pueden producir comportamientos dinámicos errados en la integración de sistemas de ecuaciones diferenciales. Los principales problemas se presentan, por ejemplo, en la convergencia a estados estables falsos, cambios en las bases de atracción y aparición de bifurcaciones falsas. La razón de estas dificultades reside en que la dinámica de los esquemas numéricos (vistos como sistemas dinámicos discretos) es más rica que la de la ecuación original. Estos problemas, a diferencia de lo que se puede pensar a priori, pudieran no resolverse disminuyendo el tamaño de paso.

A continuación se presentan algunos de los resultados sobre varios aspectos dinámicos de los métodos LL. Para ello consideraremos la ecuación autónoma

$$\begin{aligned} \frac{d\mathbf{x}(t)}{dt} &= \mathbf{f}(\mathbf{x}(t)), \quad t \in [t_0, T] \\ \mathbf{x}(t_0) &= \mathbf{x}_0 \in \mathbb{R}^d. \end{aligned} \quad (11)$$

Definición 5 [14] *Un integrador numérico se dice regular si todos los puntos de equilibrio del esquema de integración son también puntos de equilibrio del sistema de ecuaciones y viceversa.*

Teorema 4 [15, 9] *Se cumple que:*

- i) *Todos los puntos de equilibrio del sistema de ecuaciones diferenciales son también puntos de equilibrio de las discretizaciones LL de orden γ .*

- ii) Si $\mathbf{f}$ es una función diferenciable que cumple una condición de Lipschitz, entonces las discretizaciones LL de orden γ son regulares para un tamaño de paso suficientemente pequeño.

La propiedad de regularidad es una de las características distintivas de los métodos de Linealización Local, en comparación con los métodos convencionales.

Sea $\mathbf{0}$ un punto de equilibrio hiperbólico de (11), y $\mathcal{X}_s, \mathcal{X}_u$ los subespacios estables e inestables del campo vectorial hiperbólico lineal $\mathbf{f}_x(\mathbf{0})$, tal que $\mathbb{R}^d = \mathcal{X}_s \oplus \mathcal{X}_u$, $(\mathbf{x}_s, \mathbf{x}_u) = \mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\| = \max(\|\mathbf{x}_s\|, \|\mathbf{x}_u\|)$. Para $\varepsilon > 0$ sea $K_\varepsilon = \{\mathbf{x} \in \mathbb{R}^d : \|\mathbf{x}\| \leq \varepsilon\}$.

Teniendo en cuenta que las variedades locales estables e inestables en la vecindad de $\mathbf{0}$ pueden ser representadas como [1]

$$M_s = \{(\mathbf{x}_s, \mathbf{p}(\mathbf{x}_s)) : \mathbf{x}_s \in K_{\varepsilon,s}\}, \quad M_u = \{(\mathbf{q}(\mathbf{x}_u), \mathbf{x}_u) : \mathbf{x}_u \in K_{\varepsilon,u}\},$$

donde $K_{\varepsilon,s} = K_\varepsilon \cap \mathcal{X}_s$, $K_{\varepsilon,u} = K_\varepsilon \cap \mathcal{X}_u$, y $\mathbf{p} : K_{\varepsilon,s} \rightarrow K_{\varepsilon,u}$, $\mathbf{q} : K_{\varepsilon,u} \rightarrow K_{\varepsilon,s}$ son funciones tan suaves como $\mathbf{f}$, se obtiene el siguiente resultado.

Teorema 5 [15, 7, 9] Si se cumplen las condiciones

$$\mathbf{f} \in C^\gamma(\Omega, \mathbb{R}^d) \text{ o } \mathbf{f} \in C^{\gamma+1}(\Omega, \mathbb{R}^d)$$

y

$$\mathbf{x}(t; \mathbf{x}_0) \subset \Omega \text{ para todo } \mathbf{x}_0 \in K \text{ y } t \in [t_0, T]$$

para un compacto $K \subset \mathbb{R}^d$, entonces existen constantes $C, \varepsilon, \varepsilon_0, h_0 > 0$ tales que la variedad estable local M_s^h e inestable M_u^h del mapa de las discretizaciones LL de orden γ en $\mathbf{0}$ ($h \leq h_0$) son de la forma

$$M_s^h = \left\{ \left(\mathbf{x}_s, \mathbf{p}^h(\mathbf{x}_s) \right) : \mathbf{x}_s \in K_{\varepsilon,s} \right\}, \quad M_u^h = \left\{ \left(\mathbf{q}^h(\mathbf{x}_u), \mathbf{x}_u \right) : \mathbf{x}_u \in K_{\varepsilon,u} \right\},$$

donde $\mathbf{p}^h = \mathbf{p} + O(h^\gamma)$ uniformemente en $K_{\varepsilon,s}$, y $\mathbf{q}^h = \mathbf{q} + O(h^\gamma)$ uniformemente en $K_{\varepsilon,u}$. Más aún, para todo $\mathbf{x}_0 \in K_\varepsilon$ y $h \leq h_0$, existe $\mathbf{y}_0 = \mathbf{y}_0(\mathbf{x}_0, h) \in K_{\varepsilon_0}$ satisfaciendo

$$\sup \{ \|\mathbf{x}(t_n; \mathbf{x}_0) - \mathbf{y}_n(\mathbf{y}_0)\| : \mathbf{x}(t; \mathbf{x}_0) \in K_\varepsilon, \forall t \in [t_0, t_n] \} = O(h^\gamma).$$

Correspondientemente, para todo $\mathbf{y}_0 \in K_\varepsilon$ y $h \leq h_0$, existe $\mathbf{x}_0 = \mathbf{x}_0(\mathbf{y}_0, h) \in K_{\varepsilon_0}$ tal que

$$\sup \{ \|\mathbf{x}(t_n; \mathbf{x}_0) - \mathbf{y}_n(\mathbf{y}_0)\| : \mathbf{y}_i(\mathbf{x}_0) \in K_\varepsilon, \forall i = 0 \dots n \} = O(h^\gamma).$$

Este teorema muestra que el retrato de fases de un sistema dinámico en la vecindad de un punto estacionario hiperbólico es correctamente reproducido por los métodos LL. En particular, las variedades estables e inestables de las discretizaciones LL convergen a sus contrapartes continuas con un error de orden γ .

2.4 Esquemas de Linealización Local

Como puede notarse de sus respectivas definiciones, las aproximaciones Lineales Locales no pueden ser implementadas directamente por estar constituidas por integrales y EDO que en general no pueden ser calculadas analíticamente. Dependiendo de la forma de calcularlas numéricamente, diferentes esquemas numéricos pueden ser obtenidos [16]. Más precisamente tenemos la siguiente definición:

Definición 6 Dada la discretización Lineal Local $\mathbf{y}_{n+1} = \mathbf{y}_n + F_\gamma(t_n, \mathbf{y}_n; h_n)$ de orden γ , toda fórmula recursiva de la forma

$$\tilde{\mathbf{y}}_{n+1} = \tilde{\mathbf{y}}_n + \tilde{F}_\gamma(t_n, \tilde{\mathbf{y}}_n; h_n), \quad \text{con } \tilde{\mathbf{y}}_0 = \mathbf{y}_0,$$

donde $\tilde{F}_\gamma$ denota una aproximación de F_γ mediante un algoritmo numérico, es llamado genéricamente esquema de Linealización Local.

Específicamente, si la fórmula de Padé con escalamiento y potenciación [24] es utilizada para aproximar ϕ en (3) se obtiene el siguiente esquema LL de orden 2 [16]:

$$\tilde{\mathbf{y}}_{n+1} = \tilde{\mathbf{y}}_n + \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; h_n), \quad (12)$$

donde $\tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; h_n) = \mathbf{L} (\mathbf{P}_{p,q}(2^{-k_n} \tilde{\mathbf{D}}_n h_n))^{2^{k_n}} \mathbf{r}$, $\mathbf{P}_{p,q}(2^{-k_n} \tilde{\mathbf{D}}_n h_n)$ es la aproximación de Padé de orden (p, q) de $e^{2^{-k_n} \tilde{\mathbf{D}}_n h_n}$, k_n es el menor entero tal que $\|2^{-k_n} \tilde{\mathbf{D}}_n h_n\| \leq \frac{1}{2}$, y las matrices $\tilde{\mathbf{D}}_n$, $\mathbf{L}$, $\mathbf{r}$ están definidas como

$$\tilde{\mathbf{D}}_n = \begin{bmatrix} \mathbf{f}_x(t_n, \tilde{\mathbf{y}}_n) & \mathbf{f}_t(t_n, \tilde{\mathbf{y}}_n) & \mathbf{f}(t_n, \tilde{\mathbf{y}}_n) \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{(d+2) \times (d+2)},$$

$\mathbf{L} = [\mathbf{I}_d \quad \mathbf{0}_{d \times 2}]$ y $\mathbf{r}^\top = [\mathbf{0}_{1 \times (d+1)} \quad 1]$ para EDO no autónomas; y

$$\tilde{\mathbf{D}}_n = \begin{bmatrix} \mathbf{f}_x(t_n, \tilde{\mathbf{y}}_n) & \mathbf{f}(t_n, \tilde{\mathbf{y}}_n) \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{(d+1) \times (d+1)},$$

$\mathbf{L} = \begin{bmatrix} \mathbf{I}_d & \mathbf{0}_{d \times 1} \end{bmatrix}$ y $\mathbf{r}^\top = \begin{bmatrix} \mathbf{0}_{1 \times d} & 1 \end{bmatrix}$ para ecuaciones autónomas.

Si se utiliza la fórmula de Padé para aproximar ϕ en (3), y la fórmula de Runge Kutta clásico de orden 4 para integrar la ecuación auxiliar (8) se obtiene el siguiente esquema LLRK de orden 4 [9]:

$$\tilde{\mathbf{y}}_{n+1} = \tilde{\mathbf{y}}_n + \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; h_n) + \tilde{\rho}(t_n, \tilde{\mathbf{y}}_n; h_n), \quad (13)$$

donde

$$\tilde{\rho}(t_n, \tilde{\mathbf{y}}_n; h_n) = \frac{h_n}{6}(2\tilde{\mathbf{k}}_2 + 2\tilde{\mathbf{k}}_3 + \tilde{\mathbf{k}}_4),$$

con

$$\begin{aligned} \tilde{\mathbf{k}}_i &= \mathbf{f}\left(t_n + c_i h_n, \tilde{\mathbf{y}}_n + \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; t_n + c_i h_n) + c_i h_n \tilde{\mathbf{k}}_{i-1}\right) - \mathbf{f}(t_n, \tilde{\mathbf{y}}_n) \\ &\quad - \mathbf{f}_x(t_n, \tilde{\mathbf{y}}_n) \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; t_n + c_i h_n) - \mathbf{f}_t(t_n, \tilde{\mathbf{y}}_n) c_i h_n \\ \tilde{\mathbf{k}}_1 &\equiv \mathbf{0}, \quad \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; t_n + \frac{h_n}{2}) = \mathbf{L}\mathbf{A}\mathbf{r}, \quad \tilde{\phi}(t_n, \tilde{\mathbf{y}}_n; t_n + h_n) = \mathbf{L}\mathbf{A}^2\mathbf{r}, \\ \mathbf{A} &= (\mathbf{P}_{q,q}(2^{-k_n} \tilde{\mathbf{D}}_n h_n))^{2^{k_n}}, \\ c &= \begin{bmatrix} 0 & \frac{1}{2} & \frac{1}{2} & 1 \end{bmatrix}. \end{aligned}$$

Las matrices $\tilde{\mathbf{D}}_n$, $\mathbf{L}$, $\mathbf{r}$ están definidas como en el esquema anterior.

Por comodidad, los esquemas (12) y (13) se denotarán como LL2 y LLRK4, respectivamente.

3 Esquemas LL adaptativos

En esta sección se introduce una estrategia adaptativa de selección de paso que resulta una adecuación del procedimiento descrito en [10] y [20] para tener en cuenta las particularidades de los esquemas LL.

3.1 Cambio de paso

Típicamente, para implementar algoritmos de paso variable, es necesario calcular en cada iteración un tamaño de paso h adecuado en base a una estimación del error E cometido en la integración. Para los esquemas LL se propone la siguiente estrategia.

Dada la aproximación $\mathbf{y}_{n-1}$ en el tiempo t_{n-1} y un tamaño de paso $h \geq prs$, donde $prs = 10^{-15}$ es el menor tamaño de paso permisible definido en términos de la precisión de la aritmética,

1. Calcular la solución en $t_n = t_{n-1} + 2h$ con dos pasos consecutivos de tamaño h , que denotaremos $\mathbf{y}_n$.
2. Calcular la solución en t_n con un paso de tamaño $2h$, que llamaremos $\hat{\mathbf{y}}_n$.
3. Evaluar el error mediante la fórmula

$$E = \sqrt{\frac{1}{d} \sum_{i=1}^d \left(\frac{\mathbf{y}_n^i - \hat{\mathbf{y}}_n^i}{\mathbf{sc}^i} \right)^2}, \quad (14)$$

donde

$$\mathbf{sc}^i = AbsTol + RelTol \cdot \max\{|\mathbf{y}_{n-1}^i|, |\hat{\mathbf{y}}_n^i|\} \quad (15)$$

siendo $AbsTol$ y $RelTol$ la tolerancia absoluta y relativa, respectivamente.

4. Si $E \geq 1$ y $h > prs$ se rechaza el paso h . Entonces, se retorna al punto 1 con un nuevo tamaño de paso calculado mediante la expresión

$$h_{new} = \max\{prs, h \cdot \min\{fac_{\max}, \max\{fac_{\min}, fac \cdot \left(\frac{1}{E}\right)^{\frac{1}{\gamma+1}}\}\}\} \quad (16)$$

donde $fac_{\max} = 1$, $fac_{\min} = 0.1$, $fac = 0.25$ y γ el orden de convergencia del método. De lo contrario se admite el paso h y se acepta la solución $\mathbf{y}_n$ en el punto t_n , pasando a la siguiente iteración con un nuevo tamaño de paso calculado mediante la expresión (16) pero con $fac_{\max} = 5$, $fac_{\min} = 0.25$, $fac = 0.8$.

3.2 Paso inicial

En los esquemas adaptativos es fundamental implementar la selección de un tamaño de paso inicial, que cuando menos, no permita una elección totalmente incorrecta. Aunque en cualquier caso una mala elección puede ser reparada con el algoritmo de cambio de paso, esto supondría iteraciones extras y por tanto mayor costo computacional. Por tanto se propone el siguiente algoritmo:

1. Evaluar $\mathbf{f}(t_0, \mathbf{y}_0)$ y $\mathbf{f}_x(t_0, \mathbf{y}_0)$.
2. Calcular $d_0 = \|\mathbf{y}_0\|$ y $d_1 = \|\mathbf{f}(t_0, \mathbf{y}_0)\|$, donde $\|\mathbf{x}\| = \sqrt{\frac{1}{d} \sum_{i=1}^d \left(\frac{\mathbf{x}^i}{\mathbf{sc}^i} \right)^2}$.
3. Calcular la segunda derivada de y , mediante la fórmula

$$\frac{d^2 \mathbf{y}}{dt^2}(t_0, \mathbf{y}_0) = \frac{\partial \mathbf{f}}{\partial t}(t_0, \mathbf{y}_0) + \mathbf{f}_x(t_0, \mathbf{y}_0) \cdot \mathbf{f}(t_0, \mathbf{y}_0) \quad (17)$$

4. Calcular $d_2 = \left\| \frac{d^2 \mathbf{y}}{dt^2}(t_0, \mathbf{y}_0) \right\|$.
5. Si d_0 o d_1 son menores que $10 \cdot AbsTol$, entonces $h_0 = AbsTol$, de lo contrario $h_0 = 0.01 \left(\frac{d_0}{d_1} \right)$.
6. Si $\max\{d_1, d_2\} \leq prs$, entonces $h_1 = \max\{AbsTol, h_0 \cdot RelTol\}$. Si no $h_1 = \left(\frac{0.01}{\max\{d_1, d_2\}} \right)^{\frac{1}{\gamma+1}}$.
7. Culmina tomando $h_{inic} = \min\{100h_0, h_1\}$.

3.3 Interpolación

Cuando necesitamos hallar la solución en una partición $(t)_\delta$ determinada de antemano, por lo general los integradores adaptativos siguen la siguiente estrategia:

1. Calculan la solución de forma adaptativa entre el punto inicial y final de la partición $(t)_\delta$, escogiendo la partición $(t)_h$ según la estrategia de cada integrador.
2. Hallan la solución en $(t)_\delta$ interpolando los puntos de la solución previamente calculada en el punto anterior.

Típicamente el procedimiento de interpolación se realiza mediante un polinomio que preserve el orden de convergencia del integrador. Esta estrategia tiene las debilidades propias de la interpolación polinomial, como por ejemplo posibles oscilaciones [19]. Por esta razón el esquema adaptativo LL utiliza su propia fórmula para interpolar, lo cual proporciona compatibilidad, estabilidad y es sumamente útil al evitar los defectos de la interpolación polinomial.

3.4 Diferencias con el procedimiento estándar

Las principales diferencias del algoritmo adaptativo que se propone con respecto a los procedimientos de [10] y [20] son:

1. En la propuesta de [10] para el paso inicial se aproxima la segunda derivada de la solución de la ecuación (1) mediante una iteración del algoritmo de Euler. En el caso de los esquemas LL, el jacobiano es evaluado en cada iteración, por lo tanto la segunda derivada puede ser calculada exactamente mediante la fórmula (17).
2. En [20] se propone utilizar extrapolación en el proceso de hallar el término sc , la cual no es usada en la fórmula (15).

3. Al detectar un tamaño de paso menor o igual a la precisión de la maquina, el algoritmo de control de tamaño de paso no solo envía una alerta al usuario como es habitual sino que permite la continuidad del proceso de integración con el menor tamaño de paso posible, es decir, $h = prs$. Esto es particularmente útil en la integración de ecuaciones con soluciones caóticas o altamente “stiff”.
4. La función de interpolación mencionada en la Sección 3.3.

4 Esquemas LL adaptativos con precisión de Padé disminuida

Los esquemas de Linealización Local, tanto el LL como el LLRK, utilizan el método de Padé con escalamiento y potenciación para evaluar la exponencial matricial necesaria para encontrar la solución de la ecuación linealizada. Precisamente es este procedimiento el que más recursos computacionales consume, por lo que sería de gran importancia encontrar una forma de disminuir ese costo.

4.1 Fórmula de Padé para la exponencial matricial

Veamos en qué consiste la aproximación racional de Padé de la función exponencial:

Definición 7 [11] La aproximación racional de Padé $\mathbf{P}_{p,q}(z)$ de e^z está dada por

$$\mathbf{P}_{p,q}(z) = \frac{\mathbf{R}_{p,q}(z)}{\mathbf{Q}_{p,q}(z)}$$

donde

$$\begin{aligned} \mathbf{R}_{p,q}(z) = & 1 + \frac{p}{p+q}z + \frac{p(p-1)}{(p+q)(p+q-1)} \frac{z^2}{2!} + \dots + \\ & + \frac{p(p-1)\dots 1}{(p+q)(p+q-1)\dots (q+1)} \frac{z^p}{p!} \end{aligned}$$

y

$$\mathbf{Q}_{p,q}(z) = \mathbf{R}_{q,p}(-z).$$

Definición 8 Sea $\mathbf{A}$ una matriz cuadrada, y sea $\mathbf{P}_{p,q}(2^{-k}\mathbf{A})$ la aproximación de Padé de orden (p, q) de $e^{2^{-k}\mathbf{A}}$, donde k es el menor entero tal que $|2^{-k}\mathbf{A}| \leq \frac{1}{2}$. Se define la aproximación de Padé con escalamiento y potenciación como

$$\mathbf{F}_{p,q}^k(\mathbf{A}) = (\mathbf{P}_{p,q}(2^{-k}\mathbf{A}))^{2^k}. \quad (18)$$

Para esta aproximación se tiene el teorema de convergencia:

Teorema 6 [24, 16] *El error relativo y absoluto de la aproximación de Padé con escalamiento y potenciación (18) de $e^{\mathbf{A}}$ está dado por:*

$$\frac{|e^{\mathbf{A}} - \mathbf{F}_{p,q}^k(\mathbf{A})|}{|e^{\mathbf{A}}|} \leq \epsilon_{p,q} |\mathbf{A}| e^{\epsilon_{p,q} |\mathbf{A}|}$$

y

$$|e^{\mathbf{A}} - \mathbf{F}_{p,q}^k(\mathbf{A})| \leq c_{p,q}(k, |\mathbf{A}|) |\mathbf{A}|^{p+q+1},$$

donde $\epsilon_{p,q} = \alpha(\frac{1}{2})^{p+q-3}$, $c_{p,q}(k, |\mathbf{A}|) = \alpha 2^{-k(p+q)+3} e^{(1+\epsilon_{p,q})|\mathbf{A}|}$ y $\alpha = \frac{p!q!}{(p+q)!(p+q+1)!}$.

4.2 Convergencia de los esquemas LL

La convergencia y estabilidad de los esquemas LL (12) y LLRK (13) basados en la fórmula de Padé (18) se presenta en los siguientes teoremas:

Teorema 7 [16] *Sea $\tilde{\phi}(t_n, y_n; h_n)$ la aproximación de ϕ mediante la aproximación de Padé de orden (p, q) con escalamiento y potenciación. Entonces bajo las condiciones generales de existencia y unicidad para EDO y condiciones generales de suavidad se tiene*

$$\|\phi(t_n, \tilde{y}_n; h_n) - \tilde{\phi}(t_n, \tilde{y}_n; h_n)\| = O(h^{p+q+1})$$

y

$$\|\mathbf{x}(t_n) - \tilde{\mathbf{y}}_n\| = O(h^{\min\{\gamma, p+q\}})$$

$\forall t_n \in (t)_h$, donde $\tilde{\mathbf{y}}_n$ denota indistintamente los esquemas LL (12) y LLRK (13), con $\gamma = 2$ y $\gamma = 4$, respectivamente.

En cuanto a la estabilidad de los esquemas LL para las ecuaciones lineales se tiene:

Teorema 8 [16, 9] *Los esquemas LL son A-estables si para la aproximación de Padé de orden (p, q) se toma $p \leq q \leq p+2$. Además, si $q = p+1$ o $q = p+2$, el método es L-estable.*

Los teoremas 7 y 8 relacionan directamente la convergencia y estabilidad de los esquemas LL con el orden (p, q) de la aproximación de Padé involucrada. Estos resultados sugieren la posibilidad de disminuir el costo computacional de los

esquemas LL sin afectar su estabilidad y orden de convergencia. En efecto, ello es posible seleccionando $p + q = \gamma$ con las restricciones del Teorema 8, lo cual conlleva a una pérdida de precisión en el cálculo de la exponencial matricial de la discretización LL y, consecuentemente, a un posible deterioro de la precisión y propiedades dinámicas de los esquemas LL. El balance entre el aumento de eficiencia computacional y la pérdida de precisión mencionada será objeto de análisis en la próxima sección.

5 Simulaciones

En esta sección se analiza, mediante simulaciones numéricas, el comportamiento de los códigos LL adaptativos conformados por los esquemas LL2 (12) y LLRK4 (13) con la estrategia adaptativa presentada en la Sección 3. Primeramente, se evaluará la precisión y costo computacional de los esquemas y luego su comportamiento dinámico.

5.1 Precisión y costo

Para estudiar la precisión y costo de los códigos adaptativos propuestos, se tomó un conjunto de ecuaciones de prueba [10, 11, 21] que se exponen a continuación.

5.1.1 Ecuaciones de prueba

Los primeros cuatro ejemplos de ecuaciones de prueba tienen la forma semi-lineal

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}\mathbf{x} + \mathbf{f}(\mathbf{x}), \quad (19)$$

donde $\mathbf{A}$ es una matriz cuadrada, y $\mathbf{f}$ es una función de $\mathbf{x}$. El campo vectorial de los dos primeros ejemplos tienen jacobianos con valores propios cercanos o sobre el eje imaginario, lo que hace a estos osciladores difíciles de integrar por los esquemas convencionales [21]. Los otros dos ejemplos presentan también dificultades en su integración al ser ecuaciones de tipo “stiff” [21]. La cuarta ecuación posee una complejidad adicional al tener su parte lineal un jacobiano con valores propios positivos, lo cual puede afectar la integración en una vecindad del punto de equilibrio estable $\mathbf{x} = 1$. Este problema es particularmente relevante para los integradores convencionales que no actualizan el Jacobiano en cada paso de integración [21].

Ejemplo 1 Periódico lineal:

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}(\mathbf{x} + 2),$$

donde

$$\mathbf{A} = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix},$$

$$\mathbf{x}(t_0) = (-2.5, -1.5) \text{ y } [t_0, T] = [0, 4\pi].$$

Ejemplo 2 *Periódico lineal más parte no lineal:*

$$\frac{d\mathbf{x}}{dt} = \mathbf{A}(\mathbf{x} + 2) + 0.1\mathbf{x}^2,$$

donde la matriz $\mathbf{A}$ es la definida en el ejemplo anterior, $\mathbf{x}(t_0) = (1, 1)$, y $[t_0, T] = [0, 4\pi]$.

Ejemplo 3 *Ecuación “stiff” lineal:*

$$\frac{d\mathbf{x}}{dt} = -100\mathbf{H}(\mathbf{x} + 1),$$

donde $\mathbf{H}$ es la matriz de Hilbert de dimensión 12 (con número de condición 1.69×10^{16}), $\mathbf{x}^i(t_0) = 1$, $i = 1 \dots 12$, y $[t_0, T] = [0, 1]$.

Ejemplo 4 *Lineal “stiff” con parte no lineal:*

$$\frac{d\mathbf{x}}{dt} = 100\mathbf{H}(\mathbf{x} - 1) + 100(\mathbf{x} - 1)^2 - 60(\mathbf{x}^3 - 1),$$

donde $\mathbf{H}$ es la matriz de Hilbert de dimensión 12, $\mathbf{x}^i(t_0) = -0.5$, $i = 1 \dots 12$, y $[t_0, T] = [0, 1]$.

Los siguiente ejemplos son ecuaciones no lineales que se utilizan comúnmente en la prueba de integradores de EDO [21].

Ejemplo 5 *Ecuación de Brusselator [10]:*

$$\begin{aligned} \frac{dx_1}{dt} &= 1 + x_1^2 x_2 - 4x_1 \\ \frac{dx_2}{dt} &= 3x_1 - x_1^2 x_2 \end{aligned}$$

donde $\mathbf{x}(t_0) = (1.5, 3)$ y $[t_0, T] = [0, 20]$. Este es un ejemplo de ecuación no “stiff”.

Ejemplo 6 Ecuación de Van der Pol [11, 21]:

$$\begin{aligned}\frac{dx_1}{dt} &= x_2 \\ \frac{dx_2}{dt} &= \varepsilon(1 - x_2^2)x_1 - x_2\end{aligned}$$

con condiciones iniciales $\mathbf{x}(t_0) = (2, 0)$. Con los parámetros $\varepsilon = 100$ y $\varepsilon = 1$ se obtienen ecuaciones “stiff” y no “stiff”, respectivamente, las cuales se integran sobre los intervalos $[t_0, T] = [0, 300]$ y $[t_0, T] = [0, 10]$.

Ejemplo 7 Ecuación de reacción química [21]:

$$\begin{aligned}\frac{dx_1}{dt} &= 1.3(x_3 - x_1) + 10400k(x_1)x_2 \\ \frac{dx_2}{dt} &= 1880(x_4 - x_2(1 + k(x_1))) \\ \frac{dx_3}{dt} &= 1752 - 269x_3 + 267x_1 \\ \frac{dx_4}{dt} &= 0.1 + 320x_2 - 321x_4\end{aligned}$$

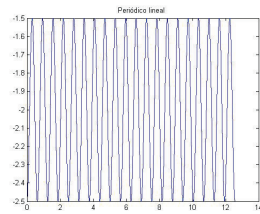
donde $k(x_1) = e^{(20.7 - \frac{1500}{x_1})}$. Con la condición inicial $\mathbf{x}(t_0) = (50, 0, 600, 0.1)$ en $[t_0, T] = [0, 1]$ es un ejemplo de ecuación moderadamente “stiff”.

Ejemplo 8 Ecuación de “Cuerpo Rígido” [10, 21]:

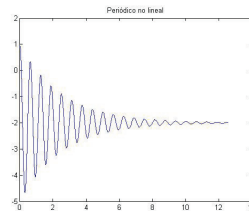
$$\begin{aligned}\frac{dx_1}{dt} &= x_2x_3 \\ \frac{dx_2}{dt} &= -x_1x_3 \\ \frac{dx_3}{dt} &= -0.51x_1x_2\end{aligned}$$

con condición inicial $\mathbf{x}(t_0) = (0, 1, 1)$ con $[t_0, T] = [0, 12]$.

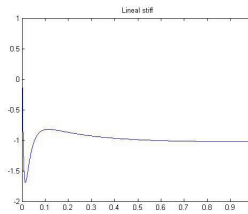
La figura 1 muestra la primera componente de la solución de cada uno de los ejemplos considerados.



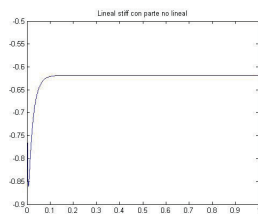
(a) Periódico lineal.



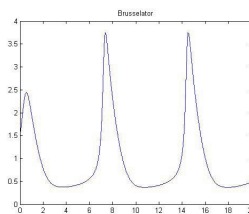
(b) Periódico no lineal.



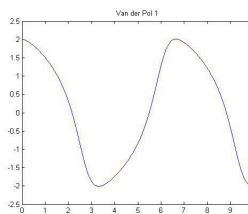
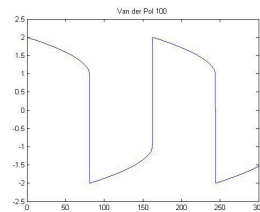
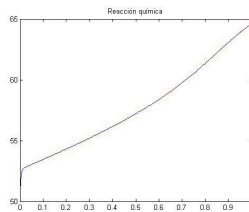
(c) Stiff lineal.



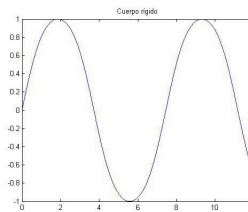
(d) Stiff no lineal.



(e) Brusselator.

(f) Van der Pol $\varepsilon = 1$.(g) Van der Pol $\varepsilon = 1000$.

(h) Reacción química.



(i) Cuerpo rígido.

Figura 1: Gráfico de la primera componente de las ecuaciones de prueba.

5.1.2 Pruebas realizadas

En las simulaciones se estudió:

1. La precisión de los códigos LL2 y LLRK4 adaptativos mediante el error relativo definido por la fórmula

$$ER = \max_{i=1, \dots, d; t_n \in (t)_h} \left\{ \left| \frac{\mathbf{x}^i(t_n) - \mathbf{y}^i(t_n)}{\mathbf{x}^i(t_n)} \right| \right\}.$$

2. El tiempo de corrida.
3. El número de pasos (NP) en la integración total.
4. El número de pasos rechazados (PR).
5. La cantidad de evaluaciones de la función (f eval) y su jacobiano ($\partial f / \partial y$ eval).
6. El número de evaluaciones de la exponencial matricial (EM).

Estos resultados se compararon con los obtenidos por varios códigos de Matlab [21]. Con ese fin, se diferenció entre los métodos de orden de convergencia 2 y 4, comparando el esquema LL2 con los códigos *ode23s* y *ode23* basados en la implementación del método de Rosenbrock de orden 2 y la fórmula de Runge Kutta (2, 3) respectivamente; y el esquema LLRK4 con los códigos *ode45* y *ode15s* que implementan los esquemas Runge Kutta (4, 5) y la “Fórmula de diferenciación numérica” respectivamente. Se utilizó el código *ode15s* con la tolerancia que aparece por defecto en el Matlab ($RelTol = 10^{-3}$, $AbsTol = 10^{-6}$) para hallar una partición inicial utilizada para comparar los resultados de todos los integradores. Asimismo, el esquema *ode15s* con tolerancia $RelTol = AbsTol = 10^{-12}$ fue el seleccionado para aproximar la solución exacta $\mathbf{x}$ de cada ejemplo con alta precisión. Los códigos LL2 y LLRK4 adaptativos fueron implementados en Matlab 7.04 y la función “*expm*” fue utilizada para evaluar las exponenciales matriciales correspondientes. De esta forma en los esquemas adaptativos LL2 y LLRK4 se utilizaron aproximaciones de Padé de orden (6, 6).

Para analizar del efecto de la precisión de la aproximación de Padé sobre los esquemas LL se repitieron las simulaciones expuestas anteriormente, pero con ordenes diferentes. Para las pruebas de estos códigos se construyeron dos nuevas funciones Matlab que implementan eficientemente las fórmulas de Padé

de orden $(1, 1)$ y $(2, 2)$ para los esquemas adaptativos LL2 y LLRK4 respectivamente. De esta forma se reduce al mínimo posible el costo computacional de esos códigos sin afectar sus ordenes de convergencia (ver Teorema 7). Al elegir estos coeficientes también se mantiene la A-estabilidad de los integradores según el Teorema 8.

5.1.3 Resultados

La precisión y costo computacional en la integración de las ecuaciones semi-lineales de los ejemplos 1-4 se presentan en las tablas 1 y 2 para los códigos de orden 2 y en las tablas 3 y 4 para los códigos de orden 4.

Los resultados de las tablas 1 - 4 muestran que para las ecuaciones semi-lineales de la forma (19):

1. Los códigos LL2(6, 6) y LLRK4(6, 6) tienen menor error relativo que los restantes códigos de orden similar, siendo el código LLRK4(6, 6) el más preciso de todos.
2. Para las ecuaciones lineales, el esfuerzo computacional de los dos códigos LL2 es mucho menor que el de los restantes. Los códigos LLRK4 tienen un costo computacional menor que el *ode15s* y el *ode45* en las ecuaciones lineales, pero mayor en las no lineales.
3. Para las ecuaciones con término no lineal, los códigos LL2 precisan de mayor cantidad de evaluaciones del jacobiano que el *ode23s*, sin embargo ahorran evaluaciones de la función. En comparación con el código *ode23*, los esquemas LL2 requieren de mayor esfuerzo computacional pero esto es compensado por una favorable diferencia entre sus precisiones.
4. Al disminuir la precisión de la aproximación racional de Padé utilizada en ambos códigos se observa un aumento en el error relativo en los ejemplos lineales, sin embargo en los no lineales este se mantiene similar, con número de pasos muy próximo en ambos casos.

Ejemplo	Código	RelTol	AbsTol	NP	Error Relativo
1 : Periódico lineal	<i>ode23s</i>	4×10^{-4}	4×10^{-7}	481	9.20×10^{-2}
	<i>ode23</i>	2×10^{-4}	2×10^{-7}	485	0.20
	<i>LL2</i> (6, 6)	10^{-12}	10^{-15}	10	4.06×10^{-9}
	<i>LL2</i> (1, 1)	10^{-12}	10^{-15}	10	0.18
2 : Periódico lineal con parte no lineal	<i>ode23s</i>	10^{-3}	10^{-5}	297	0.24
	<i>ode23</i>	6×10^{-4}	7×10^{-7}	279	0.12
	<i>LL2</i> (6, 6)	5×10^{-7}	4.9×10^{-5}	279	2.77×10^{-3}
	<i>LL2</i> (1, 1)	5×10^{-7}	4.9×10^{-5}	280	5.60×10^{-2}
3 : Lineal stiff	<i>ode23s</i>	3×10^{-4}	3×10^{-7}	76	3.15×10^{-2}
	<i>ode23</i>	4×10^{-3}	4×10^{-6}	82	0.11
	<i>LL2</i> (6, 6)	10^{-4}	10^{-6}	6	1.16×10^{-9}
	<i>LL2</i> (1, 1)	10^{-4}	10^{-6}	10	4.55×10^{-2}
4 : Lineal stiff con parte no lineal	<i>ode23s</i>	10^{-4}	10^{-5}	71	7.51×10^{-2}
	<i>ode23</i>	10^{-2}	10^{-3}	134	2.71
	<i>LL2</i> (6, 6)	10^{-5}	5×10^{-6}	63	2.93×10^{-2}
	<i>LL2</i> (1, 1)	10^{-5}	5×10^{-6}	63	8.75×10^{-2}

Tabla 1: Errores de los códigos de orden 2 en la integración de los ejemplos 1-4.

Ejemplo	Código	Tiempo	NP	PR	f eval	$\partial f/\partial y$ eval	LU	EL	EM
1	<i>ode23s</i>	0.63	481	0	2407	481	481	1443	0
	<i>ode23</i>	0.18	485	0	1456	0	0	0	0
	<i>LL2</i> (6, 6)	0.03	10	0	22	22	0	0	21
	<i>LL2</i> (1, 1)	0.02	10	0	22	22	0	0	21
2	<i>ode23s</i>	0.41	297	0	1487	297	297	891	0
	<i>ode23</i>	0.11	279	0	838	0	0	0	0
	<i>LL2</i> (6, 6)	0.33	279	0	560	560	0	0	559
	<i>LL2</i> (1, 1)	0.23	280	0	562	562	0	0	561
3	<i>ode23s</i>	0.26	76	7	1156	76	83	249	0
	<i>ode23</i>	0.04	82	4	259	0	0	0	0
	<i>LL2</i> (6, 6)	0.03	6	0	14	14	0	0	13
	<i>LL2</i> (1, 1)	0.02	10	4	30	30	0	0	29
4	<i>ode23s</i>	0.21	71	0	1067	71	71	213	0
	<i>ode23</i>	0.06	134	4	415	0	0	0	0
	<i>LL2</i> (6, 6)	0.09	63	1	130	130	0	0	129
	<i>LL2</i> (1, 1)	0.08	63	1	130	130	0	0	129

Tabla 2: Costo computacional de los códigos de orden 2 en la integración de los ejemplos 1-4 con la precisión mostrada en la Tabla 1. LU y EL denotan, respectivamente, el número de descomposiciones matriciales LU realizadas y de sistemas lineales algebraicos resueltos.

Ejemplo	Código	RelTol	AbsTol	NP	Error Relativo
1 : Periódico lineal	<i>ode15s</i>	10^{-4}	10^{-7}	471	1.59×10^{-5}
	<i>ode45</i>	10^{-6}	10^{-8}	468	2.16×10^{-2}
	<i>LLRK4</i> (6, 6)	10^{-4}	10^{-5}	5	4.06×10^{-9}
	<i>LLRK4</i> (2, 2)	10^{-4}	10^{-5}	5	3.01×10^{-4}
2 : Periódico lineal con parte no lineal	<i>ode15s</i>	10^{-3}	10^{-5}	286	0.30
	<i>ode45</i>	7×10^{-6}	4×10^{-7}	279	1.55×10^{-4}
	<i>LLRK4</i> (6, 6)	1.3×10^{-8}	5×10^{-8}	279	1.18×10^{-7}
	<i>LLRK4</i> (2, 2)	1.3×10^{-8}	5×10^{-8}	279	2.46×10^{-6}
3 : Lineal stiff	<i>ode15s</i>	10^{-4}	10^{-6}	70	2.28×10^{-2}
	<i>ode45</i>	10^{-4}	10^{-5}	68	1.96×10^{-3}
	<i>LLRK4</i> (6, 6)	10^{-7}	10^{-9}	6	9.54×10^{-9}
	<i>LLRK4</i> (2, 2)	10^{-7}	10^{-9}	7	1.39×10^{-5}
4 : Lineal stiff con parte no lineal	<i>ode15s</i>	10^{-3}	10^{-3}	91	6.10×10^{-2}
	<i>ode45</i>	10^{-2}	10^{-3}	105	0.13
	<i>LLRK4</i> (6, 6)	10^{-8}	2×10^{-7}	52	3.12×10^{-6}
	<i>LLRK4</i> (2, 2)	10^{-8}	2×10^{-7}	52	4.76×10^{-6}

Tabla 3: Errores de los códigos de orden 4 en la integración de los ejemplos 1-4.

Ejemplo	Códigos	Tiempo	NP	PR	f_{eval}	$\partial f/\partial y_{\text{eval}}$	LU	EL	EM
1	<i>ode15s</i>	0.59	471	0	946	1	11	942	0
	<i>ode45</i>	0.27	468	0	2809	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.01	5	0	60	12	0	0	11
	<i>LLRK4</i> (2, 2)	0.01	5	0	60	12	0	0	11
2	<i>ode15s</i>	0.24	286	3	419	1	48	415	0
	<i>ode45</i>	0.15	279	0	1675	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.49	279	0	3074	560	0	0	559
	<i>LLRK4</i> (2, 2)	0.42	279	0	3074	560	0	0	559
3	<i>ode15s</i>	0.07	70	2	101	1	15	87	0
	<i>ode45</i>	0.05	68	1	415	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.01	6	0	60	12	0	0	11
	<i>LLRK4</i> (2, 2)	0.01	7	1	93	18	0	0	17
4	<i>ode15s</i>	0.06	91	1	193	2	19	166	0
	<i>ode45</i>	0.09	105	0	631	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.12	52	1	588	108	0	0	107
	<i>LLRK4</i> (2, 2)	0.09	52	1	588	108	0	0	107

Tabla 4: Costo computacional de los códigos de orden 4 en la integración de los ejemplos 1-4 con la precisión mostrada en la Tabla 3. LU y EL denotan, respectivamente, el número de descomposiciones matriciales LU realizadas y de sistemas lineales algebraicos resueltos.

Estos resultados se explican por las siguientes razones:

1. La dinámica de las ecuaciones semi-lineales depende principalmente de la parte lineal del campo vectorial.
2. Los métodos LL preservan la estabilidad de cualquier sistema lineal para todos los tamaños de paso, lo cual no cumplen los integradores explícitos convencionales.
3. Los códigos LL2(6, 6) y LLRK4(6, 6) integran “exactamente” (con la precisión de la aritmética) las ecuaciones lineales.
4. Los esquemas LL actualizan el jacobiano del campo vectorial en cada paso de integración, lo que no sucede en la mayoría de los integradores.
5. La componente “stiff” de las ecuaciones de los ejemplos 3 y 4 no está incluida en la ecuación auxiliar asociada a los esquemas LLRK4, por lo que pueden ser fácilmente integrada por el código Runge Kutta de dichos esquemas.

Ejemplo	Código	RelTol	AbsTol	NP	Error Relativo
5 : Brusselator	<i>ode23s</i>	1.2×10^{-4}	10^{-7}	247	2.16×10^{-2}
	<i>ode23</i>	5×10^{-5}	5×10^{-7}	245	4.31×10^{-3}
	LL2(6, 6)	1×10^{-4}	4.2×10^{-5}	246	3.70×10^{-2}
	LL2(1, 1)	1×10^{-4}	4.2×10^{-5}	247	5.12×10^{-2}
6 : Van der Pol $\varepsilon = 1$	<i>ode23s</i>	10^{-6}	10^{-6}	729	8.59×10^{-3}
	<i>ode23</i>	10^{-6}	10^{-7}	720	5.34×10^{-5}
	LL2(6, 6)	3×10^{-6}	10^{-7}	700	3.36×10^{-3}
	LL2(1, 1)	3×10^{-6}	10^{-7}	700	1.98×10^{-3}
6 : Van der Pol $\varepsilon = 100$	<i>ode23s</i>	1.3×10^{-7}	1.3×10^{-7}	6151	1.92×10^{-2}
	<i>ode23</i>	5×10^{-4}	2×10^{-8}	22431	1.23
	LL2(6, 6)	10^{-7}	5×10^{-8}	6187	3.04×10^{-2}
	LL2(1, 1)	10^{-7}	5×10^{-8}	6183	2.92×10^{-2}
7 : Reacción química	<i>ode23s</i>	10^{-4}	10^{-7}	116	7.11×10^{-4}
	<i>ode23</i>	10^{-2}	10^{-3}	888	0.1293
	LL2(6, 6)	10^{-6}	2×10^{-7}	103	1.00×10^{-4}
	LL2(1, 1)	10^{-6}	2×10^{-7}	106	3.60×10^{-4}
8 : Cuerpo Rígido	<i>ode23s</i>	1.4×10^{-4}	10^{-7}	110	0.25
	<i>ode23</i>	10^{-4}	2×10^{-7}	111	0.01
	LL2(6, 6)	10^{-4}	10^{-6}	110	0.19
	LL2(1, 1)	10^{-4}	10^{-6}	110	0.46

Tabla 5: Errores de los códigos de orden 2 en la integración de los ejemplos 5-8.

La precisión y costo computacional en la integración de las ecuaciones no lineales de los ejemplos 5-8 se presentan en las tablas 5 y 6 para los códigos de orden 2, y en las tablas 7 y 8 para los códigos de orden 4.

Ejemplo	Código	Tiempo	NP	PR	f eval	$\partial f/\partial y$ eval	LU	EL	EM
5	<i>ode23s</i>	0.26	247	7	1251	247	254	762	0
	<i>ode23</i>	0.07	245	10	766	0	0	0	0
	<i>LL2</i> (6, 6)	0.16	246	18	530	530	0	0	529
	<i>LL2</i> (1, 1)	0.12	247	19	534	534	0	0	533
6 $\varepsilon = 1$	<i>ode23s</i>	0.72	729	0	3647	729	729	2187	0
	<i>ode23</i>	0.15	720	1	2164	0	0	0	0
	<i>LL2</i> (6, 6)	0.37	700	13	1426	1426	0	0	1425
	<i>LL2</i> (1, 1)	0.27	700	13	1426	1426	0	0	1425
6 $\varepsilon = 100$	<i>ode23s</i>	6.37	6151	0	30757	6151	6151	18453	0
	<i>ode23</i>	4.39	22431	43	22431	0	0	0	0
	<i>LL2</i> (6, 6)	6.51	6187	7	12390	12390	0	0	12389
	<i>LL2</i> (1, 1)	5.38	6183	6	12380	12380	0	0	12379
7	<i>ode23s</i>	0.14	116	1	817	116	117	351	0
	<i>ode23</i>	0.17	888	6	2683	0	0	0	0
	<i>LL2</i> (6, 6)	0.11	103	1	210	210	0	0	209
	<i>LL2</i> (1, 1)	0.09	106	4	222	222	0	0	221
8	<i>ode23s</i>	0.13	110	5	672	110	115	345	0
	<i>ode23</i>	0.03	111	8	358	0	0	0	0
	<i>LL2</i> (6, 6)	0.09	110	9	240	240	0	0	239
	<i>LL2</i> (1, 1)	0.07	110	9	240	240	0	0	239

Tabla 6: Costo computacional de los códigos de orden 2 en la integración de los ejemplos 5-8 con la precisión mostrada en la Tabla 5. LU y EL denotan, respectivamente, el número de descomposiciones matriciales LU realizadas y de sistemas lineales algebraicos resueltos.

Los resultados de las tablas 5-8 muestran que para las ecuaciones no lineales de los ejemplos (5-8):

1. Los códigos LL2 muestran una precisión aceptable, siendo notable el caso de la ecuación de la reacción química (ejemplo 7) donde estos tienen un menor error relativo con menor número de pasos y esfuerzo computacional que los restantes esquemas (incluyendo a los métodos de orden 4).
2. Los códigos LL2 tienen por lo general menor costo computacional que el *ode23s*.

Ejemplo	Código	RelTol	AbsTol	NP	Error Relativo
5 : Brusselator	<i>ode15s</i>	6×10^{-5}	1×10^{-6}	254	6.52×10^{-3}
	<i>ode45</i>	4×10^{-8}	4×10^{-8}	257	1.00×10^{-6}
	<i>LLRK4</i> (6, 6)	7.5×10^{-7}	10^{-8}	250	7.28×10^{-7}
	<i>LLRK4</i> (2, 2)	7.5×10^{-7}	10^{-8}	250	7.18×10^{-7}
6 : Van der Pol $\varepsilon = 1$	<i>ode15s</i>	10^{-5}	10^{-5}	211	9.63×10^{-3}
	<i>ode45</i>	10^{-8}	10^{-8}	215	1.64×10^{-6}
	<i>LLRK4</i> (6, 6)	4×10^{-8}	10^{-10}	205	6.05×10^{-6}
	<i>LLRK4</i> (2, 2)	4×10^{-8}	10^{-10}	205	6.32×10^{-6}
6 : Van der Pol $\varepsilon = 100$	<i>ode15s</i>	1.65×10^{-10}	1.75×10^{-10}	6106	9.00×10^{-5}
	<i>ode45</i>	2×10^{-4}	2×10^{-4}	16918	0.47
	<i>LLRK4</i> (6, 6)	10^{-7}	10^{-7}	6101	1.46×10^{-4}
	<i>LLRK4</i> (2, 2)	10^{-7}	10^{-7}	6101	1.46×10^{-4}
7 : Reacción química	<i>ode15s</i>	10^{-4}	10^{-7}	116	1.85×10^{-3}
	<i>ode45</i>	10^{-2}	5×10^{-3}	675	0.29
	<i>LLRK4</i> (6, 6)	10^{-5}	5×10^{-7}	109	2.49×10^{-2}
	<i>LLRK4</i> (2, 2)	10^{-5}	5×10^{-7}	109	2.49×10^{-2}
8 : Cuerpo Rígido	<i>ode15s</i>	4×10^{-5}	4×10^{-6}	117	0.14
	<i>ode45</i>	4×10^{-8}	10^{-8}	113	1.66×10^{-5}
	<i>LLRK4</i> (6, 6)	10^{-7}	10^{-8}	110	1.57×10^{-6}
	<i>LLRK4</i> (2, 2)	10^{-7}	10^{-8}	110	2.80×10^{-6}

Tabla 7: Errores de los códigos de orden 4 en la integración de los ejemplos 5-8.

Ejemplo	Código	Tiempo	NP	PR	f eval	$\partial f/\partial y$ eval	LU	EL	EM
5	<i>ode15s</i>	0.25	254	40	536	2	73	529	0
	<i>ode45</i>	0.08	257	9	1597	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.21	250	1	2766	504	0	0	503
	<i>LLRK4</i> (2, 2)	0.18	250	1	2766	504	0	0	503
$\varepsilon = 1$	<i>ode15s</i>	0.18	211	1	376	1	44	372	0
	<i>ode45</i>	0.07	215	6	1327	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.15	205	0	2260	412	0	0	411
	<i>LLRK4</i> (2, 2)	0.13	205	0	2260	412	0	0	411
$\varepsilon = 100$	<i>ode15s</i>	4.44	6106	224	9901	12	558	9864	0
	<i>ode45</i>	4.60	16918	1109	108163	0	0	0	0
	<i>LLRK4</i> (6, 6)	9.19	6101	3	67149	12210	0	0	12209
	<i>LLRK4</i> (2, 2)	8.27	6101	3	67149	12210	0	0	12209
7	<i>ode15s</i>	0.10	116	10	183	1	30	176	0
	<i>ode45</i>	0.18	675	46	4327	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.14	109	2	1226	224	0	0	223
	<i>LLRK4</i> (2, 2)	0.12	109	2	1226	224	0	0	223
8	<i>ode15s</i>	0.10	117	8	205	1	26	200	0
	<i>ode45</i>	0.04	113	0	679	0	0	0	0
	<i>LLRK4</i> (6, 6)	0.09	110	0	1215	222	0	0	221
	<i>LLRK4</i> (2, 2)	0.08	110	0	1215	222	0	0	221

Tabla 8: Costo computacional de los códigos de orden 4 en la integración de los ejemplos 5-8 con la precisión mostrada en la tabla 7. LU y EL denotan, respectivamente, el número de descomposiciones matriciales LU realizadas y de sistemas lineales algebraicos resueltos.

3. Los dos códigos LLRK4 son más precisos en todas las ecuaciones excepto en la 7 que, al ser del tipo “stiff”, es mejor integrada por el código *ode15s*. En esta ecuación es notable la mejoría que le confiere la Linealización Local al algoritmo de Runge Kutta clásico.
4. En la ecuación de Van der Pol con el parámetro $\varepsilon = 100$, la precisión y el costo computacional del código *ode15s*, diseñado especialmente para este tipo de problemas, es mejorado por los esquemas LLRK4.
5. Los códigos LLRK4 son los más confiables en el sentido de tener un aceptablemente bajo error relativo en todos los ejemplos.
6. Al disminuir la precisión del algoritmo de Padé, el error relativo y el número de pasos de los códigos LL no varía significativamente en ninguno de los ejemplos.
7. El esfuerzo computacional de los códigos LLRK4 es mayor que el de Runge Kutta (4, 5).

Este último resultado desfavorable es causado por la necesidad de evaluaciones adicionales del campo vectorial y su jacobiano para estimar el error en cada paso de integración. Esta desventaja puede ser fácilmente superada por el uso de métodos anidados (embedding methods).

5.2 Propiedades dinámicas

En esta sección se estudia el efecto de variar la precisión de la aproximación racional de Padé sobre las propiedades dinámicas de los esquemas LL. Para ello se estudió el comportamiento de diferentes esquemas LL en la vecindad de los puntos de equilibrio de la siguiente ecuación [1].

Ejemplo 9

$$\begin{aligned}\frac{dx_1}{dt} &= -2x_1 + x_2 + 1 - \mu f(x_1, \lambda) \\ \frac{dx_2}{dt} &= x_1 - 2x_2 + 1 - \mu f(x_2, \lambda)\end{aligned}$$

donde $f(u, \lambda) = u(1 + u + \lambda u^2)^{-1}$.

Para $\mu = 15$ y $\lambda = 57$ este sistema tiene, en la región $0 \leq x_1, x_2 \leq 1$, tres puntos de equilibrio: dos estables y uno inestable. El punto de equilibrio

inestable tiene una variedad estable M_s de dimensión 1 que delimita el área de atracción de los puntos de equilibrio estables (ver figura 2). Esta variedad estable puede ser determinada numéricamente por cada integrador mediante el método de bisección. Para cada integrador, las tablas 9 y 10 muestran los valores de la intersección de su aproximación M_s^h con el eje x_2 para diferentes valores de h . Las tablas también muestran el estimado r_h del orden de convergencia de cada integrador calculado mediante la fórmula:

$$r_h = \frac{1}{\ln 2} \ln\left(\frac{\xi_h - \xi_{h/2}}{\xi_{h/2} - \xi_{h/4}}\right).$$

El comportamiento asintótico esperado para los valores de r_h se ha estudiado para los esquemas LL2(6, 6) y LLRK4(6, 6) en [15] y [9] y para los esquemas de Runge Kutta en [1]. Estos valores están dados por $r_h \approx 4$ para los métodos de orden de convergencia cuatro y por $r_h \approx 2$ para los de orden dos.

La figura 2 muestra el retrato de fases de la ecuación del ejemplo 9 para los integradores RK23, RK45, LL2(6, 6), LLRK4(6, 6), LL2(1, 1) y LLRK4(2, 2) con tamaño de paso $h = 2^{-2}$. La solución “exacta” fue hallada con el código LLRK4(6, 6), con tamaño de paso $h = 2^{-13}$. Aquí los nombres RK23 y RK45 denotan la implementación con paso fijo de los códigos *ode23* y *ode45*.

La figura 2 y las tablas 9 y 10 muestran que:

1. Los esquemas de Runge Kutta fallan al intentar reproducir el retrato de fases, especialmente cerca de uno de los puntos de atracción, mientras los códigos LL, logran una buena aproximación del mismo.
2. El código RK45, al ser de orden cuatro logra aproximar la intersección de la variedad estable con el eje x_2 mejor que los códigos LL2, sin embargo falla en la estimación de r_h .
3. Los códigos LL2 aproximan mejor el retrato de fases que el Runge Kutta (2, 3), integrador con el mismo orden de convergencia.
4. Los esquemas LLRK4 presentan una clara ventaja frente a los otros integradores al reproducir el retrato de fases y determinar la variedad estable del punto de equilibrio inestable.
5. Al disminuir el orden de la aproximación de Padé en los esquemas LL no se produce una diferencia significativa en el retrato de fases ni en la aproximación de ξ_h y r_h .

	<i>RK23</i>		<i>LL2(6, 6)</i>		<i>LL2(1, 1)</i>	
h	ξ_h	r_h	ξ_h	r_h	ξ_h	r_h
2^{-1}	0.3562		0.7191		0.7189	
2^{-2}	0.5002	2.46	0.6969	1.931	0.6964	1.965
2^{-3}	0.5754	3.38	0.6173	2.190	0.6163	2.205
2^{-4}	0.5890	3.44	0.5964	2.145	0.5958	2.590
2^{-5}	0.5903	3.25	0.5918	2.056	0.5914	2.547
2^{-6}	0.5904	3.12	0.5908	2.027	0.5906	1.941
2^{-7}	0.5905	3.06	0.5905	2.014	0.5905	1.972
2^{-8}	0.5905		0.5905		0.5905	

Tabla 9: Valores de ξ_h y r_h hallados mediante los códigos RK23, LL2(6, 6), LL2(1, 1) en la integración del ejemplo 9, para diferentes valores del tamaño de paso h .

	<i>RK45</i>		<i>LLRK4(6, 6)</i>		<i>LLRK4(2, 2)</i>	
h	ξ_h	r_h	ξ_h	r_h	ξ_h	r_h
2^{-1}	0.7038		0.5662		0.5662	
2^{-2}	0.5367	3.00	0.5844	5.142	0.5844	5.145
2^{-3}	0.5986	4.18	0.5903	2.384	0.5903	2.368
2^{-4}	0.5909	7.23	0.5905	3.354	0.5905	3.304
2^{-5}	0.5905	6.93	0.5905	3.901	0.5905	3.918
2^{-6}	0.5905	6.39	0.5905	3.973	0.5905	3.980
2^{-7}	0.5905	5.98	0.5905	3.989	0.5905	3.993
2^{-8}	0.5905		0.5905		0.5905	

Tabla 10: Valores de ξ_h y r_h hallados mediante los códigos RK45, LLRK4(6, 6), LLRK4(2, 2) en la integración del ejemplo 9, para diferentes valores del tamaño de paso h .

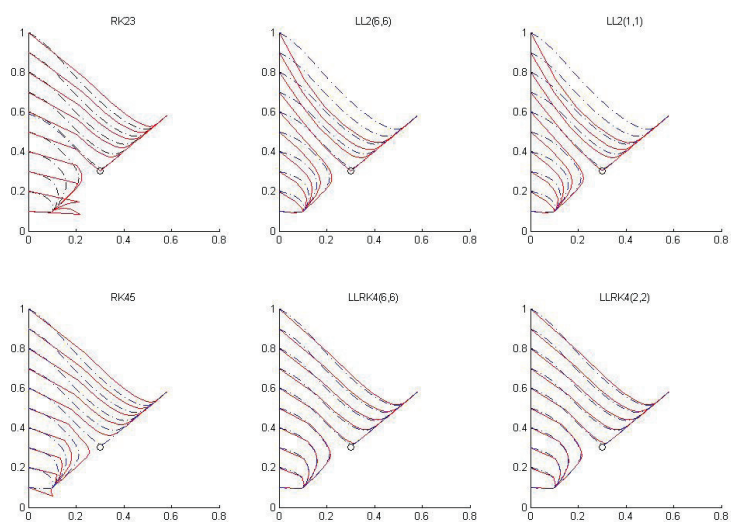


Figura 2: Retrato de fases de los códigos Matlab y LL con tamaño de paso $h = 2^{-2}$.

6 Conclusiones

En este trabajo se construyeron códigos adaptativos de esquemas de Linealización Local y de Linealización Local de Orden Superior. Se mostró, mediante simulaciones, que los códigos adaptativos propuestos integran la mayoría de las ecuaciones de prueba consideradas con mayor precisión y menor número de pasos que los principales códigos de Matlab. Se demostró que el esfuerzo de cómputo de los esquemas LL adaptativos puede ser notablemente disminuido al reducir el orden de la aproximación racional de Padé utilizada, sin afectar significativamente la precisión y propiedades dinámicas del integrador. En general, los códigos LLRK4 adaptivos son los más confiables en el sentido de tener un aceptablemente bajo error relativo en todos los ejemplos aunque su costo computacional es mayor que el de los esquemas Runge Kutta considerados. Estos resultados motivan el estudio de algoritmos adaptativos alternativos que permitan mantener la alta precisión de los esquemas LL mejorando su eficiencia computacional, como por ejemplo los algoritmos con esquemas “embedding” considerados en [18].

Referencias

- [1] Beyn, W.J. (1987) “On the numerical approximation of phase portraits near stationary points”, *SIAM J. Numer. Anal.* **24**: 1095–1113.
- [2] Carbonell, F.; Jimenez, J.C.; Biscay, R.J. (2006) “Weak local linear discretizations for stochastic differential equations: convergence and numerical schemes”, *J. Comput. Appl. Math.* **197**: 578–596.
- [3] Carbonell, F.; Jimenez, J.C.; Biscay, R.; de la Cruz, H. (2005) “The Local Linearization method for numerical integration of random differential equations”, *BIT Numerical Mathematics* **45**: 1–14.
- [4] Cartwright, J.H.E.; Piro, O. (1992) “The dynamics of Runge-Kutta methods”, *Int. J. Bifurc. Chaos* **2**: 427–449.
- [5] Cox, S.M.; Matthews, P.C. (2002) “Exponential time differencing for stiff systems”, *J. Comput. Phys.* **176**: 430–455.
- [6] de la Cruz, H.; Biscay, R.J.; Carbonell, F.; Jimenez, J.C.; Ozaki, T. (2006) “Local Linearization-Runge Kutta (LLRK) methods for solving ordinary differential equations”, in: *Lecture Notes in Computer Sciences 3991*, Springer-Verlag, Berlin: 132–139.

- [7] de la Cruz, H.; Biscay, R.J.; Carbonell, F.; Ozaki, T.; Jimenez, J.C. (2007) A higher order Local Linearization method for solving ordinary differential equations, *Appl. Math. Comput.* **185**: 197–212.
- [8] de la Cruz, H.; Biscay, R.J.; Jimenez, J.C.; Carbonell, F.; Ozaki, T. (2010) “High order local linearization methods: an approach for constructing A-stable high order explicit schemes for stochastic differential equations with additive noise”, *BIT Numerical Mathematics* **50**: 509–539.
- [9] de la Cruz H., Biscay R.J., Jimenez J.C. and Carbonell F., Local Linearization-Runge-Kutta methods: A class of A-stable explicit integrators for dynamical systems, *Math. Comput. Modell.*, 57 (2013) 720–740.
- [10] Hairer, E.; Norsett, S.P.; Wanner, G. (1993) *Solving Ordinary Differential Equations I*, 2nd ed. Springer-Verlag, Berlin.
- [11] Hairer, E.; Wanner, G. (1996) *Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems*, 3th ed. Springer-Verlag, Berlin.
- [12] Hochbruck, M.; Ostermann, A.; Schweitzer, J. (2009) “Exponential Rosenbrock type methods”, *SIAM J. Numer. Anal.* **47**: 786–803.
- [13] Hochbruck, M.; Ostermann, A. (2010) “Exponential integrators”, *Acta Numer.* **19**: 209–286.
- [14] Iserles, A. (1990) “Stability and dynamics of numerical methods for non-linear ordinary differential equations”, *IMA J. Numer. Anal.* **10**: 1–30.
- [15] Jimenez, J.C.; Biscay, R.; Mora, C.; Rodriguez, L.M. (2002) “Dynamic properties of the Local Linearization method for initial-value problems”, *Appl. Math. Comput.* **126**: 63–81.
- [16] Jimenez, J.C.; Carbonell, F. (2005) “Rate of convergence of local linearization schemes for initial-value problems”, *Appl. Math. Comput.* **171**: 1282–1295.
- [17] Jimenez, J.C.; Pedroso, L.; Carbonell, F.; Hernandez, V. (2006) “Local linearization method for numerical integration of delay differential equations”, *SIAM J. Numer. Analysis* **44**: 2584–2609.
- [18] Jimenez, J.C.; Sotolongo, A.; Sanchez-Bornot, J.M. (2012) Locally Linearized Runge Kutta method of Dormand and Prince, in: <http://arxiv.org/abs/1209.1415.pdf>

- [19] Prenter P.M. (1975) *Splines and Variational Methods*. John Wiley & Sons, New York.
- [20] Rahunanthana, A.; Stanescu, D. (2010) “High-order W-methods”, *Journal of Computational and Applied Mathematics* **233**: 1798–1811.
- [21] Shampine, L.F.; Reichelt, M.W. (1997) “The MATLAB ODE Suite” *SIAM Journal on Scientific Computing* **18**: 1–22.
- [22] Skufca, J.D. (2004) “Analysis still matters: a surprising instance of failure of Runge-Kutta-Felberg ODE solvers”, *SIAM Review* **46**: 729–737
- [23] Stewart, I. (1992) “Numerical methods: Warning-handle with care!”, *Nature* **355**: 16–17.
- [24] Van Loan, C.F.; Moler, C. (2003) “Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later”, *SIAM Review* **45**: 3–49.

