FRANZ CHOULY

APROXIMACIÓN NUMÉRICA DE ECUACIONES DIFERENCIALES ORDINARIAS

DIRAC

Aproximación numérica de ecuaciones diferenciales ordinarias

Franz Chouly

Chouly, Franz
Aproximación numérica de ecuaciones diferenciales ordinarias / Franz Chouly – Montevideo : DIRAC, 2026.

152 p. : il., cuadros.
ISBN (versión impresa): 978-9974-0-2373-4
ISBN (versión digital): 978-9974-0-2374-1

1. ANÁLISIS NUMÉRICO 2. ECUACIONES DIFERENCIALES ORDINARIAS

I. *Aproximación numérica de ecuaciones diferenciales ordinarias*

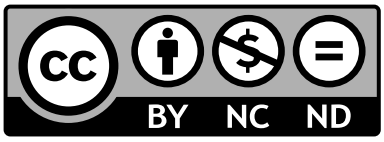
AMS-MSC2020 34-01

Los conceptos vertidos en los libros editados por la Facultad de Ciencias de la Universidad de la República son responsabilidad de sus autores. Su publicación no implica que sean compartidos por las mencionadas instituciones.

Edición y maquetación: Franz Chouly
Corrección y asistencia en la edición: Gabriel Santoro
Diseño de tapas: Rodolfo Fuentes RF[DS]
Figura de tapa: *The Lorenz attractor*, de Jos Leys, tomada de https://www.imaginary.org

Publicado por
DIRAC - Facultad de Ciencias - Universidad de la República
Iguá 4225 - Montevideo 11400 - Uruguay
Tel.: (+598) 2525.8618 int. 219
E-mail: dirac@fcien.edu.uy

© 2026 DIRAC - Facultad de Ciencias - Udelar



Esta obra está bajo una Licencia Creative Commons Atribución-NoComercial-SinDerivadas 4.0 Internacional.

Tabla de contenido

Agradecimientos

Quiero agradecer, primero, a los estudiantes de la Facultad de Ciencias que cursaron esta materia en 2025, por los errores que han señalado en los apuntes del curso, por sus sugerencias de mejora y por las conversaciones enriquecedoras que tuvimos acerca del tema. Agradezco también a los colegas del Centro de Matemática, del Instituto de Matemática y Estadística Rafael Laguardia, del IRL-2030 CNRS IFUMI y a mis cercanos (familia y amigos) por su apoyo. Agradezco, particularmente, a Juan Pablo Borthagaray, Ignacio Bustamente, Guillermo Bustos, Francisco Carballal, Alejandro Cholaquidis, Françoise Dal'Bo, Florence Hubert, José R. León, Sofía Llavayol, Roberto Markarian, Ernesto Mordecki, Alejandro Passeggi, Javier Peraza, Rafael Potrie, Martín Reiris, Martín Sambarino y Mauricio Velasco por el interés que mostraron, así como por los intercambios que tuvimos; y por lo mismo, a Roland Becker de la Universidad de Pau, a Stéphane Bordas de la Universidad de Luxemburgo y Martin Gander de la Universidad de Genève. Una mención también para los colegas que me introdujeron a los métodos para integración en tiempo, particularmente a Miguel A. Fernández, Yvon Maday, Sidi M. Kaber, Frédéric Legoll y Martin Gander. Otra mención a los colegas con los cuales trabajé en temas cercanos, particularmente (en orden más o menos cronológico) a Ulrich Razafison, Matteo Astorino, Alfio Quarteroni, Alexei Lozinski, Pauline Klein, Yves Renard, Véronique Martin, Hao Huang, Nicolas Pignet y Christian Klein. Agradezco muchísimo a Sofía Llavayol y a Alejandro Passeggi por sus trabajos cuidadosos de revisión y sugerencias, a Gabriel Santoro por su trabajo de revisión y de edición y a Ana Trevellini por su trabajo de gestión. Agradezco, finalmente, al sello editorial DIRAC de la Facultad de Ciencias por permitir la edición y publicación de este libro.

Prólogo

Este libro tiene su origen en un curso que dicté para las licenciaturas de Matemática y de Física de la Facultad de Ciencias de la Universidad de la República. El mismo brinda las nociones básicas que permiten aproximar una ecuación diferencial ordinaria. Esto significa, básicamente, describir un algoritmo que permite obtener una secuencia de números que se aproximan al valor de la solución de la ecuación para varios tiempos definidos. Para saber si el algoritmo funciona bien, es decir, si proporciona efectivamente números cercanos a los valores de la solución exacta y si se comporta de manera razonable cuando modificamos los parámetros, las matemáticas son imprescindibles. Es, particularmente, el rol del análisis numérico asegurar que el método de aproximación funcione bien o señalar sus fallas.

Quise hacer algo muy introductorio que reflejara parte de mi propia experiencia en investigación y que requiriera solo un mínimo de conocimientos previos, en la línea de los libros de M. Crouzeix y A. L. Mignot [16], y de J. P. Demailly [17]. También busqué hacer algo que no solo se dedicara a ecuaciones diferenciales, sino que también pudiera servir como una introducción al análisis numérico en general, y que pudiera extenderse después a las ecuaciones en derivadas parciales. De hecho, para ecuaciones en derivadas parciales que dependen del tiempo, como la ecuación del calor o la ecuación de ondas, se obtiene un sistema de ecuaciones diferenciales ordinarias una vez realizada la discretización espacial con algún método como diferencias finitas, volúmenes finitos o elementos finitos.

Existen varios libros dedicados al tema, por ejemplo, los clásicos de A. Iserles [46], J. D. Lambert [49], y los de E. Hairer y G. Wanner y coautores [36], [37], [38], [39]. Además, varios libros clásicos generales de análisis numérico tratan de aproximación de ecuaciones diferenciales ordinarias, por ejemplo, los de W. Gander, M. J. Gander y F. Kwok [33], W. Gautschi [34], J. H. Hubbard y F. Hubert [44], J. Rappaz y M. Picasso [53], M. Schatzman [56] y el de E. Süli y D. F. Mayers [60].

Sin duda, recomiendo consultar las referencias mencionadas anteriormente y varias otras para profundizar en el tema, además de descubrir otros aspectos. Por otra parte, recomiendo probar los códigos Python asociados a los diversos métodos y ejemplos (la mayor parte estando disponibles también en la web [12]).

Franz Chouly

Capítulo 1
Introducción

El objetivo de este libro es ofrecer una primera aproximación a los métodos numéricos para resolver ecuaciones diferenciales ordinarias inspiradas en la física, la geometría, la ingeniería, entre otras disciplinas. Se centra tanto en los fundamentos y justificaciones matemáticas de los métodos numéricos como en los aspectos prácticos relacionados con su implementación.

Se asume que el lector cuenta con conocimientos básicos de análisis y algunas nociones sobre ecuaciones diferenciales ordinarias.

1.1. Un ejemplo para comenzar

Para motivar todo lo que sigue, comenzamos con un ejemplo que puede encontrarse en un libro como el de W. Gander, M. J. Gander y F. Kwok [33]: el problema del perro y el corredor. Veamos primero cómo se puede obtener la ecuación diferencial.

1.1.1. Formulación

Un corredor (el objetivo) tiene su posición en el plano dada por las coordenadas $(\eta(t), \xi(t))$ con t el tiempo. Es seguido por un perro, cuya posición es $(x(t), y(t))$, ver figura 1. Hay dos hipótesis que permiten determinar la trayectoria del perro:

- *Hipótesis 1.* La magnitud de la velocidad w del perro es constante en el tiempo. Esto se traduce en la ecuación:

$$\dot{x}(t)^2 + \dot{y}(t)^2 = w^2 (> 0), \tag{1.1}$$

  para todo t real positivo, y donde $\dot{x}(t)$, respectivamente $\dot{y}(t)$, representa la derivada en tiempo de $x(t)$, respectivamente de $y(t)$.

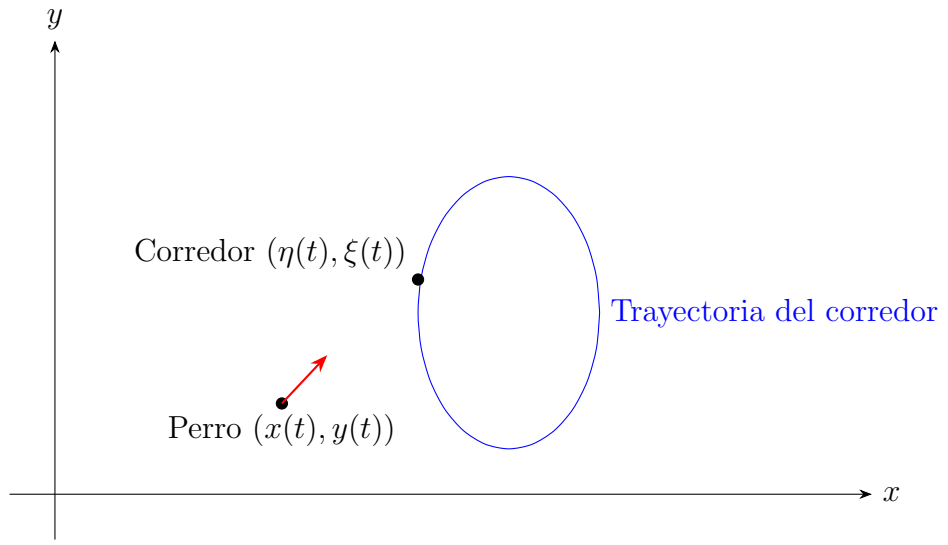


Figura 1: Perro y corredor.

- *Hipótesis 2.* El perro siempre apunta hacia el corredor (ver figura 1). Es decir, existe una función $\lambda(t) > 0$ tal que:

$$\dot{x}(t) = \lambda(t)(\eta(t) - x(t)), \tag{1.2}$$
$$\dot{y}(t) = \lambda(t)(\xi(t) - y(t)). \tag{1.3}$$

Ahora utilizamos (1.2) y (1.3) en la primera ecuación (1.1) para determinar λ:

$$(\lambda(t))^2\big((\eta(t) - x(t))^2 + (\xi(t) - y(t))^2\big) = w^2.$$

De donde se obtiene:

$$\lambda(t) = \frac{w}{\sqrt{(\eta(t) - x(t))^2 + (\xi(t) - y(t))^2}}.$$

Finalmente, el sistema de ecuaciones (1.2)-(1.3) queda expresado como:

$$\dot{x}(t) = w\frac{\eta(t) - x(t)}{\sqrt{(\eta(t) - x(t))^2 + (\xi(t) - y(t))^2}}, \tag{1.4}$$
$$\dot{y}(t) = w\frac{\xi(t) - y(t)}{\sqrt{(\eta(t) - x(t))^2 + (\xi(t) - y(t))^2}}. \tag{1.5}$$

Este sistema debe complementarse con dos ecuaciones que indiquen la posición inicial (x_0, y_0) del perro (condiciones iniciales en $t = 0$):

$$x(0) = x_0, \quad y(0) = y_0. \tag{1.6}$$

El sistema de ecuaciones (1.4)-(1.5)-(1.6) no tiene una solución analítica evidente. Sin embargo, puede ser aproximado y resuelto numéricamente mediante el esquema de Euler explícito[1].

1.1.2. Aproximación de Euler

Se toma un paso de tiempo pequeño $\tau > 0$ y dividimos el intervalo de tiempo $[0, T]$ así:

$$0 = t_0 < t_1 < \ldots t_N = T,$$

con $t_n = n\tau$, $n = 0, \ldots, N$. Para cada valor de n, buscamos aproximaciones x_n y y_n de x y y en el tiempo t_n:

$$x_n \simeq x(t_n), \quad y_n \simeq y(t_n).$$

En el sistema de ecuaciones (1.4)-(1.5) utilizamos una aproximación de diferencias finitas para las derivadas:

$$\begin{aligned} \dot{x}(t_n) &\simeq \frac{x(t_n + \tau) - x(t_n)}{\tau}, \\ \dot{y}(t_n) &\simeq \frac{y(t_n + \tau) - y(t_n)}{\tau}, \end{aligned}$$

donde asumimos que τ es suficientemente pequeño para estar cerca del límite. Se obtiene

$$\frac{x_{n+1} - x_n}{\tau} = w \frac{\eta(t_n) - x_n}{\sqrt{(\eta(t_n) - x_n)^2 + (\xi(t_n) - y_n)^2}}, \tag{1.7}$$

$$\frac{y_{n+1} - y_n}{\tau} = w \frac{\xi(t_n) - y_n}{\sqrt{(\eta(t_n) - x_n)^2 + (\xi(t_n) - y_n)^2}}. \tag{1.8}$$

[1]Sobre la historia de L. Euler y de este método ver, por ejemplo, [32], [33], [37], [38], [60].

Lo que se escribe, también:

$$x_{n+1} = x_n + \tau w \frac{\eta(t_n) - x_n}{\sqrt{(\eta(t_n) - x_n)^2 + (\xi(t_n) - y_n)^2}}, \tag{1.9}$$

$$y_{n+1} = y_n + \tau w \frac{\xi(t_n) - y_n}{\sqrt{(\eta(t_n) - x_n)^2 + (\xi(t_n) - y_n)^2}}. \tag{1.10}$$

Y tenemos fórmulas recursivas para calcular las coordenadas al paso $n+1$ con las del paso n. Estas se complementan con las condiciones iniciales x_0 y y_0 para $n = 0$. Esto se implementa sin ninguna dificultad.

1.1.3. Algo de Python

El código siguiente se encarga de calcular una trayectoria aproximada del perro y del corredor utilizando las fórmulas de Euler (1.9)-(1.10). Necesita solamente las librerías NumPy[2] y Matplotlib[3].

```
import numpy as np
import matplotlib.pyplot as plt

# Parámetros
w = 1.0  # Velocidad del perro
dt = 0.01  # Tamaño del paso de tiempo
t_end = 10.0  # Tiempo final

# Definir eta(t) y xi(t) como funciones del tiempo
def eta(t):
    return np.sin(t)
def xi(t):
    return np.cos(t)

# Inicializar arreglos
t = np.arange(0, t_end, dt)  # Arreglo de tiempo
x = np.zeros_like(t)  # x(t)
y = np.zeros_like(t)  # y(t)

# Condiciones iniciales
x[0] = 0.0  # Modificar si es necesario
```

[2]NumPy: The fundamental package for scientific computing with Python (https://numpy.org/).

[3]Matplotlib: Visualization with Python (https://matplotlib.org/).

```
y[0] = 0.0  # Modificar si es necesario

# Método de Euler (hacia adelante)
for i in range(1, len(t)):
    r = np.sqrt((eta(t[i-1]) - x[i-1])**2 + (xi(t[i-1]) -
        y[i-1])**2)
    x[i] = x[i-1] + dt * (w / r) * (eta(t[i-1]) - x[i-1])
    y[i] = y[i-1] + dt * (w / r) * (xi(t[i-1]) - y[i-1])

# Graficar la trayectoria en el espacio de fase
plt.figure(figsize=(8, 6))
plt.plot(x, y, label='Trayectoria (x, y)')
plt.xlabel('x')
plt.ylabel('y')
plt.title('Trayectoria del perro')
plt.legend()
plt.grid()
plt.show()
```

Listado 1: Implementación en Python de Euler explícito para el problema del perro y corredor.

El resultado obtenido se puede ver en la figura 2. El perro parte del origen $(0, 0)$ y se acerca a la trayectoria circular del corredor.

1.1.4. Epílogo

Acá vemos que un método como el de Euler permite obtener fácilmente una solución muy precisa de una ecuación diferencial, incluso cuando es muy compleja y no tiene una solución analítica evidente. Por supuesto, el problema que resolvemos en Python ya no es exactamente el mismo que el problema inicial, porque:

1. aproximamos una función continua por una sucesión finita de números reales;

2. aproximamos una derivada exacta por un cociente entre dos valores de funciones;

3. la implementación en la computadora introduce otro sesgo, ya que los números reales se aproximan mediante números de punto flotante (con una precisión fija en la cantidad de decimales).

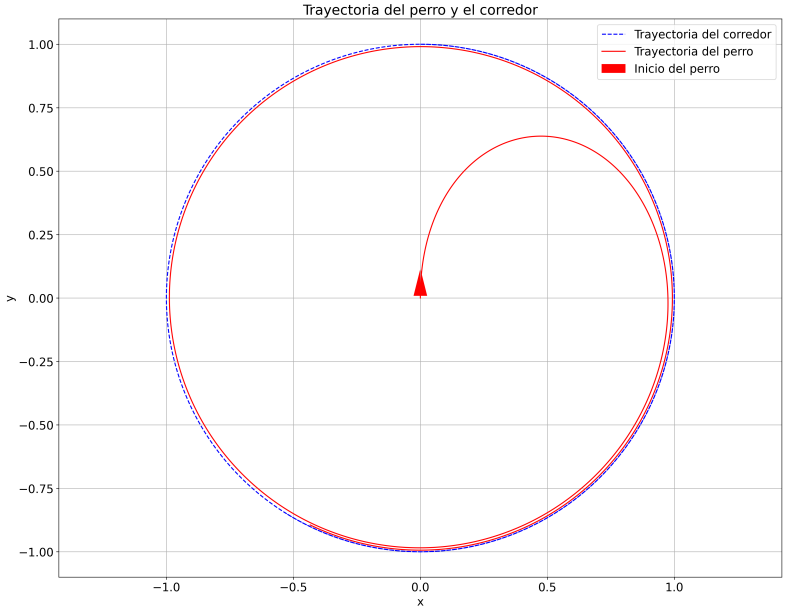


Figura 2: Trayectoria $(x(t), y(t))$ del perro a partir de su posición inicial $(0, 0)$ en el plano de fase, calculada en Python con el método de Euler explícito.

Entonces, se introducen errores. Si queremos ser rigurosos, tenemos que justificar matemáticamente que las aproximaciones realizadas van a tener un impacto mínimo y controlado en los errores, particularmente cuando elegimos el paso de tiempo τ pequeño. Además, sería interesante saber cómo se mejora la aproximación de la solución cuando τ disminuye.

Mencionamos también que el esquema de Euler puede dar resultados desastrosos en ciertas situaciones, por lo que es clave entender por qué ocurre esto y contar con alternativas para la aproximación. Para terminar, mencionamos también que muchas librerías de cálculo, como SciPy o Julia, tienen instrucciones predefinidas para solucionar de forma aproximada EDO y no es siempre preciso implementar un algoritmo completo. Estas instrucciones utilizan automáticamente un método predefinido (veremos después cuáles) y también se puede especificar opcionalmente un método a elección.

1.2. En pocas palabras

Ahora volvemos a un *problema de valor inicial* simple, pero más general que en la sección anterior 1.1. Sea $T > 0$ un real.

$$\begin{cases} y'(t) &= f(t, y(t)), \quad 0 \le t \le T, \\ y(0) &= y^0. \end{cases} \tag{1.11}$$

Aquí $y : [0, T] \to \mathbb{R}$ es la solución (desconocida), una función que depende de la variable (tiempo) t, y' es la derivada de primer orden de y, y $f : \mathbb{R}^+ \times \mathbb{R} \to \mathbb{R}$ es una función dada (conocida) de dos variables. El escalar (real) y^0 es conocido. La primera ecuación de (1.11) es una ecuación diferencial ordinaria (EDO) de primer orden y la segunda es la condición inicial en el tiempo $t = 0$. Suponemos que f es lo suficientemente amable, por ejemplo, globalmente Lipschitz, para que (1.11) tenga una única solución y, bien definida y diferenciable de forma continua en todo el intervalo $[0, T]$.

1.2.1. Diferencias finitas

Supongamos que f, aunque lo suficientemente amable como para que (1.11) esté bien planteado, no es tan amable, sin embargo, como para que sea fácil encontrar una solución explícita en forma cerrada. Entonces, si realmente estamos interesados en resolver esto, una opción realista es resolverlo aproximadamente usando una computadora. El principal problema, en este punto, es que una computadora es muy eficiente para realizar rápidamente operaciones aritméticas básicas, pero no lo suficientemente inteligente como para entender los conceptos de función y derivada. Así que no podemos darle directamente (1.11) y decirle: "¡Por favor, resuelve esto!", necesitamos preprocesarlo para que solo tenga que tratar con secuencias de números. La forma más sencilla de hacer esto es dividir la línea medio-real en pequeños intervalos de tamaño τ, donde $\tau > 0$ es un parámetro elegido llamado el *paso de tiempo*. Como resultado, definimos la secuencia de valores reales $t_n := n\tau$, para cualquier $0 \le n \le N$, $N = T/\tau$:

$$0 = t_0 < t_1 (= \tau) < \ldots < t_n (= n\tau) < \ldots < t_N = T.$$

A cada valor de n también le asociamos una incógnita real y_n, y trabajaremos lo suficiente para que, al final, la computadora proporcione un valor explícito de y_n tal que

$$y_n \simeq y(t_n),$$

y más adelante precisaremos el significado de $\simeq$. La idea es que, si τ es lo suficientemente pequeño, la secuencia (y_n) sea lo suficientemente precisa como para representar aproximadamente la función y. Por supuesto, faltan muchos valores de y, pero podrían recuperarse, si es necesario, utilizando, por ejemplo, interpolación lineal entre dos valores sucesivos y_n y y_{n+1}.

Ahora necesitamos aproximar el valor de la derivada y' usando la secuencia (y_n). La idea básica es hacer uso de la *diferencia finita*:

$$y'(t_n) \simeq \frac{y_{n+1} - y_n}{\tau}.$$

Solo hagamos una pequeña observación por el momento: si (y_n) representa con precisión la función y, y si τ es muy pequeño, deberíamos aproximar correctamente la derivada, dado que

$$y'(t_n) = \lim_{\tau \to 0} \frac{y(t_{n+1}) - y(t_n)}{\tau}.$$

La condición inicial se transcribe simplemente como

$$y_0 = y^0.$$

A partir de aquí surgen tres opciones naturales para la discretización de (1.11), que dependen de cómo tratemos el lado derecho de la primera ecuación $(f(t, y(t)))$:

1. La primera opción: tomamos $f(t, y(t))$ en el paso de tiempo n, lo que lleva a la ecuación, para todo $0 \leq n < N$:

$$\frac{y_{n+1} - y_n}{\tau} = f(t_n, y_n). \tag{1.12}$$

   A esto se le llama el *esquema de Euler hacia adelante* o *explícito*. Es el mismo de la sección anterior 1.1.

2. La segunda opción: tomamos $f(t, y(t))$ en el paso de tiempo $n+1$, lo que lleva a la ecuación, para todo $0 \leq n < N$:

$$\frac{y_{n+1} - y_n}{\tau} = f(t_{n+1}, y_{n+1}). \tag{1.13}$$

   A esto se le llama el *esquema de Euler hacia atrás* o *implícito*.

3. La tercera opción: tomamos una combinación convexa de los pasos de tiempo n y $n+1$. Esto lleva, por ejemplo, a la siguiente ecuación, para todo $0 \leq n < N$:

$$\frac{y_{n+1} - y_n}{\tau} = \frac{1}{2}(f(t_n, y_n) + f(t_{n+1}, y_{n+1})). \tag{1.14}$$

   A esto se le llama el *esquema de Crank-Nicolson*. El valor de $1/2$ no es casual, y veremos también cómo se justifica.

1.2.2. Hacia la implementación

Empecemos con la implementación del primer esquema (1.12), el método de Euler explícito. Notemos que la relación (1.12) se puede reescribir como:

$$y_{n+1} = y_n + \tau f(t_n, y_n). \tag{1.15}$$

Esto nos da una fórmula de recurrencia sencilla para calcular la aproximación y_{n+1} en el tiempo t_{n+1}, a partir de la solución aproximada y_n en el tiempo t_n. No hay más nada que hacer. Como la solución aproximada en un instante de tiempo discreto se obtiene explícitamente a partir de la solución previa, este esquema se llama *explícito*.

Para el esquema de Euler implícito (1.13), también podemos reescribirlo de la siguiente forma:

$$y_{n+1} - \tau f(t_{n+1}, y_{n+1}) = y_n. \tag{1.16}$$

Esto también proporciona una fórmula de recurrencia, pero no es tan sencilla como la de Euler explícito. En efecto, la aproximación y_{n+1} no se obtiene directamente, sino que debe calcularse como la raíz de la función $I - \tau f(t_{n+1}, \cdot) - y_n$ (donde I denota la identidad). Este cálculo se puede

hacer de varias formas, por ejemplo, usando el método de Newton, pero requiere un esfuerzo adicional y un mayor costo computacional.

Por esta razón, este esquema se llama *implícito* (y se lo conoce como método de Euler implícito en el tiempo). A pesar de esta dificultad, este esquema sigue siendo interesante en muchas situaciones. Más adelante veremos exactamente por qué.

El esquema de Crank-Nicolson (1.14) se puede reescribir como:

$$y_{n+1} - \frac{1}{2}\tau f(t_{n+1}, y_{n+1}) = y_n + \frac{1}{2}\tau f(t_n, y_n). \tag{1.17}$$

Notemos que este esquema sigue siendo implícito. También es interesante notar que cualquier otra combinación convexa entre los tiempos n y $n+1$ podría tener sentido, pero el esquema de Crank-Nicolson corresponde a los valores $(\frac{1}{2}, \frac{1}{2})$ (si se usan otros valores, se habla de esquemas θ). A primera vista, este esquema parece no tener ventajas sobre el de Euler implícito, pero más adelante veremos que sí las tiene.

Una última observación sobre la implementación: aunque todos estos esquemas son muy fáciles de programar, su principal desventaja es que, para obtener la solución en un tiempo final dado, es necesario calcular iterativamente todas las soluciones intermedias, lo que puede ser costoso en ciertos problemas. De hecho, en el caso de las EDO, no hay una forma directa de resolverlas rápidamente usando arquitecturas paralelas y se requieren esquemas especiales si se quiere lograr esto [30], [50].

1.2.3. Estabilidad numérica

Ahora, para cerrar esta primera parte, introduzcamos algunas nociones básicas sobre la estabilidad numérica de los esquemas anteriores. Para esto, consideremos la siguiente variante de (1.11):

$$\begin{cases} y'(t) &= -\alpha\, y(t), \\ y(0) &= 1. \end{cases} \tag{1.18}$$

Aquí, $\alpha \in \mathbb{C}$ es un parámetro complejo, con $\Re(\alpha) \geq 0$. La ecuación anterior tiene la siguiente solución exacta:

$$y(t) = \exp(-\alpha t).$$

Esta solución es *acotada*, en el sentido de que su módulo se mantiene menor o igual a 1:

$$|y(t)| \leq 1, \qquad \forall t \in \mathbb{R}^+.$$

Aproximemos ahora esta solución utilizando el esquema de Euler explícito. Siguiendo la ecuación (1.15), obtenemos:

$$y_{n+1} = y_n - \alpha\tau y_n = (1 - \alpha\tau)\, y_n. \tag{1.19}$$

En este caso, no hace falta usar una computadora para obtener la solución discreta, ya que está dada por la siguiente fórmula para cualquier n:

$$y_n = (1 - \alpha\tau)^n.$$

Por lo tanto, el módulo de la solución discreta es:

$$|y_n| = |1 - \alpha\tau|^n.$$

Esperamos que, al menos, la solución discreta se mantenga acotada, pero acá observamos que esto no siempre sucede. Una condición necesaria y suficiente, en este caso, es que:

$$|1 - \alpha\tau| \leq 1$$

o, equivalentemente:

$$(1 - \Re(\alpha)\tau)^2 + \Im(\alpha)^2\tau^2 \leq 1.$$

Particularmente, para α real, tenemos:

$$-1 \leq 1 - \alpha\tau \leq 1,$$

lo que se puede reformular como:

$$\tau \leq \frac{2}{\alpha}$$

(donde usamos que α y τ son positivos). Esto significa que la solución discreta (y_n) se mantiene acotada solo si el paso de tiempo τ es lo suficientemente pequeño. En este caso, decimos que el esquema de Euler explícito es *condicionalmente estable*: es estable si τ es lo bastante chico, e inestable en caso contrario. Obviamente, queremos evitar la inestabilidad numérica, ya que en esa situación la solución discreta no se parece en nada a la solución exacta.

1.3. Descripción de cada capítulo

El contenido del libro se estructura de la siguiente manera:

- El capítulo 2 presenta nociones generales sobre las ecuaciones diferenciales ordinarias, en particular, resultados de existencia y unicidad.
- El capítulo 3 introduce los métodos explícitos de un paso para aproximar ecuaciones diferenciales, que son los más sencillos. En particular, revisamos el método de Euler explícito y presentamos los métodos de Runge-Kutta.
- El capítulo 4 se centra en los métodos multipaso explícitos, que permiten alcanzar una buena precisión a bajo costo. No obstante, son más delicados desde el punto de vista de la estabilidad numérica.
- El capítulo 5 introduce varios métodos implícitos, que requieren la resolución de ecuaciones algebraicas en cada tiempo discreto. Estos métodos son más adecuados para las ecuaciones diferenciales rígidas.
- El capítulo 6 es una introducción a métodos simplécticos para sistemas hamiltonianos, que permiten la conservación de ciertas cantidades de interés geométrico o físico.

Terminamos con una bibliografía, con la esperanza de que proporcione ideas para profundizar en el tema y descubrir otras nociones.

Capítulo 2
Ecuaciones diferenciales ordinarias

En este capítulo presentamos la teoría básica sobre ecuaciones diferenciales ordinarias. Comenzamos con algunas definiciones; luego enunciamos un primer resultado fundamental de existencia (Cauchy-Peano-Arzelà), y después un resultado de existencia y unicidad (Cauchy-Lipschitz o Picard-Lindelöf). Seguimos el libro de J. P. Demailly [17].

2.1. Algunas definiciones básicas

Consideramos ecuaciones diferenciales en $\mathbb{R}$ para simplificar. No obstante, lo que sigue se extiende sin mayor problema a sistemas de ecuaciones diferenciales en $\mathbb{R}^d$, $d \geq 1$. Sea U un conjunto abierto de $\mathbb{R} \times \mathbb{R}$ y

$$f : U \to \mathbb{R}$$

una función continua. Una ecuación diferencial ordinaria es de la forma

$$y' = f(\cdot, y). \tag{2.1}$$

Una solución de (2.1) en un intervalo I de $\mathbb{R}$ es una función derivable y tal que:

$$\begin{array}{ll} (t, y(t)) \in U, & \forall t \in I, \\ y'(t) = f(t, y(t)) & \forall t \in I. \end{array} \tag{2.2}$$

Especificamos, además, un punto $(t_0, y_0) \in U$, y un *problema de Cauchy* consiste en encontrar un intervalo I que contenga a t_0 y una función y que:

1. resuelva la ecuación diferencial (2.2), y
2. satisfaga la condición inicial $y(t_0) = y_0$.

Ejemplo 1. *Tomemos* $U = \mathbb{R}^2$, $\alpha \in \mathbb{R}$ *y* $f(t, y) = \alpha y$. *Sea* $I = \mathbb{R}$, *una solución de la ecuación diferencial*

$$y' = \alpha y$$

es de la forma $y(t) = Ce^{\alpha t}$, *donde* $C \in \mathbb{R}$ *es una constante arbitraria. Si fijamos* $y(0) = 1$, *obtenemos* $y(t) = e^{\alpha t}$, *que es la solución del problema de Cauchy:* $y' = \alpha y$, $y(0) = 1$.

Hemos visto otro ejemplo en el capítulo 1. Existen varios otros ejemplos clásicos que vienen de la física, la geometría, la biología, la química, etc. Nos referimos, particularmente, a las ecuaciones de la *tractrix*, del péndulo, de Lorenz, de Van der Pol, entre otros clásicos. Algunas de ellas las veremos en los siguientes capítulos. Más ejemplos se pueden encontrar en [1], [17], [33], [36], [37], [38], [39], entre otros.

2.2. Teorema de Cauchy-Peano-Arzelà

Ahora presentamos un primer resultado general sobre la existencia de soluciones locales para un problema de Cauchy. Solo se necesita una hipótesis de continuidad en f. Por otra parte, esta hipótesis sola no garantiza la unicidad: pueden coexistir varias soluciones de un mismo problema de Cauchy. Seguimos con la ecuación:

$$y' = f(t, y), \tag{2.3}$$

con la hipótesis de que $f : U \to \mathbb{R}$ es continua en U, donde U es un abierto. Vamos a demostrar, en varias etapas, el resultado siguiente.

Teorema 1 (Cauchy-Peano-Arzelà – primera versión)**.** *Sea* $t_0, y_0 \in \mathbb{R}$ *con* $(t_0, y_0) \in U$. *Supongamos que* $f : U \to \mathbb{R}$ *es continua en* U. *Entonces, existe* $T > 0$ *y (por lo menos) una solución de* (2.3) *en* $[t_0 - T, t_0 + T]$ *que satisface la condición inicial* $y(t_0) = y_0$.

Ahora vamos a demostrar este teorema, y al fin a dar una versión más precisa.

2.2.1. Otra formulación del problema de Cauchy

Va a ser conveniente formular el problema de Cauchy integrándolo. Es el propósito del siguiente lema:

Lema 1. *Sea $I \subset \mathbb{R}$ un intervalo. La función $y : I \to \mathbb{R}$ es solución del problema de Cauchy*

$$y' = f(t, y), \quad y(t_0) = y_0,$$

de datos iniciales (t_0, y_0) si y solo si:

1. *para todo $t \in I$, $(t, y(t)) \in U$,*
2. *para todo $t \in I$,*

$$y(t) = y_0 + \int_{t_0}^{t} f(s, y(s))\ \mathrm{d}s.$$

Demostración. Demostramos la equivalencia:

1. si y es solución del problema de Cauchy, satisface el punto 1 inmediatamente, y también el punto 2 si integramos la ecuación (2.3) entre t_0 y t y luego tomamos en cuenta la condición $y(t_0) = y_0$;
2. ahora, si y satisface 1 y 2, derivamos 2 y encontramos la ecuación (2.3). Con 2 en $t = t_0$ verificamos la condición inicial $y(t_0) = y_0$.

□

2.2.2. Cilindros de seguridad

Como U es abierto, existe un cilindro

$$C_0 = [t_0 - T_0, t_0 + T_0] \times [y_0 - r_0, y_0 + r_0]$$

contenido en U, con $T_0 > 0$ y $r_0 > 0$. Dado que C_0 es cerrado y acotado en $\mathbb{R}^2$, se sigue que es compacto. Como consecuencia, la función f, al ser continua, es acotada en C_0. Es decir, existe $M < +\infty$ tal que

$$M = \sup_{(t,y) \in C_0} |f(t, y)|.$$

Sea $T \leq T_0$ y definimos

$$C := [t_0 - T, t_0 + T] \times [y_0 - r_0, y_0 + r_0].$$

Definición 1. *Decimos que C es un* cilindro de seguridad *para* (2.3)*, si toda solución y del problema de Cauchy con condición inicial $y(t_0) = y_0$, definida en un intervalo $I \subset [t_0 - T, t_0 + T]$, permanece contenida en $[y_0 - r_0, y_0 + r_0]$.*

Demostramos ahora que existe siempre un tiempo T que permite tener un cilindro de seguridad. Es suficiente elegir T suficientemente pequeño.

Proposición 1. *Si $T > 0$, cumple con lo siguiente:*

$$T \leq \text{mín}\left(T_0, \frac{r_0}{M}\right), \tag{2.4}$$

entonces, C es un cilindro de seguridad.

Demostración. Supongamos que la solución y escapa de C en el intervalo $[t_0, t_0 + T]$, y sea τ el primer instante en el que esto ocurre:

$$\tau := \text{ínf}\{t \in [t_0, t_0 + T] \mid |y(t) - y_0| > r_0\}.$$

Por definición de τ, se cumple que $|y(t) - y_0| \leq r_0$ para todo $t < \tau$, y por continuidad de y, se tiene que $|y(\tau) - y_0| = r_0$. Como $(t, y(t)) \in C \subset C_0$ para $t_0 \leq t \leq \tau$, se deduce que

$$|y'(t)| = |f(t, y(t))| \leq M.$$

Integrando lo anterior entre t_0 y τ, se obtiene

$$r_0 = |y(\tau) - y_0| = \left| \int_{t_0}^{\tau} y'(s) \mathrm{d}s \right| \leq \int_{t_0}^{\tau} |y'(s)| \, \mathrm{d}s \leq M(\tau - t_0).$$

Por lo tanto,

$$\tau - t_0 \geq \frac{r_0}{M},$$

lo que no es posible si suponemos la relación (2.4). □

2.2.3. Aproximación de Euler

Volvemos ahora a encontrarnos con el método de Euler explícito del capítulo 1. Construimos en $[t_0; t_0 + T]$ una aproximación de la solución de (2.3). Es suficiente considerar $[t_0; t_0 + T]$ como la mitad del intervalo $[t_0 - T; t_0 + T]$, dado que el proceso se aplica de la misma forma en el otro subintervalo $[t_0 - T; t_0]$. Elegimos $N \geq 1$ y definimos el paso de tiempo $\tau = T/N$. Notamos $t_n = t_0 + n\tau$, $0 \leq n \leq N$. Definimos una función y_E de la forma siguiente:

$$\begin{aligned} y_E(0) &= y_0, && (2.5) \\ y_E(t) &= y_n + (t - t_n) f(t_n, y_n),\ t \in [t_n; t_{n+1}],\ \forall\ 0 \leq n < N, && (2.6) \end{aligned}$$

con la convención $y_n = y_E(t_n)$. Es otra forma de definir el método de Euler explícito. De hecho, volvemos a encontrar la fórmula

$$y_{n+1} = y_n + \tau f(t_n, y_n)$$

si evaluamos (2.6) en t_{n+1}, $0 \leq n < N$. Simplemente hacemos una interpolación lineal (con polinomios de grado uno) entre los pasos. También se puede hablar de método poligonal. Esto permite reconstruir, a partir de la secuencia $(y_n)_{0 \leq n < N}$, una función y_E en el sentido usual. Se verifica directamente que y_E es globalmente continua en $[t_0; t_0 + T]$ y, además, es $\mathcal{C}^1$ por trozos $(t_n; t_{n+1})$ en $[t_0; t_0 + T]$. Recordamos que la notación $\mathcal{C}^1$ se refiere a funciones derivables de derivada continua. Probamos ahora que estas soluciones aproximadas se quedan también adentro de los cilindros de seguridad.

Proposición 2. *Sea C un cilindro de seguridad de tamaño T, con*

$$T \leq \text{mín}\left(T_0, \frac{r_0}{M}\right).$$

Entonces, toda solución aproximada de Euler y_E permanece dentro de $[y_0 - r_0; y_0 + r_0]$.

Demostración. Procedemos por recurrencia en n. Verificamos que para todo n:

$$y_E([t_0, t_n]) \subset [y_0 - r_0; y_0 + r_0] \,\text{y}\, |y_E(t) - y_0| \leq M(t - t_0),\ \forall\ t_0 \leq t \leq t_n.$$

Para $n = 0$, esto es inmediato. Supongamos que la propiedad se cumple hasta el paso n. Como $(t_n, y_n) \in C$, se tiene que $|f(t_n, y_n)| \leq M$. Entonces, para $t_n \leq t \leq t_{n+1}$, partimos de (2.6) y acotamos:

$$|y_E(t) - y_n| = (t - t_n)|f(t_n, y_n)| \leq M(t - t_n).$$

Usando la hipótesis de recurrencia, sabemos que $|y_n - y_0| \leq M(t_n - t_0)$. Por la desigualdad triangular, para $t_n \leq t \leq t_{n+1}$,

$$|y_E(t) - y_0| \leq |y_E(t) - y_n| + |y_n - y_0| \leq M(t - t_n) + M(t_n - t_0) = M(t - t_0).$$

Dado que $MT \leq r_0$, se concluye por otro lado que $y_E(t)$ partenece a $[y_0 - r_0; y_0 + r_0]$. □

2.2.4. Error del método de Euler

Sea, ahora, ω_f el módulo de continuidad de f en C, definido como:

$$\omega_f(u) := \sup\{|f(t_1, y_1) - f(t_2, y_2)| \mid (t_1, y_1), (t_2, y_2) \in C(u)\},$$

con

$$C(u) := \{(t_1, y_1), (t_2, y_2) \in C,\ |t_2 - t_1| + |y_2 - y_1| \leq u\}.$$

Dado que el cilindro de seguridad C es compacto, la función f es uniformemente continua en C, por lo que

$$\lim_{u \to 0} \omega_f(u) = 0. \tag{2.7}$$

Supongamos que C es un cilindro de seguridad con $T \leq \min(T_0, r_0/M)$. Definimos el residuo asociado a la ecuación diferencial (2.3) aproximada por Euler como:

$$\epsilon := \sup_{t_0 \leq t \leq t_0 + T_0} |y'_E(t) - f(t, y_E(t))|.$$

En otros términos, mesuramos hasta qué punto la solución aproximada y_E satisface la ecuación diferencial. Demostramos primero lo siguiente.

Proposición 3. *Sea y_E una solución aproximada de Euler definida por (2.5)–(2.6) y asociada a un paso de tiempo $\tau > 0$. Entonces, el residuo ϵ cumple la cota:*

$$\epsilon \leq \omega_f((M + 1)\tau),$$

donde ω_f es el módulo de continuidad de f y $M = \sup_{(t,y) \in C_0} |f(t, y)|$.

Demostración. Sea $0 \leq n < N$ y $t \in [t_n; t_{n+1}]$. Dado que $y'_E(t) = f(t_n, y_n)$, integrando obtenemos:

$$|y_E(t) - y_n| = (t - t_n)|f(t_n, y_n)| \leq M\tau.$$

Por definición del módulo de continuidad,

$$|f(t_n, y_n) - f(t, y(t))| \leq \omega_f(\tau + M\tau) = \omega_f((M+1)\tau).$$

Dado que $y'_E(t) = f(t_n, y_n)$, tenemos ahora

$$|y'_E(t) - f(t, y(t))| \leq \omega_f((M+1)\tau),$$

lo que termina la demostración. □

Luego, la siguiente proposición va a tener un rol importante.

Proposición 4. *Sea $(y_{E,k})_{k\geq 0}$ una sucesión de soluciones de Euler aproximadas definidas por (2.5)–(2.6), con residuos asociados $(\epsilon_k)_{k\geq 0}$ que cumplen*

$$\lim_{k\to+\infty} \epsilon_k = 0. \tag{2.8}$$

Si $(y_{E,k})_{k\geq 0}$ converge uniformemente a una función $y^\star$ en $[t_0; t_0 + T]$, entonces, $y^\star$ es una solución de la ecuación diferencial (2.3).

Demostración. Estimamos primero

$$\begin{aligned}
& \left| y_{E,k}(t) - y_{E,k}(t_0) - \int_{t_0}^{t} f(s, y_{E,k}(s))\mathrm{d}s \right| \\
= & \left| \int_{t_0}^{t} (y'_{E,k}(s) - f(s, y_{E,k}(s)))\mathrm{d}s \right| \\
\leq & \int_{t_0}^{t} \left| (y'_{E,k}(s) - f(s, y_{E,k}(s))) \right| \mathrm{d}s \\
\leq & \ \epsilon_k \int_{t_0}^{t} \mathrm{d}s
\end{aligned}$$

dada la definición del residuo. Entonces,

$$\left| y_{E,k}(t) - y_{E,k}(t_0) - \int_{t_0}^{t} f(s, y_{E,k}(s))\mathrm{d}s \right| \leq (t - t_0)\epsilon_k. \tag{2.9}$$

Sea $y^\star = \lim_{k\to+\infty} y_{E,k}$. Como la convergencia es uniforme, $y^\star$ también es continua, y podemos definir $\delta_k = \max_{t_0\le t\le t_0+T} |y^\star(t) - y_{E,k}(t)|$. Por definición del módulo de continuidad de f, se tiene que:

$$|f(s, y_{E,k}(s)) - f(s, y^\star(s))| \le \omega_f(\delta_k),$$

para $t_0 \le s \le t_0 + T$. Si integramos

$$\left| \int_{t_0}^{t} f(s, y_{E,k}(s))\mathrm{d}s - \int_{t_0}^{t} f(s, y^\star(s))\mathrm{d}s \right| \le (t - t_0)\omega_f(\delta_k), \tag{2.10}$$

la convergencia uniforme implica $\delta_k \to 0$ cuando $k \to +\infty$. Se sigue que $\omega_f(\delta_k) \to 0$ cuando $k \to +\infty$. Ahora pasamos al límite en k en (2.9), utilizamos (2.10) y obtenemos:

$$\left| y^\star(t) - y_0^\star - \int_{t_0}^{t} f(s, y^\star(s))\mathrm{d}s \right| = 0,$$

lo que implica la formulación integral de (2.3). Aplicando el lema 1, concluimos que $y^\star$ es una solución de (2.3). □

Observación 1. *En lo anterior, no se supone nada sobre la secuencia $(y_{E,k})_{k\ge0}$, sino que los residuos correspondientes tienden hacia cero. Obviamente, tal como hemos definido las aproximaciones de Euler, esto quiere decir que hacemos variar los pasos de tiempo correspondientes $(\tau)_{k\ge0}$ o, de forma equivalente, $(N_k)_{k\ge0}$.*

2.2.5. Teorema de existencia

Antes de mostrar el teorema de existencia necesitamos un resultado de análisis funcional, que es una variante simplificada del teorema de Ascoli-Arzelà. No lo vamos a demostrar, por lo que recomendamos ver [17] para la demostración detallada.

Teorema 2 (Ascoli-Arzelà)**.** *Sean $[a_1 : b_1]$ y $[a2 : b2]$ dos intervalos cerrados de $\mathbb{R}$ ($a_i, b_i \in \mathbb{R}, a_i < b_i$ para $i = 1, 2$). Sea $l \ge 0$ una sucesión $(\varphi_n)_{n\ge0}$ de funciones l-Lipschitz de $[a_1 : b_1]$ en $[a2 : b2]$:*

$$|\varphi_n(x) - \varphi_n(y)| \le l|x - y|, \quad \forall x, y \in [a_1 : b_1], \quad \forall n \in \mathbb{N}.$$

Entonces, se puede extraer de $(\varphi_n)_{n\ge0}$ una subsucesión convergente cuya límite φ sigue siendo una función l-Lipschitz.

Para un enunciado más general, puede consultarse el sitio PlanetMath[4] o también a los libros [14], [52], [61], donde el enunciado viene con una demostración completa.

Ahora podemos enunciar y mostrar:

Teorema 3 (teorema de existencia de Cauchy-Peano-Arzelà)**.** *Sea U un abierto de $\mathbb{R}^2$ y $f : U \to \mathbb{R}$ una funcion continua. Sea $(t_0, y_0) \in U$ y $C \subset U$ un cilindro de seguridad asociado. Entonces, existe una solución de $y' = f(t, y)$ en C que satisface $y(t_0) = y_0$.*

Demostración. Sea $C = [t_0 - T : t_0 + T] \times [y_0 - r_0; y_0 + r_0]$ un cilindro de seguridad. Sea $(y_{E,k})_{k\geq 0}$ la sucesión de soluciones aproximadas con el método de Euler (2.5)–(2.6) y con paso $\tau_k = T/k$. Con la fórmula (2.6), deducimos directamente que cada función $y_{E,k} : [t_0; t_0 + T] \to [y_0 - r_0; y_0 + r_0]$ es Lipschitz con constante M. Aplicamos el teorema 2 de Ascoli-Arzelà: se puede extraer una subsecuencia de $(y_{E,k})_{k\geq 0}$ que converge uniformemente a un límite y. Para simplificar, seguimos anotando $(y_{E,k})_{k\geq 0}$ esta misma subsecuencia.

Para cada $y_{E,k}$, el residuo ϵ_k verifica (proposición 3):

$$\epsilon_k \leq \omega_f \left(\frac{(M+1)T}{k} \right),$$

con ω_f el módulo de continuidad de f y M la constante que acota f. Sabemos, con (2.7), que ω_f tiende a 0 cuando $k \to +\infty$ y, entonces, verificamos (2.8). Finalmente, cumplimos todos los requisitos de la proposición 4: y es una solución exacta de $y' = f(t, y)$ en C. La condición inicial $y(t_0) = y_0$ está satisfecha, también, dada la convergencia uniforme de $(y_{E,k})_{k\geq 0}$. □

2.2.6. Ejemplo con soluciones múltiples

Suponiendo únicamente la continuidad de f, la solución y del teorema 3 no es siempre única. En caso de soluciones múltiples, hay una subsucesión de aproximaciones de Euler que converge hacia una de las

[4]PlanetMath (https://planetmath.org/ascoliarzelatheorem).

soluciones. Damos un ejemplo de problema de Cauchy con soluciones múltiples:

$$y'(t) = 3|y(t)|^{2/3}, \quad y(0) = 0$$

tiene como soluciones $y(t) = 0$ y $y(t) = t^3$ para todo $t \in \mathbb{R}$. Se pueden encontrar otros ejemplos en [60].

2.3. Teorema de Cauchy-Lipschitz

Presentamos aquí una forma muy general de volver a encontrar unicidad de soluciones. Será enunciado en un teorema muy famoso, el de Cauchy-Lipschitz o Picard-Lindelhöf.

Supongamos ahora que f, además de ser continua en U, es localmente Lipschitz en y: para todo punto $(t_0, y_0) \in U$, existe un cilindro $C_0 = [t_0 - T_0; t_0 + T_0] \times [y_0 - r_0; y_0 + r_0] \subset U$ y una constante $K = K(t_0, y_0) \geq 0$ tales que, para todo (t, y_1) y (t, y_2) en C_0, se cumple

$$|f(t, y_1) - f(t, y_2)| \leq K|y_1 - y_2|. \tag{2.11}$$

Una posibilidad para mostrar existencia y unicidad de soluciones bajo la hipótesis (2.11) se basa en el lema de Gronwall, que vamos a enunciar y demostrar.

2.3.1. Lema de Gronwall

Sean $y_{E,1}$ y $y_{E,2}$ dos soluciones aproximadas obtenidas por el método de Euler (2.5)–(2.6), con residuos asociados ϵ_1, ϵ_2, respectivamente. Sea de nuevo

$$M = \sup_{(t,y) \in C_0} |f(t, y)|.$$

Se deduce de (2.6) que $y_{E,1}$ y $y_{E,2}$ tienen sus derivadas acotadas por M. Entonces, $y_{E,1}$ y $y_{E,2}$ permanecen dentro del cilindro C siempre que

$$T \leq \text{mín}\left(T_0, \frac{r_0}{M}\right).$$

El lema de Gronwall nos da una estimación sobre la diferencia entre $y_{E,1}$ y $y_{E,2}$.

Lema 2 (Gronwall)**.** *Supongamos ahora que f es continua en U y verifica* (2.11)*. Sean $y_{E,1}$ y $y_{E,2}$ dos soluciones aproximadas obtenidas por el método de Euler, con residuos ϵ_1, ϵ_2. Entonces, para todo $t \in [t_0-T, t_0+T]$ se cumple*

$$|y_{E,1}(t) - y_{E,2}(t)| \leq \frac{\epsilon_1 + \epsilon_2}{K} \left(\exp(K|t - t_0|) - 1\right),$$

donde $K \geq 0$ es la constante de Lipschitz que aparece en (2.11)*.*

Demostración. Tomamos $t_0 = 0$, para simplificar, y sea $t \in (0, T)$. Primero tenemos

$$\begin{array}{rl} & |y'_{E,1}(t) - y'_{E,2}(t)| \\ \leq & |y'_{E,1}(t) - f(t, y_{E,1}(t))| + |y'_{E,2}(t) - f(t, y_{E,2}(t))| \\ & +|f(t, y_{E,1}(t)) - f(t, y_{E,2}(t))|. \end{array}$$

Como $y_{E,1}$ y $y_{E,2}$ son aproximaciones de Euler de residuos ϵ_1 y ϵ_2, obtenemos

$$\begin{array}{rcl} |y'_{E,1}(t) - y'_{E,2}(t)| & \leq & |f(t, y_{E,1}(t)) - f(t, y_{E,2}(t))| + \epsilon_1 + \epsilon_2 \\ & \leq & K|y_{E,1}(t) - y_{E,2}(t)| + \epsilon_1 + \epsilon_2, \end{array}$$

lo último por la hipótesis de que f es K-Lipschitz en y. Además, como las soluciones satisfacen la misma condición inicial, se cumple

$$y_{E,1}(t) - y_{E,2}(t) = \int_0^t (y'_{E,1}(s) - y'_{E,2}(s))\mathrm{d}s.$$

Pongamos $\epsilon := \epsilon_1 + \epsilon_2$. De lo anterior deducimos que

$$|y_{E,1}(t) - y_{E,2}(t)| \leq K \int_0^t |y_{E,1}(s) - y_{E,2}(s)|\mathrm{d}s + \epsilon t. \tag{2.12}$$

Definimos

$$v(t) = \int_0^t |y_{E,1}(s) - y_{E,2}(s)|\mathrm{d}s$$

y (2.12) se reformula

$$v'(t) \leq Kv(t) + \epsilon t.$$

Restando $Kv(t)$ y multiplicando por e^{-Kt} obtenemos

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(v(t)\right)e^{-Kt} - Kv(t)e^{-Kt} \leq \epsilon t e^{-Kt}$$

y, entonces,

$$\frac{\mathrm{d}}{\mathrm{d}t}\left(v(t)e^{-Kt}\right) \leq \epsilon t e^{-Kt}.$$

Integramos y utilizamos $v(0) = 0$ para obtener

$$v(t)e^{-Kt} \leq \int_0^t \epsilon u e^{-Ku}\mathrm{d}u.$$

Ahora integramos por partes

$$v(t)e^{-Kt} \leq \epsilon\left(-\frac{t}{K}e^{-Kt} - \frac{1}{K^2}(e^{-Kt} - 1)\right).$$

Reorganizamos:

$$v(t)e^{-Kt} \leq \frac{\epsilon}{K^2}\left(1 - (1 + Kt)e^{-Kt}\right)$$

y, por lo tanto,

$$v(t) \leq \frac{\epsilon}{K^2}\left(e^{Kt} - (1 + Kt)\right).$$

Ahora volvemos a la ecuación (2.12) que se escribe también:

$$|y_{E,1}(t) - y_{E,2}(t)| \leq Kv(t) + \epsilon t \leq \frac{\epsilon}{K}\left(e^{Kt} - (1 + Kt)\right) + \epsilon t.$$

Simplificamos:

$$|y_{E,1}(t) - y_{E,2}(t)| \leq \frac{\epsilon}{K}\left(e^{Kt} - 1\right).$$

Obtenemos la cota deseada, lo que concluye la prueba. □

2.3.2. Teorema de Cauchy-Lipschitz

Ahora enunciamos lo siguiente.

Teorema 4 (Cauchy-Lipschitz)**.** *Sea U un abierto de $\mathbb{R}^2$ y $f : U \to \mathbb{R}$ una función continua y localmente Lipschitz en la segunda variable (ver (2.11)). Sea $(t_0, y_0) \in U$ y $C = [t_0 - T; t_0 + T] \times [y_0 - r_0; y_0 + r_0]$ un cilindro de seguridad centrado en (t_0, y_0), con $T > 0$ y $r_0 > 0$. Entonces, existe una única solución y del problema de Cauchy $y' = f(t, y)$ con $y(t_0) = y_0$. Además, toda secuencia $(y_{E,k})_{k\geq 0}$ de soluciones aproximadas de residuos asociados $(\epsilon_k)_{k\geq 0}$ que verifica:*

$$\lim_{k\to+\infty} \epsilon_k = 0$$

converge uniformemente hacia y.

Demostración. Sea una secuencia $(y_{E,k})_{k\geq 0}$ de soluciones aproximadas de residuos asociados $(\epsilon_k)_{k\geq 0}$ que verifica:

$$\lim_{k\to+\infty} \epsilon_k = 0.$$

Deducimos del lema de Gronwall que $(y_{E,k})_{k\geq 0}$ es una secuencia de Cauchy en el espacio $\mathcal{C}([t_0 - T : t_0 + T])$ de funciones continuas en el compacto $[t_0 - T : t_0 + T]$ y con la norma

$$\|z\|_\infty := \max_{t\in[t_0-T:t_0+T]} |z(t)|,$$

donde $z \in \mathcal{C}([t_0 - T : t_0 + T])$. El espacio $(\mathcal{C}([t_0 - T : t_0 + T]), \|\cdot\|_\infty)$ es un espacio de Banach [14]. Entonces, $(y_{E,k})_{k\geq 0}$ converge uniformemente hacia un límite $y^\star$. Aun utilizando la proposición 4 se comprueba que $y^\star$ es solución del problema de Cauchy. Esto nos da la existencia. Ahora, si tomamos una solución y cualquiera del problema de Cauchy, se puede también considerar como una solución aproximada con residuo $\epsilon_2 = 0$. Del lema de Gronwall deducimos

$$|y_{E,k}(t) - y(t)| \leq \frac{\epsilon_k}{K} \left(\exp(K|t - t_0|) - 1\right).$$

Entonces, $(y_{E,k})_{k\geq 0}$ converge también uniformemente hacia $y : y = y^\star$. Esto establece la unicidad de la solución al problema de Cauchy. □

Observación 2. *Si, por ejemplo, f es derivable según y con la derivada parcial en y continua en el cilindro C, entonces, esta misma derivada es acotada: existe $K \geq 0$ que verifica*

$$\left|\frac{\partial f}{\partial y}(t,y)\right| \leq K$$

para todo $(t,y) \in C$. Por el teorema de Lagrange (o teorema del valor medio), la condición (2.11) *se cumple. Es una forma fácil y usual de obtenerla.*

Observación 3. *Otra forma clásica de mostrar existencia y unicidad de soluciones es mediante el teorema de iteraciones sucesivas de Picard o teorema de punto fijo de Banach [14]. Para pruebas que se hacen de esta forma ver, por ejemplo, los libros de J. P. Demailly [17] y de E. Süli y D. F. Mayers [60, Theorem 12.1]. Notar que esta forma de demostrar el teorema de Cauchy-Lipschitz / Picard-Lindelhöf es también constructiva, tal como la que hicimos anteriormente con las aproximaciones de Euler. Constructiva se tiene que entender en el sentido que explicitamos aproximaciones de la solución, que se pueden calcular de forma efectiva, y demostramos que estas aproximaciones iteradas tienen como límite la solución. Es un proceso diferente que en otros resultados, como el punto fijo de Brouwer, donde la existencia se demuestra por contradicción [14].*

2.4. Existencia global

Bajo más hipótesis en f se pueden alcanzar resultados de existencia global: la solución $t \mapsto y(t)$ del problema de Cauchy existe para todo $t \in \mathbb{R}$, o al menos para todo $t \in [0, +\infty)$. Por ejemplo, si la función f es globalmente acotada en $\mathbb{R}^2$, con

$$M = \sup_{(t,y)\in\mathbb{R}^2} |f(t,y)|,$$

entonces, para todo $T > 0$, la condición (2.4) se verifica si tomamos r_0 y T_0 suficientemente grandes. Obviamente, puede ser demasiado exigir que f sea globalmente acotada. Un resultado un poco más general de existencia global es el siguiente.

Teorema 5. *Sea $f : \mathbb{R}^2 \to \mathbb{R}$ una función continua y globalmente Lipschitz en y: existe $K > 0$ tal que*

$$|f(t, y_1) - f(t, y_2)| \leq K|y_1 - y_2|,$$

para todo $t, y_1, y_2 \in \mathbb{R}$. Sea $(t_0, y_0) \in \mathbb{R}^2$ y supongamos que $f(\cdot, y_0)$ es globalmente acotada: existe $L > 0$ tal que

$$|f(t, y_0)| \leq L$$

para todo $t \in \mathbb{R}$. Entonces, existe una única solución y, definida en todo $\mathbb{R}$, del problema de Cauchy $y' = f(t, y)$ con $y(t_0) = y_0$.

No vamos a demostrar este resultado. Se obtiene mediante una adaptación directa del teorema de Picard tal como se describe en el capítulo 12 del libro de E. Süli y D. F. Mayers [60, Theorem 12.1]. Veamos unos ejemplos donde se puede aplicar.

Ejemplo 2. *Como primer ejemplo, tomamos $a, b, y_0 \in \mathbb{R}$ y el problema de Cauchy*

$$y' = ay + b, \quad y(0) = y_0.$$

Podemos aplicar el teorema 5 con

$$K = |a|, \quad L = |ay_0 + b|,$$

para asegurar que existe una única solución y en todo $\mathbb{R}$.

Ejemplo 3. *Sea el problema de Cauchy*

$$y'(t) = |y(t)| + \operatorname{sen}(y(t)) + \exp\left(-\frac{1}{2}t^2\right), \ t \in \mathbb{R}, \qquad y(0) = 1.$$

Aquí tenemos $f(t, y) = |y| + \operatorname{sen}(y) + \exp(-1/2t^2)$. Tomamos $y_1, y_2 \in \mathbb{R}$ y calculamos:

$$\begin{aligned} & f(t, y_1) - f(t, y_2) \\ = \ & |y_1| + \operatorname{sen}(y_1) + \exp\left(-1/2t^2\right) - \left(|y_2| + \operatorname{sen}(y_2) + \exp\left(-1/2t^2\right)\right) \\ = \ & |y_1| - |y_2| + \operatorname{sen}(y_1) - \operatorname{sen}(y_2). \end{aligned}$$

Primero hacemos:

$$||y_1| - |y_2|| \leq |y_1 - y_2|.$$

Luego utilizamos

$$\operatorname{sen}(y_1) - \operatorname{sen}(y_2) = \int_{y_2}^{y_1} \cos(t)\mathrm{d}t,$$

para tener:

$$|\operatorname{sen}(y_1) - \operatorname{sen}(y_2)| \leq \int_{y_2}^{y_1} |\cos(t)|\mathrm{d}t \leq |y_1 - y_2|.$$

Finalmente, lo anterior da:

$$|f(t, y_1) - f(t, y_2)| \leq 2|y_1 - y_2|,$$

para $y_1, y_2 \in \mathbb{R}$. *Además, tenemos*

$$|f(t, 1)| \leq 3.$$

Se puede, de nuevo, aplicar el teorema 5, con $K = 2$ *y* $L = 3$, *para obtener una única solución global y en todo* $\mathbb{R}$.

Ejemplo 4. *Consideramos, finalmente, el problema de Cauchy siguiente:*

$$y'(t) = \frac{y(t)^2}{1 + y(t)^2} + \cos(t^2), \qquad t \in \mathbb{R}, \qquad y(0) = 1.$$

Esto corresponde a:

$$f(t, y) = \frac{y^2}{1 + y^2} + \cos(t^2).$$

Calculamos

$$\frac{\partial f}{\partial y}(t, y) = \frac{2y}{(1 + y^2)^2},$$

y luego

$$\frac{\partial^2 f}{\partial y^2}(t, y) = \frac{2(1 - 3y^2)}{(1 + y^2)^3}.$$

Se deduce que la derivada $\partial f/\partial y$ admite un mínimo en $-\sqrt{3}/3$, que tiene como valor $-3\sqrt{3}/8$, y (por simetría) un máximo en $\sqrt{3}/3$, que tiene como valor $3\sqrt{3}/8$. Entonces, utilizando el teorema de Lagrange, se obtiene:

$$\left|\frac{\partial f}{\partial y}(t,y)\right| \leq 3\sqrt{3}/8.$$

Por otro lado, tenemos

$$|f(t,1)| \leq \frac{3}{2}$$

para todo $t \in \mathbb{R}$. Aplicamos el teorema 5, con $K = 3\sqrt{3}/8$ y $L = 3/2$, y el problema de Cauchy tiene una única solución global y en todo $\mathbb{R}$.

Observación 4. *Estudiando los casos $|y| \leq 1$ y $|y| > 1$, se obtiene directamente:*

$$\left|\frac{\partial f}{\partial y}(t,y)\right| \leq 2.$$

Comprobar la existencia y unicidad es un poco más fácil así, a cambio de una cuota menos precisa ($K = 2$).

Observación 5. *En el ejemplo anterior, podemos mostrar que f es Lipschitz en y directamente sin hacer cálculo de derivadas.*

Sea $x, y \in \mathbb{R}$, tenemos

$$\frac{x^2}{1+x^2} - \frac{y^2}{1+y^2} = \frac{(1+y^2)x^2 - (1+x^2)y^2}{(1+x^2)(1+y^2)} = \frac{x^2-y^2}{(1+x^2)(1+y^2)}$$

y, entonces,

$$\frac{x^2}{1+x^2} - \frac{y^2}{1+y^2} = \frac{(x-y)(x+y)}{(1+x^2)(1+y^2)}.$$

Luego, utilizamos

$$x \leq \frac{1}{2}(1+x^2), \quad y \leq \frac{1}{2}(1+y^2)$$

para deducir

$$\frac{x+y}{(1+x^2)(1+y^2)} \leq \frac{1}{2}\frac{(1+x^2)+(1+y^2)}{(1+x^2)(1+y^2)} = \frac{1}{2}\left(\frac{1}{1+y^2} + \frac{1}{1+x^2}\right) \leq 1.$$

De lo anterior viene

$$\left| \frac{x^2}{1+x^2} - \frac{y^2}{1+y^2} \right| \leq |x-y|.$$

Finalmente, obtenemos $K = 1$. Otra vez, la estimación fina de los extremos de la derivada da un mejor valor de K, pero no es tan importante dado nuestro propósito (mostrar existencia y unicidad).

Notar que en el libro de J. P. Demailly [17] se encuentran, igualmente, teoremas que permiten obtener una solución global. Finalmente, el ejemplo siguiente [60] ilustra que para una función f localmente Lipschitz en y, pero no globalmente Lipschitz, no hay necesariamente existencia global:

$$y' = y^2, \quad y(0) = 1,$$

es un problema de Cauchy que tiene como solución exacta

$$y(t) = \frac{1}{1-t},$$

definida en $[0, T]$, $T < 1$.

2.5. Preparación para lo que sigue

En los tres capítulos siguientes siempre nos va a interesar un problema de Cauchy escalar de tipo:

$$y'(t) = f(t, y(t)), \quad 0 \leq t \leq T, \quad y(0) = y_0.$$

En otros términos, para simplificar lo anterior, tomamos $t_0 = 0$ y no tomamos más un intervalo centrado en t_0. Adaptaciones al caso más general se podrán hacer sin problemas. También vamos a suponer, en general, que f y T permiten satisfacer las condiciones del teorema de Cauchy-Lipschitz. De hecho, si no hay existencia o unicidad de la solución del problema de Cauchy, no es nada claro lo que puede producir el método numérico. Particularmente, vamos a suponer la mayor parte del tiempo que f es localmente Lipschitz en y, porque esto va a tener un rol importante para asegurar que los métodos numéricos son estables.

Mencionamos, finalmente, que los resultados anteriores, tal como los métodos numéricos presentados en los próximos capítulos, se extienden sin mayor problema a sistemas de varias ecuaciones, donde

$$y : [0, T] \to \mathbb{R}^d,$$

con $d \geq 1$ arbitrario. Ver, por ejemplo, [17]. Recordamos que ecuaciones diferenciales de orden dos o más siempre se pueden transformar en sistemas de orden uno equivalentes, por ejemplo:

$$y"(t) = f(t, y(t), y'(t)),$$

es equivalente a

$$z'(t) = f(t, y(t), z(t)), \quad y'(t) = z(t),$$

y los problemas de Cauchy asociados tienen dos condiciones iniciales sobre y y su derivada.

2.6. Referencias clásicas

Los libros de E. A. Coddington y N. Levinson [15] y de J. P. Demailly [17] pueden ser una buena introducción al tema de ecuaciones diferenciales. Podemos mencionar, además, los libros de V. I. Arnold [1] y de M. W. Hirsch, S. Smale y R. L. Devaney [42], [43], más orientados hacia sistemas dinámicos. Para una visión más enfocada en la historia de la matemática se puede consultar el libro de E. Hairer y G. Wanner [38].

Capítulo 3
Métodos explícitos de un paso

Comenzamos con la discretización y la aproximación numérica mediante métodos explícitos de un paso, los cuales son los más básicos y populares para las ecuaciones diferenciales ordinarias. Nos enfocaremos, particularmente, en los métodos de Runge-Kutta (explícitos), ampliamente utilizados y estudiados. En este contexto, volveremos a encontrar el método de Euler explícito como un caso particular. Veremos cómo estos métodos se implementan. Además, estudiaremos sus propiedades básicas de estabilidad y convergencia. Finalmente, abordaremos algunas nociones sobre los errores de redondeo y la adaptación del paso.

3.1. Definición y primeros ejemplos

Queremos aproximar numéricamente la solución $y : [0, T] \to \mathbb{R}$ del problema de Cauchy

$$y' = f(t, y), \quad 0 \leq t \leq T, \quad y(0) = y_0, \tag{3.1}$$

donde $T > 0$ y $f : [0, T] \times \mathbb{R} \to \mathbb{R}$ es una función. La extensión al caso vectorial se realiza sin dificultad.

Sea $\tau > 0$ un paso de tiempo. Sea $(t_n)_{0 \leq n \leq N}$ una subdivisión del intervalo $[0, T]$ que verifica:

$$\tau = t_{n+1} - t_n, \tag{3.2}$$

para todo $0 \leq n < N$. Vamos a calcular iterativamente una secuencia de reales $(y_n)_{0 \leq n \leq N}$, de tal forma que $y_n \simeq y(t_n)$, donde $y(t_n)$ es la solución exacta (pero desconocida) de (3.1) evaluada en el tiempo t_n.

Definimos un *método de un paso* como aquel que permite calcular y_{n+1} únicamente a partir del iterado anterior y_n (como en el método de Euler explícito). En contraste, un *método multipasos* de $k \geq 2$ pasos utiliza los

valores anteriores $y_n, y_{n-1}, \ldots, y_{n-k+1}$. El método es *explícito* si el cálculo de y_{n+1} se hace directamente vía fórmulas, sin resolver ninguna ecuación algebraica ni ningún sistema de ecuaciones. Si no es el caso, es *implícito* (ver el capítulo 5). Un método de un paso tiene la forma:

$$y_{n+1} = y_n + \tau F(t_n, y_n, \tau), \tag{3.3}$$

donde F es una función continua. Veamos primero unos ejemplos.

Ejemplo 5. *El método de Euler explícito*

$$y_{n+1} = y_n + \tau f(t_n, y_n)$$

es un método de un paso, que corresponde a

$$F(t, y, \tau) = f(t, y).$$

En este caso, F no depende de τ. El método de Euler explícito se puede obtener con diferencias finitas o interpolación lineal como lo hemos visto anteriormente. También se puede obtener utilizando integración numérica [16], [17]. De hecho, si integramos la ecuación diferencial en (3.1) *a través de un paso de tiempo τ, llegamos a:*

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\mathrm{d}t \simeq y(t_n) + \tau f(t_n, y(t_n)).$$

Luego, sustituimos la solución exacta $y(t_n)$ por su aproximación y_n. En términos de integración numérica, viene del método de rectángulos a la izquierda.

Ejemplo 6. *El método de Heun corresponde a*

$$F(t, y, \tau) = \frac{1}{2}\left(f(t, y) + f(t + \tau, y + \tau f(t, y))\right).$$

Se puede obtener mediante integración numérica y el método de trapecios. Hacemos

$$\begin{aligned} y(t_{n+1}) &= y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\mathrm{d}t \\ &\simeq y(t_n) + \frac{\tau}{2}\left(f(t_n, y(t_n)) + f(t_{n+1}, y(t_{n+1}))\right) \end{aligned}$$

y luego la aproximación $y(t_{n+1}) \simeq y_n + \tau f(t_n, y_n)$ *en el último término a la derecha, para volver a encontrar un método explícito. La expresión final del método es*

$$y_{n+1} = y_n + \frac{\tau}{2}\left(f(t_n, y_n) + f(t_n + \tau, y_n + \tau f(t_n, y_n))\right).$$

Ejemplo 7. *El método del punto medio explícito viene con*

$$F(t, y, \tau) = f\left(t + \frac{\tau}{2}, y + \frac{\tau}{2} f(t, y)\right).$$

De nuevo, se puede derivar mediante integración numérica y el método del punto medio. Hacemos

$$\begin{aligned} y(t_{n+1}) &= y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\mathrm{d}t \\ &\simeq y(t_n) + \tau f\left(t_n + \frac{\tau}{2}, y\left(t_n + \frac{\tau}{2}\right)\right) \end{aligned}$$

y luego la aproximación $y(t_n + \tau/2) \simeq y(t_n) + \tau/2 f(t_n, y(t_n))$ *para volver a encontrar un método explícito. La expresión final del método es*

$$y_{n+1} = y_n + \tau f\left(t_n + \frac{\tau}{2}, y_n + \frac{\tau}{2} f(t_n, y_n)\right).$$

3.2. Convergencia

En esta sección vamos a hacer el análisis numérico general de métodos de un paso, y mostrar que bajo unas condiciones mínimas la solución obtenida con un método de un paso tiende hacia la solución exacta del problema de Cauchy cuando el paso de tiempo τ tiende hacia cero. Los puntos claves consisten en estudiar la consistencia y la estabilidad del método.

3.2.1. Error de consistencia

Ahora precisamos definir propiedades fundamentales de consistencia y estabilidad. Luego veremos que la convergencia es una consecuencia lógica de ambas. Empecemos por definir la consistencia.

Definición 2 (error de consistencia)**.** *El error de consistencia asociada al método de un paso* (3.3) *es:*

$$e_n := y(t_{n+1}) - (y(t_n) + \tau F(t_n, y(t_n), \tau)), \tag{3.4}$$

para todo $0 \leq n < N$*, donde* $y(t_{n+1})$ *es la solución exacta de* (3.1) *en* $t = t_{n+1}$*, y donde* $y(t_n)$ *es la solución exacta de* (3.1) *en* $t = t_n$*.*

Veamos ahora cómo se calcula en el caso de Euler explícito.

Ejemplo 8. *Supongamos, por ejemplo, que la solución* y *de* (3.1) *pertenece a* $\mathcal{C}^2([0,T])$ *(dos veces derivable con derivada y derivada segunda continuas en* $[0,T]$*). El error de consistencia asociada al método de Euler explícito es:*

$$\begin{aligned} e_n &= y(t_{n+1}) - \big(y(t_n) + \tau f(t_n, y(t_n))\big) \\ &= y(t_n) + \tau y'(t_n) + \frac{1}{2}\tau^2 y''(t_n) + \mathcal{O}(\tau^3) - \big(y(t_n) + \tau y'(t_n)\big) \\ &= \frac{1}{2}\tau^2 y''(t_n) + \mathcal{O}(\tau^3). \end{aligned}$$

Hemos utilizado, primero, la definición del error de consistencia y la ecuación diferencial en (3.1)*. Después, hicimos un desarrollo de Taylor-Young al orden 2 en* t_n*. Recordamos que la notación de Landau tiene el significado siguiente: para dos funciones arbitrarias* f *y* g*, tenemos la relación*

$$f(\tau) = \mathcal{O}(g(\tau))$$

si y solo si

$$\lim_{\tau \to 0} \frac{f(\tau)}{g(\tau)} = c$$

con $c \in \mathbb{R}$ *(el límite existe y es finito).*

Observamos aquí la dependencia en τ del error de consistencia y, particularmente, que se comporta como $e_n = \mathcal{O}(\tau^2)$: cuando τ disminuye, la consistencia disminuye también como una función cuadrática de τ. Este comportamiento va a tener un rol clave en probar la convergencia del método. Además, se puede mostrar, utilizando una técnica similar, que para los métodos de Heun y del punto medio explícito obtenemos

$e_n = \mathcal{O}(\tau^3)$. Veremos después una técnica más expedita para comprobarlo. A continuación vamos a definir la noción de método de un paso consistente.

Definición 3 (método consistente)**.** *Un método de un paso* (3.3) *es consistente si* $y_0 = y(0)$ *y si, para toda solución exacta y del problema de Cauchy* (3.1)*, la suma de los errores de consistencia satisface:*

$$\sum_{n=0}^{N-1} |e_n| \to 0 \quad \textit{cuando } \tau \to 0. \tag{3.5}$$

3.2.2. Condición de consistencia

Veamos aquí un resultado útil que permite asegurar, sin mayor esfuerzo, si un método es consistente o no. De hecho, es una condición necesaria y suficiente de consistencia.

Teorema 6 (caracterización de la consistencia)**.** *Supongamos que* f *y* F *son funciones continuas en sus dominios de definición. Un método de un paso* (3.3) *es consistente si y solo si para todo* $t \in [0, T]$ *y para todo* $y \in \mathbb{R}$*, tales que* $f(t, y)$ *sea definido, verificamos:*

$$F(t, y, 0) = f(t, y). \tag{3.6}$$

Demostración. Sea $0 \leq n < N$ y sea y la solución de (3.1) con el error de consistencia asociado:

$$e_n = y(t_{n+1}) - y(t_n) - \tau F(t_n, y(t_n), \tau). \tag{3.7}$$

Utilizando el teorema del valor medio, sabemos que existe $c_n \in (t_n; t_{n+1})$ tal que:

$$y(t_{n+1}) - y(t_n) = \tau y'(c_n) = \tau f(c_n, y(c_n)). \tag{3.8}$$

Entonces,

$$e_n = \tau\big(f(c_n, y(c_n)) - F(t_n, y(t_n), \tau)\big) = \tau(a_n + b_n) \tag{3.9}$$

con

$$a_n = f(c_n, y(c_n)) - F(c_n, y(c_n), 0)$$

y

$$b_n = F(c_n, y(c_n), 0) - F(t_n, y(t_n), \tau).$$

Podemos, sin perder generalidad, suponer $\tau \leq 1$ (nos interesan valores pequeños de τ). Como $(t, \tau) \to F(t, y(t), \tau)$ es continua en el compacto $[0, T] \times [0, 1]$, entonces, es continua uniforme: sea $\varepsilon > 0$, existe $\eta > 0$ tal que para todo $\tau \leq \eta$ y para todo $0 \leq n < N$: $|b_n| \leq \varepsilon$ (dado que también $|c_n - t_n| \leq \tau$). De esto podemos deducir:

$$\left| \sum_{n=0}^{N-1} |e_n| - \tau \sum_{n=0}^{N-1} |a_n| \right| \leq \tau \sum_{n=0}^{N-1} |b_n| \leq \tau \sum_{n=0}^{N-1} \varepsilon = \tau N \varepsilon = T\varepsilon.$$

Lo que implica:

$$\lim_{\tau \to 0} \sum_{n=0}^{N-1} |e_n| = \lim_{\tau \to 0} \tau \sum_{n=0}^{N-1} |a_n|.$$

Por definición de la integral de Riemann tenemos, por otro lado:

$$\lim_{\tau \to 0} \tau \sum_{n=0}^{N-1} |a_n| = \int_0^T |f(t, y(t)) - F(t, y(t), 0)| \mathrm{d}t.$$

De ahí deducimos la condición necesaria y suficente del teorema. □

Ejemplo 9. *Los métodos introducidos anteriormente en los ejemplos 5, 6 y 7 son todos consistentes, dado que en cada caso se verifica:*

$$F(t, y, 0) = f(t, y).$$

3.2.3. Estabilidad

Introducimos una primera noción de estabilidad para un método de un paso. Esta noción implica, en cierto sentido, que el método no amplifica demasiado pequeños errores que se pueden introducir a cada iteración. Esta noción de estabilidad no es muy exigente, y veremos en el capítulo 5 una noción más fuerte que es imprescindible para cierta categoría de ecuaciones diferenciales. Sin embargo, va a ser suficiente para asegurar que la aproximación converge hacia la solución exacta cuando refinamos el paso de tiempo.

Definición 4 (estabilidad)**.** *Un método de un paso* (3.3) *es estable si verifica lo siguiente. Sea* $\tau_M > 0$*. Existe una constante de estabilidad* $C_s > 0$ *tal que, para todo valor de* $\tau \leq \tau_M$ *(o equivalentemente* $N \geq T/\tau_M$*), y para toda pareja de soluciones* $(y_n)_{0\leq n\leq N}$ *y* $(\widetilde{y}_n)_{0\leq n\leq N}$ *definidas de la siguiente forma:*

$$\begin{aligned} y_{n+1} &= y_n + \tau F(t_n, y_n, \tau), \\ \widetilde{y}_{n+1} &= \widetilde{y}_n + \tau F(t_n, \widetilde{y}_n, \tau) + \epsilon_n, \end{aligned}$$

para todo $0 \leq n < N$*, a partir de reales* y_0 *y* $\widetilde{y}_0$*, y con* $(\epsilon_n)_{0\leq n<N}$ *una secuencia arbitraria de reales, se cumple la desigualdad:*

$$\max_{n=0,\ldots,N} |\widetilde{y}_n - y_n| \leq C_s \left(|\widetilde{y}_0 - y_0| + \sum_{n=0}^{N-1} |\epsilon_n| \right). \tag{3.10}$$

Observación 6. *Aclaramos que la constante de estabilidad* C_s *tiene que ser independiente de* τ*, o de* N*, para que la definición tenga sentido. Por otro lado, y como veremos, esta constante depende del problema de Cauchy que se quiere resolver y del método numérico. Particularmente,* C_s *depende, en general, del tiempo final* T*.*

3.2.4. Condición de estabilidad

Ahora nos interesa una forma simple de caracterizar le estabilidad para un método de un paso. Empezamos con una variante del lema de Gronwall, visto anteriormente en el capítulo 2.

Lema 3 (Gronwall discreto)**.** *Sea* $\kappa \geq 0$ *y dos secuencias de reales* $(\theta_n)_{n\geq 0}$*,* $(\epsilon_n)_{n\geq 0}$ *que verifican, para todo* $n \geq 0$*:*

$$|\theta_{n+1}| \leq (1+\kappa)|\theta_n| + |\epsilon_n|. \tag{3.11}$$

Como consecuencia, se cumple, para todo $n \geq 0$*:*

$$|\theta_n| \leq e^{\kappa n}|\theta_0| + \sum_{i=1}^{n} e^{\kappa(n-i)}|\epsilon_{i-1}|. \tag{3.12}$$

Demostración. Se puede hacer por inducción. Empezamos con $n = 0$ y verificamos que $\theta_0 = \theta_0$. Supongamos que la desigualdad (3.12) se verifica en $n \geq 0$. Como la función exponencial es convexa, tenemos:

$$1 + \kappa \leq e^{\kappa}.$$

Entonces, de (3.11) deducimos

$$|\theta_{n+1}| \leq e^{\kappa}|\theta_n| + |\epsilon_n|.$$

Aplicamos (3.12) en n:

$$|\theta_{n+1}| \leq e^{\kappa}\left(e^{\kappa n}|\theta_0| + \sum_{i=1}^{n} e^{\kappa(n-i)}|\epsilon_{i-1}|\right) + |\epsilon_n|.$$

Además,

$$e^{\kappa}\sum_{i=1}^{n} e^{\kappa(n-i)}|\epsilon_{i-1}| + |\epsilon_n| = \sum_{i=1}^{n} e^{\kappa(n+1-i)}|\epsilon_{i-1}| + e^{\kappa(n+1-n+1)}|\epsilon_n|.$$

Lo que permite establecer

$$|\theta_{n+1}| \leq e^{\kappa(n+1)}|\theta_0| + \sum_{i=1}^{n+1} e^{\kappa(n+1-i)}|\epsilon_{i-1}|.$$

Encontramos exactamente la desigualdad (3.12) para $n + 1$. □

Lo anterior nos va a permitir establecer la siguiente condición suficiente de estabilidad.

Teorema 7. *Sea un método de un paso definido por* (3.3). *Sea* $\tau_M > 0$. *Supongamos que la función* F *en* (3.3) *asociada al método sea Lipschitz en* y: *existe* $K_F \geq 0$ *tal que, para todo* $0 < \tau \leq \tau_M$, *para todo* $t \in [0, T]$ *y para todo* $(y_1, y_2) \in \mathbb{R}^2$, *tengamos:*

$$|F(t, y_1, \tau) - F(t, y_2, \tau)| \leq K_F|y_1 - y_2|. \tag{3.13}$$

Entonces, el método de un paso (3.3) *es estable en el sentido de la definición 4. Además, se puede tomar* $C_s = e^{K_F T}$ *como constante de estabilidad en la desigualdad* (3.10).

Demostración. Sea $0 \leq n < N$. Consideramos las siguientes secuencias de la definición 4:

$$y_{n+1} = y_n + \tau F(t_n, y_n, \tau),$$
$$\widetilde{y}_{n+1} = \widetilde{y}_n + \tau F(t_n, \widetilde{y}_n, \tau) + \epsilon_n,$$

restamos y luego aplicamos la condición (3.13) (F Lipschitz en y):

$$|y_{n+1} - \widetilde{y}_{n+1}| \leq (1 + \tau K_F)|y_n - \tilde{y}_n| + |\epsilon_n|.$$

Como cumplimos con (3.11), con $\kappa = \tau K_F$, podemos aplicar el lema de Gronwall y (3.12) se vuelve a escribir

$$|y_n - \widetilde{y}_n| \leq e^{\tau K_F n}|y_0 - \widetilde{y}_0| + \sum_{i=1}^{n} e^{\tau K_F (n-i)}|\epsilon_{i-1}|.$$

Como $n - i \leq n \leq N$ y $\tau N = T$, resulta que

$$e^{\tau K_F (n-i)} \leq e^{\tau K_F n} \leq e^{\tau K_F N} = e^{K_F T}.$$

Acotamos

$$|y_n - \widetilde{y}_n| \leq e^{K_F T}|y_0 - \widetilde{y}_0| + e^{K_F T} \sum_{i=1}^{N} |\epsilon_{i-1}|.$$

Lo que corresponde a (3.10) con $C_s = e^{K_F T}$. □

Observación 7. *En el caso anterior, la constante K_F y, entonces, C_s dependen de T y, eventualmente, de τ_M. Veremos en la práctica que, para una clase importante de métodos, K_F es polinomial en τ_M. El punto importante consiste en asegurarse de que K_F sea acotada cuando τ tiende hacia cero.*

Ejemplo 10. *Si la función f del problema* (3.1) *es K-Lipschitz, los métodos introducidos anteriormente en los ejemplos 5, 6 y 7 son todos estables. Se puede verificar fácilmente en cada caso que F es Lipschitz. En particular, en el caso de Euler explícito, tenemos $F = f$ y, entonces, $K_F = K$. Para los otros dos métodos de Heun y del punto medio explícito veremos luego un resultado más general.*

3.2.5. Convergencia

Ahora que hemos visto las propiedades básicas de consistencia y estabilidad, nos vamos a centrar primero en la convergencia del método de un paso. Definimos el error de discretización de la siguiente manera:

Definición 5. *El error de discretización asociado a un método de un paso* (3.3)*, dado un paso de tiempo* $\tau > 0$*, se define como*

$$E_\tau := \max_{0 \le n \le T/\tau} |y_n - y(t_n)|, \tag{3.14}$$

donde $y(t_n)$ *es la solución exacta de* (3.1) *y* y_n *es la solución aproximada por* (3.3)*, ambas evaluadas en el tiempo* $t_n = n\tau$*. En otras palabras,* E_τ *mide la diferencia máxima entre las soluciones exacta y aproximada en el intervalo de tiempo* $[0, T]$*.*

Observamos en lo anterior que $N = T/\tau$ y que también podríamos definir $\widetilde{E}_N = E_{T/N}$ e interesarnos en los valores de $\widetilde{E}_N$ cuando N es grande. Notar también que el error de discretización se podría definir de otra forma, utilizando otra función de distancia, por ejemplo. No obstante, hay dos motivaciones para utilizar una distancia que involucra el máximo del valor absoluto de la diferencia entre las soluciones:

1. es algo que aparece naturalmente en nuestro análisis;
2. tiene un interés práctico, sobre todo cuando nos interesamos por la calidad de la solución en tiempos específicos t_n.

Lo que nos va a interesar a continuación es cómo se comporta el error de discretización E_τ cuando τ se hace pequeño.

Estabilidad y consistencia dan convergencia

En el capítulo 2, el teorema 3 de Cauchy-Peano-Arzelà permitía garantizar que, al aproximar la solución de un problema de Cauchy con aproximaciones de Euler explícito, existe una subsucesión que converge uniformemente hacia una solución exacta del problema. Ahora veremos un resultado similar que asegura que cualquier método de un paso consistente y estable proporciona una solución aproximada que converge a

la solución exacta cuando el paso de tiempo τ tiende a cero. En otros términos, demostraremos que el error de discretización E_τ tiende a cero cuando $\tau \to 0$. Observemos que la regla *consistencia más estabilidad implican convergencia* es un clásico en análisis numérico.

Teorema 8. *Sea un método de un paso* (3.3) *consistente (definición 3) y estable (definición 4), asociado a un paso de tiempo* $\tau > 0$. *Entonces, el método es convergente:*

$$\lim_{\tau \to 0} E_\tau = 0.$$

Demostración. Dado que el error de consistencia e_n satisface (3.4):

$$y(t_{n+1}) = y(t_n) + \tau F(t_n, y_n) + e_n,$$

por estabilidad (ecuación (3.10)), el error de discretización global E_τ cumple:

$$E_\tau = \max_{n=0,\ldots,N} |y(t_n) - y_n| \leq C_s \left(|y(t_0) - y_0| + \sum_{n=0}^{N-1} |e_n| \right). \tag{3.15}$$

Se concluye con la propiedad de consistencia dada por la definición 3: $y(0) = y_0$ y

$$\sum_{n=0}^{N-1} |e_n| \to 0$$

cuando $\tau \to 0$ ($N \to +\infty$). □

Ejemplo 11. *Si la función* f *del problema de Cauchy* (3.1) *es* K*-Lipschitz, los métodos introducidos anteriormente en los ejemplos 5, 6 y 7 son todos convergentes.*

Observación 8. *Observamos que la relación* (3.15) *permite acotar el error de discretización en función de la suma de los errores de consistencia. Esto ilustra bien el mecanismo que genera el error de discretización: los errores de consistencia se acumulan a lo largo de los pasos de tiempo. La propiedad de estabilidad permite demostrar que esta acumulación del error tiene un efecto limitado sobre la precisión global.*

Observación 9. *Veremos más adelante que el error de discretización es una fuente importante de error en las simulaciones numéricas. Lamentablemente, no es la única fuente de errores, ya que se combinan con errores de modelización y errores numéricos.*

El error de modelización surge cuando la ecuación diferencial no representa con precisión el sistema físico relevante. Por ejemplo, en el caso de la ecuación del péndulo, cuando sustituimos

$$\ddot{\theta} + \operatorname{sen}(\theta) = 0$$

por

$$\ddot{\theta} + \theta = 0,$$

asumiendo que el ángulo θ es pequeño, estamos introduciendo un error de modelización. Aquí, θ es la función que representa el ángulo del péndulo en función del tiempo y $\ddot{\theta}$ es su derivada segunda.

Estos errores son relevantes en la práctica, pero su estudio riguroso está fuera del alcance de este libro. Por otro lado, los errores numéricos provienen típicamente de la representación inexacta de los números reales en una computadora, o también de las aproximaciones e iteraciones necesarias para resolver ecuaciones diferenciales (por ejemplo, la resolución de ecuaciones implícitas o de sistemas lineales). Más adelante veremos cómo se pueden manejar estos errores.

Orden de convergencia

El resultado de convergencia anterior no permite saber en la práctica a partir de qué valor de τ la aproximación es aceptable. Solo es una forma de asegurar que el método no tiene patologías. De hecho, un método que no tiene propiedades de consistencia y estabilidad produce resultados inaceptables en la práctica la mayor parte del tiempo. El paso siguiente del análisis consiste en tener tasas explícitas de convergencia. En otros términos, queremos saber cómo el error se reduce cuando τ es reducido. Particularmente, hemos visto que algunos métodos de un paso tienen un error de consistencia en τ^2 y otras en τ^3. Sería interesante averiguar que los métodos con error de consistencia de orden más alto producen una mejor aproximación en la práctica, sobre todo porque son más difíciles de implementar e involucran más cálculos.

Definición 6. *Supongamos ahora que f sea de clase $\mathcal{C}^p$, con $p \geq 0$, derivable hasta el orden p con derivadas parciales de orden p continuas en todo el dominio de definición. Decimos que un método de un paso es de orden p, o más, si el error de consistencia satisface lo siguiente. Existe una constante $C_c > 0$ tal que, para todo $\tau > 0$ y todo $n = 0, \ldots, N$, tengamos:*

$$|e_n| \leq C_c \tau^{p+1}.$$

Verificamos ahora que el método de Euler explícito es de orden uno según la definición anterior. Retomamos el cálculo anterior y hacemos un Taylor-Lagrange en vez de un Taylor-Young:

$$\begin{aligned} e_n &= y(t_{n+1}) - \big(y(t_n) + \tau f(t_n, y(t_n))\big) \\ &= y(t_n) + \tau y'(t_n) + \frac{1}{2}\tau^2 y''(\xi_n) - \big(y(t_n) + \tau y'(t_n)\big) \\ &= \frac{1}{2}\tau^2 y''(\xi_n), \end{aligned}$$

con $t_n \leq \xi_n \leq t_{n+1}$. Entonces, obtenemos

$$|e_n| \leq C_c \tau^2, \qquad C_c = \frac{1}{2} \max_{0 \leq t \leq T} |y''(t)|.$$

Se puede asegurar que $C_c < +\infty$ bajo algunas condiciones sobre f y T (ver capítulo 2).

Observación 10. *Se observa en la definición anterior que C_c tiene que ser independiente del paso de tiempo τ (o de N). No obstante, puede ser dependiente de T, de f y de los coeficientes que aparecen en el método de un paso. Veamos un ejemplo simple donde se puede acotar explícitamente C_c. Sea $\alpha \in \mathbb{R}$ y el problema de Cauchy*

$$y' = \alpha y, \quad y(0) = 1,$$

en $[0; T]$. Para todo $t \in [0; T]$ tenemos la solución exacta

$$y(t) = e^{\alpha t}.$$

Derivamos dos veces:

$$y"(t) = \alpha^2 e^{\alpha t}.$$

Entonces, en este caso tenemos

$$C_c = \frac{1}{2}\,\text{máx}\{1, e^{\alpha}T\}.$$

Veamos el resultado siguiente.

Teorema 9. *Si un método de un paso* (3.3) *es estable y consistente de orden $p \geq 1$, entonces, el error de discretización es de orden $\mathcal{O}(\tau^p)$:*

$$E_\tau \leq CT\tau^p.$$

La constante $C > 0$ anterior es independiente de τ, y depende de las constantes de estabilidad C_s y de orden de consistencia C_c.

Demostración. Acotamos primero

$$\sum_{n=0}^{N-1} |e_n| \leq C_c \sum_{n=0}^{N-1} \tau^{p+1} = C_c \left(\sum_{n=0}^{N-1} \tau \right) \tau^p = C_c T \tau^p,$$

y el resultado sigue utilizando la estabilidad (ecuación (3.10)). En efecto, el error de discretización global E_τ cumple (3.15), que se vuelve a escribir, asumiendo $y(t_0) = y_0$:

$$E_\tau \leq C_s \left(\sum_{n=0}^{N-1} |e_n| \right) \leq C_s C_c T \tau^p.$$

Observamos que $C = C_s C_c$. □

Observación 11. *Para ecuaciones diferenciales simples y Euler explícito se puede obtener una expresión explícita de la constante $C(= C_s C_c)$ (ver, por ejemplo, [53]).*

Finalmente, antes de descubrir los métodos de Runge-Kutta, derivamos una caracterización del orden que nos va a ser útil. Para esto, introducimos primero la notación siguiente [17].

Supongamos que $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ sea de clase $\mathcal{C}^p$, $p \geq 0$, y que la función y satisface la ecuación diferencial

$$y'(t) = f(t, y(t)), \quad t \in [0; T],$$

del problema de Cauchy (3.1). Derivamos una vez la ecuación diferencial y obtenemos:

$$y'' = (f_{,t} + f_{,y}f)(t, y),$$

donde $f_{,t}$, respectivamente $f_{,y}$, denota la derivada parcial de f relativa a t, respectivamente a y. Ahora notaremos $y' = f^{[0]}(t, y)$ y $y'' = f^{[1]}(t, y)$, con

$$f^{[0]} = f, \quad f^{[1]} = f_{,t} + f_{,y}f.$$

Siguiendo la misma idea, para $k \geq 0$ notamos $f^{[k]}$ la función de f y de sus distintas derivadas que satisface:

$$y^{(k+1)} = f^{[k]}.$$

Derivando la ecuación diferencial, se demuestra por inducción que tenemos, por $k \geq 1$:

$$f^{[k]} \;=\; (f^{[k-1]})_{,t} + (f^{[k-1]})_{,y}f. \tag{3.16}$$

Proposición 5. *Supongamos que f y F sean ambas de clase $\mathcal{C}^p$, $p \geq 0$. Un método de un paso* (3.3) *es de orden p, o más, si y solo si se cumple la relación siguiente:*

$$\frac{\partial^q F}{\partial \tau^q}(t, y, 0) = \frac{1}{q+1} f^{[q]}(t, y), \quad 0 \leq q \leq p-1, \tag{3.17}$$

en todo punto $(t, y) \in [0, T] \times \mathbb{R}$, donde $\partial^q F/\partial\tau^q$ denota la derivada parcial de F relativamente a τ, de orden q, y donde $f^{[q]}$ se obtiene a partir de f con la recurrencia (3.16).

Demostración. El error de consistencia es dado por la fórmula (3.4):

$$e_n = y(t_{n+1}) - y(t_n) - \tau F(t_n, y(t_n), \tau).$$

Aplicamos Taylor-Young a F en $\tau = 0$ y al orden p:

$$F(t_n, y(t_n), \tau) = \sum_{q=0}^{p} \frac{1}{q!} \tau^q \frac{\partial^q F}{\partial \tau^q}(t_n, y(t_n), 0) + o(\tau^p).$$

Por otra parte, como f es $\mathcal{C}^p$, y es $\mathcal{C}^{p+1}$ y podemos escribir:

$$y(t_{n+1}) - y(t_n) = \sum_{k=1}^{p+1} \frac{1}{k!}\tau^k \frac{\partial^k y}{\partial \tau^k}(t_n) + o(\tau^{p+1}),$$

o también

$$y(t_{n+1}) - y(t_n) = \sum_{q=0}^{p} \frac{1}{(q+1)!}\tau^{q+1} f^{[q]}(t_n, y(t_n)) + o(\tau^{p+1}).$$

Deducimos

$$e_n = \sum_{q=0}^{p} \frac{1}{(q+1)!}\tau^{q+1} f^{[q]}(t_n, y(t_n)) - \sum_{q=0}^{p} \frac{1}{q!}\tau^{q+1} \frac{\partial^q F}{\partial \tau^q}(t_n, y(t_n), 0) + o(\tau^{p+1}).$$

Entonces, si la condición (3.17) se cumple, tenemos $e_n = o(\tau^{p+1})$ y, de forma idéntica a lo anterior, podemos mostrar que existe $C_c \geq 0$ tal que $|e_n| \leq C_c \tau^{p+1}$. Por otra parte, si la condición (3.17) no se cumple, deducimos de la fórmula anterior que por $q \leq p$, $e_n = C(f, F, q)\tau^q + o(\tau^{q+1})$, entonces, el método no puede ser de orden p. □

Observación 12. *El resultado anterior implica lo siguiente: un método de un paso es consistente si y solo si es de orden $p \geq 1$.*

Observación 13. *El criterio* (3.17) *permite determinar si un método es* exactamente *de orden p. En otros términos, permite saber si es de orden p pero no cumple con el orden $p+1$. De hecho, si la condición* (3.17) *está satisfecha y también tenemos*

$$\frac{\partial^p F}{\partial \tau^p}(t, y, 0) \neq \frac{1}{p+1} f^{[p]}(t, y),$$

entonces, el método de un paso no puede ser de orden $p+1$. Por ejemplo, vamos a ver de esta forma, y con resultados numéricos, que el método de Euler explícito es exactamente de orden 1: no puede ser, en general, de orden 2 o más.

3.3. Métodos de Runge-Kutta

Los métodos de Runge-Kutta son algunos de los más utilizados para resolver ecuaciones diferenciales ordinarias (EDO). Su relevancia persiste en bibliotecas modernas de cómputo científico, como SciPy (Python) y DifferentialEquations.jl (Julia). Aunque son métodos explícitos, ofrecen una precisión significativamente mayor sin incrementar excesivamente la complejidad computacional.

Estos métodos se basan en el método de Euler, propuesto originalmente por Leonhard Euler en 1768. La idea central es componer múltiples pasos del método de Euler para aumentar el orden de precisión. Esta técnica fue introducida por Carl Runge en 1895 [54] y desarrollada posteriormente por Karl Heun en 1900 [41]. La formulación general de los métodos de Runge-Kutta fue presentada por primera vez en el trabajo de Wilhelm Kutta en 1901 [48]. Durante el siglo XX, John C. Butcher realizó un estudio sistemático y exhaustivo de estos métodos [6], consolidando su teoría y aplicaciones.

Para un análisis detallado, incluyendo notas históricas y desarrollos teóricos, se recomienda consultar el texto de Hairer, Nørsett y Wanner [37].

3.3.1. Idea del método

Describimos primero el método. Seguimos la presentación de [17]. Sea $q \geq 1$ y unos $c_i \in [0,1]$ para $i = 1, \ldots, q$. Sean en cada intervalo $[t_n; t_{n+1}]$ unos puntos intermedios

$$t_i (= t_{n,i}) = t_n + c_i \tau.$$

Introducimos

$$p_i = f(t_i, y_i).$$

La solución exacta y del problema de Cauchy (3.1) verifica:

$$y(t_i) - y(t_n) = \int_{t_n}^{t_i} f(t, y(t)) \mathrm{d}t.$$

Hacemos el cambio de variable $t = t_n + \tau s$, notamos

$$\widetilde{f}(s) := f(t_n + s\tau, y(t_n + s\tau))$$

y obtenemos

$$y(t_i) - y(t_n) = \tau \int_0^{c_i} \widetilde{f}(s)\mathrm{d}s.$$

De la misma forma podemos escribir:

$$y(t_{n+1}) - y(t_n) = \tau \int_0^{1} \widetilde{f}(s)\mathrm{d}s.$$

Para cada $i = 1, \ldots, q$ hacemos una aproximación de la siguiente forma, basada en cualquier método de integración numérica:

$$\int_0^{c_i} \widetilde{f}(s)\mathrm{d}s \simeq \sum_{1\leq j<i} a_{ij}\widetilde{f}(c_j), \tag{3.18}$$

donde $(a_{ij})_{1\leq j<i}$ son coeficientes específicos. Veremos luego cómo se pueden elegir. De la misma forma podemos escribir:

$$\int_0^{1} \widetilde{f}(s)\mathrm{d}s \simeq \sum_{1\leq j\leq q} b_j\widetilde{f}(c_j). \tag{3.19}$$

De nuevo, $(b_j)_{1\leq j\leq q}$ son coeficientes específicos. Esto permite obtener el método de Runge-Kutta siguiente:

$$\begin{aligned} t_i &= t_n + c_i\tau, \\ y_i &= y_n + \tau \sum_{1\leq j<i} a_{ij}p_j, \\ p_i &= f(t_i, y_i), \\ t_{n+1} &= t_n + \tau, \\ y_{n+1} &= y_n + \tau \sum_{1\leq j\leq q} b_j p_j. \end{aligned}$$

Antes de estudiarlo, veamos cómo se puede implementar.

3.3.2. Aspectos de implementación

Vamos a dar ahora más detalles sobre su implementación.

Tabla de Butcher

Una forma cómoda de describir un método de Runge-Kutta de q etapas es representarlo con una tabla que da los coeficientes (tabla de Butcher):

$$\begin{array}{c|cccc} c_1 & a_{11} & 0 & \cdots & 0 \\ \vdots & \vdots & a_{22} & \ddots & \vdots \\ c_q & a_{q1} & \cdots & \cdots & 0 \\ \hline & b_1 & \cdots & \cdots & b_q \end{array} \tag{3.20}$$

donde:

- q es el número de etapas del método;
- (a_{ij}) es la matriz de coeficientes;
- $b = (b_1, \ldots, b_q)$ son los pesos de la combinación lineal final;
- $c = (c_1, \ldots, c_q)$ son los nodos de evaluación intermedia.

Algoritmo

Dado un paso de tiempo τ, el esquema de Runge-Kutta avanza de y_n a y_{n+1} como sigue:

1. se calculan los valores p_i para $i = 1, \ldots, q$:

$$p_i = f\left(t_n + c_i\tau, y_n + \tau \sum_{j=1}^{i-1} a_{ij} p_j\right); \tag{3.21}$$

2. se actualiza la solución:

$$y_{n+1} = y_n + \tau \sum_{j=1}^{s} b_j p_j. \tag{3.22}$$

Ejemplo (Heun)

Para $q = 2$, el método de Runge-Kutta (RK2) se puede escribir con una tabla de Butcher:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ 1 & 1 & 0 \\ \hline & 1/2 & 1/2 \end{array}$$

El esquema de actualización correspondiente es:

$$\begin{aligned} p_1 &= f(t_n, y_n), \\ p_2 &= f(t_n + \tau, y_n + \tau p_1), \\ y_{n+1} &= y_n + \frac{\tau}{2}(p_1 + p_2). \end{aligned}$$

Volvemos a encontrar exactamente el método de Heun presentado en el ejemplo 6. Para $q = 1$, volvemos a encontrar Euler explícito.

Python

A continuación, damos una implementación genérica de Runge-Kutta en Python.

```
import numpy as np
import matplotlib.pyplot as plt

def runge_kutta_butcher(f, y0, t0, T, N, A, b, c):
#     Resuelve la ecuación diferencial y' = f(t, y)
# usando un método de Runge-Kutta
#     definido por la tabla de Butcher.

#     Parámetros:
#     f  : función derivada f(t, y)
#     y0 : condición inicial
#     t0 : tiempo inicial
#     T  : tiempo final
#     N  : cantidad de pasos de tiempo
#     A  : matriz de coeficientes de la tabla de Butcher
#     b  : pesos para actualizar y_n
#     c  : nodos de evaluación intermedia

#     Retorna:
#     t  : array con los valores de tiempo
```

```
21 #     y  : array con las aproximaciones de la solución
22
23     q = len(b)  # Etapas del método
24     # Intervalo
25     t = np.linspace(t0, T, N+1)
26     # Array para la solución numérica
27     y = np.zeros(N+1)
28     # Asignamos la condición inicial
29     y[0] = y0
30     # Tamaño de paso
31     tau = (T - t0) / N
32
33     for n in range(N):
34         tn, yn = t[n], y[n]
35         p = np.zeros(s)  # Intermedios p_i
36
37         # Calculamos los p_i
38         # de acuerdo a la tabla de Butcher
39         for i in range(q):
40             ti = tn + c[i] * tau
41             yi = yn + tau * np.sum(A[i, :i] * p[:i])
42             p[i] = f(ti, yi)
43
44         # Actualizamos y_n usando los p_i
45         y[n+1] = yn + tau * np.sum(b * p)
46
47     return t, y
48
49 # ejemplo de uso: método de Heun (RK2)
50 def f(t, y):
51     return -y + np.sin(t)
52     # Definimos la ecuación diferencial
53
54 # Parámetros
55 T, N = 10, 100    # Tiempo final y cantidad de pasos
56 y0 = 1            # Condición inicial
57
58 # Tabla de Butcher para el método de Heun (RK2)
59 A = np.array([[0, 0],
60               [1, 0]])
61 b = np.array([0.5, 0.5])
62 c = np.array([0, 1])
63
64 # Llamamos al solver
```

```
t, y = runge_kutta_butcher(f, y0, 0, T, N, A, b, c)

# Graficamos la solución
plt.plot(t, y, label='RK (Heun)')
plt.xlabel('Tiempo t')
plt.ylabel('y(t)')
plt.legend()
plt.grid()
plt.show()
```

Listado 2: **Implementación de Runge-Kutta en Python.**

3.3.3. Estabilidad de los métodos de Runge-Kutta

Volvemos al análisis y primero estudiamos, de forma general, la estabilidad numérica de los métodos de Runge-Kutta. Podemos escribir un método de Runge-Kutta como:

$$y_{n+1} = y_n + \tau\ F(t_n, y_n, \tau),$$

con

$$\begin{aligned} F(t, y, \tau) &= \sum_{1 \le j \le q} b_j f(t + c_j\tau, y_j) \\ y_i &= y + \tau \sum_{1 \le j < i} a_{ij} f(t + c_j\tau, y_j). \end{aligned} \tag{3.23}$$

Supongamos que f es K-Lipschitz, entonces, demostramos que F es Lipschitz y más precisamente lo siguiente.

Lema 4. *Sea*

$$\alpha := \max_{1 \le i \le q} \left(\sum_{1 \le j \le i} |a_{ij}| \right)$$

y $\Lambda := \alpha K \tau$*, entonces,*

$$|y_i - z_i| \le (1 + \Lambda + \ldots + \Lambda^{i-1})|y - z|, \tag{3.24}$$

donde los y_i*, respectivamente los* z_i*, son obtenidos a partir de* y*, respectivamente* z*, en la fórmula de Runge-Kutta* (3.23)*.*

Demostración. Para $i = 1$ tenemos $y_i = y$ y $z_i = z$, entonces, (3.24) se verifica. Supongamos $i \geq 2$ y que la relación (3.23) es válida para todo $j < i$. Entonces, (3.23) y f K-Lipschitz implican

$$|y_i - z_i| \leq |y - z| + \tau \sum_{1 \leq j < i} |a_{ij}| K |y_j - z_j|,$$

y luego

$$|y_i - z_i| \leq |y - z| + \tau K \left(\max_{1 \leq j < i} |y_j - z_j| \right) \sum_{1 \leq j < i} |a_{ij}|;$$

y, con las definiciones de α y Λ,

$$|y_i - z_i| \leq |y - z| + \Lambda \max_{1 \leq j < i} |y_j - z_j|.$$

Finalmente, basta aplicar la fórmula de recurrencia

$$|y_i - z_i| \leq |y - z| + \Lambda(1 + \Lambda + \ldots + \Lambda^{i-2})|y - z|$$

para terminar la demostración. □

Deducimos que los métodos de Runge-Kutta son estables.

Teorema 10. *Sea el método de Runge-Kutta definido por* (3.23)*. Sea* $\tau_M > 0$*. Supongamos que la función* f *en* (3.1) *sea* $K-$*Lipschitz en* y*. Entonces, la función* F *en* (3.3) *asociada al método de Runge-Kutta* (3.23) *es Lipschitz en* y*:*

$$|F(t, y, \tau) - F(t, z, \tau)| \leq K_F |y - z|, \tag{3.25}$$

con

$$K_F = K \left(\sum_{1 \leq j \leq q} |b_j| \right) (1 + \Lambda + \ldots + \Lambda^{q-1}).$$

Esto implica que el método Runge-Kutta (3.3) *es estable en el sentido de la definición 4.*

Demostración. Partamos de la definición (3.23) y acotamos:

$$|F(t,y,\tau) - F(t,z,\tau)| \leq \sum_{1\leq j\leq q} |b_j||f(t+c_j\tau, y_j) - f(t+c_j\tau, z_j)|.$$

Como f es K-Lipschitz en y:

$$|F(t,y,\tau) - F(t,z,\tau)| \leq K \sum_{1\leq j\leq q} |b_j||y_j - z_j|.$$

Aplicamos el resultado anterior (3.24) junto con $j \leq q$:

$$|F(t,y,\tau) - F(t,z,\tau)| \leq K \left(\sum_{1\leq j\leq q} |b_j| \right) (1 + \Lambda + \ldots + \Lambda^{q-1})|y - z|,$$

lo que concluye la prueba. □

3.3.4. Orden de los métodos

Estudiamos ahora cómo los coeficientes de los métodos de Runge-Kutta determinan su orden. Vamos a aplicar el resultado anterior (fórmula (3.17)).

Observación preliminar

Se suele suponer que los métodos de integración numérica (3.18) y (3.19) son de orden cero, en el sentido de que son exactos para las funciones constantes. Esto implica

$$c_i = \sum_{1\leq j<i} a_{ij}, \quad i = 1, \ldots, q \tag{3.26}$$

y

$$1 = \sum_{1\leq j\leq q} b_j. \tag{3.27}$$

Particularmente,

$$c_1 = 0, \quad t_{1,1} = t_n, \quad y_1 = y_n, \quad p_1 = f(t_n, y_n).$$

3.3.5. Condiciones de orden

Recordemos que

$$\begin{aligned} F(t,y,\tau) &= \sum_{1\le j\le q} b_j f(t+c_j\tau, y_j) \\ y_i &= y+\tau \sum_{1\le j<i} a_{ij} f(t+c_j\tau, y_j). \end{aligned}$$

Veamos cuáles son las relaciones entre los coeficientes que permiten tener métodos de diferentes órdenes.

Orden uno Tenemos

$$F(t,y,0) = \sum_{1\le j\le q} b_j f(t,y) = f(t,y) \sum_{1\le j\le q} b_j = f(t,y),$$

dada la condición (3.27). Entonces, todos los métodos son de orden uno o más (o consistentes). Como además son estables, son también convergentes.

Orden dos Derivamos:

$$\begin{aligned} \frac{\partial F}{\partial \tau}(t,y,\tau) &= \sum_{1\le j\le q} b_j \left(c_j \frac{\partial f}{\partial t}(t+c_j\tau, y_j) + \frac{\partial f}{\partial y}(t+c_j\tau, y_j) \frac{\partial y_j}{\partial \tau} \right), \\ \frac{\partial y_i}{\partial \tau} &= \sum_{1\le j<i} a_{ij} f(t+c_j\tau, y_j) \\ &\quad +\tau \sum_{1\le j<i} a_{ij} \left(c_j \frac{\partial f}{\partial t}(t+c_j\tau, y_j) + \frac{\partial f}{\partial y}(t+c_j\tau, y_j) \frac{\partial y_j}{\partial \tau} \right). \end{aligned}$$

Evaluamos lo anterior en $\tau = 0$ y luego aplicamos la condición (3.27):

$$\begin{aligned} \frac{\partial F}{\partial \tau}(t,y,0) &= \sum_{1\le j\le q} b_j \left(c_j \frac{\partial f}{\partial t}(t,y) + \frac{\partial f}{\partial y}(t,y) \frac{\partial y_j}{\partial \tau}(\tau=0) \right) \\ \frac{\partial y_i}{\partial \tau}(\tau=0) &= \sum_{1\le j<i} a_{ij} f(t,y) = c_i f(t,y). \end{aligned}$$

Combinamos los dos cálculos anteriores y viene:

$$\frac{\partial F}{\partial \tau}(t,y,0) = \sum_{1\le j\le q} b_j \left(c_j \frac{\partial f}{\partial t}(t,y) + \frac{\partial f}{\partial y}(t,y) c_j f(t,y) \right)$$

Simplificamos

$$\begin{aligned} \frac{\partial F}{\partial \tau}(t,y,0) &= \sum_{1\le j\le q} b_j c_j \left(\frac{\partial f}{\partial t}(t,y) + \frac{\partial f}{\partial y}(t,y) f(t,y) \right) \\ &= \left(\sum_{1\le j\le q} b_j c_j \right) f^{[1]}(t,y). \end{aligned}$$

La condición (3.17) escrita al orden 2 implica

$$\frac{\partial F}{\partial \tau}(t,y,0) = \frac{1}{2} f^{[1]}(t,y), \tag{3.28}$$

y obtenemos la siguiente condición sobre los coeficientes:

$$\frac{1}{2} = \sum_{1\le j\le q} b_j c_j. \tag{3.29}$$

Orden tres y más Se puede, mediante unos cálculos adicionales, conseguir condiciones para el orden tres y más. Se pueden estudiar los detalles en [17] o también en [6], [37].

3.3.6. Métodos de Runge-Kutta clásicos y órdenes

Veamos ahora algunos ejemplos precisos de métodos por orden creciente.

Métodos de un subpaso Tomamos $q = 1$ y la tabla de Butcher es:

$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

Entonces, Euler explícito es el único método de Runge-Kutta de 1 subpaso. Como $b_1 = 1$ y $b_1 c_1 = 0$, deducimos de (3.17) que es un método de orden 1 exactamente. No puede ser, en general, de orden 2.

Métodos de dos subpasos De forma general, tomando $q = 2$, la tabla de Butcher que corresponde a Runge–Kutta de dos subpasos es:

$$\begin{array}{c|cc} 0 & 0 & 0 \\ c_2 & a_{21} & 0 \\ \hline & b_1 & b_2 \end{array}$$

Para tener consistencia (u orden 1) necesitamos satisfacer

$$b_1 + b_2 = 1.$$

Supongamos también

$$c_2 = a_{21}.$$

Quedan, entonces, dos parámetros libres. Para tener un método de orden 2, la condición (3.17) impone

$$\frac{1}{2} = b_2 c_2.$$

Con lo anterior dedujimos que todos los métodos de orden 2 se escriben de la forma

$$\begin{array}{c|cc} 0 & 0 & 0 \\ \alpha & \alpha & 0 \\ \hline & 1 - \frac{1}{2\alpha} & \frac{1}{2\alpha} \end{array} \tag{3.30}$$

con α un parámetro libre. Se encuentran así los métodos de Heun, con $\alpha = 1$ y de punto medio explícito con $\alpha = 1/2$. Se puede comprobar que no se alcanza el orden 3 con $q = 2$: no hay ningún valor de α que permite cumplir las condiciones de orden 3 dadas por ejemplo en [17].

Métodos de tres subpasos o más A partir de $q \geq 3$ hay mucha más libertad para tener métodos diferentes. La idea es optimizar el valor de los diferentes coeficientes de forma de alcanzar el orden más alto posible sin alterar demasiado la estabilidad. A pesar de que todos los métodos sean estables, la constante de estabilidad depende de los coeficientes del método. Tener coeficientes pequeños en valor absoluto mejora la estabilidad. El método más conocido en este sentido es el método de Runge-Kutta

clásico (RK4, con $q = 4$) que es un buen compromiso entre precisión y estabilidad, y que sigue siendo muy utilizado. Su tabla de Butcher es

0	0	0	0	0
1/2	1/2	0	0	0
1/2	0	1/2	0	0
1	0	0	1	0
	1/6	1/3	1/3	1/6

Es obtenido gracias a las fórmulas de integración numérica de punto medio, de rectángulos a la izquierda y, finalmente, de Simpson [17], [60]. Se puede comprobar que es de orden 4 [17], [36]. De hecho, se verifica fácilmente que es de orden 2 o más.

3.4. Órdenes y precisión en la práctica

Nos vamos a enfocar en el siguiente ejemplo, que viene del libro de A. Fortin [28] y que tiene una solución analítica:

$$y' = -y + t + 1, \quad y(0) = 1.$$

Tomemos, primero, un paso de tiempo $\tau = 0{,}1$ y veamos los resultados obtenidos con tres métodos de Runge-Kutta: Euler (RK1), de orden 1; Heun (RK2), de orden 2, y RK4, de orden 4. Presentamos en el cuadro 1 los valores obtenidos por los tres métodos. Observamos que en $t = 1$ encontramos la solución exacta con: 1 decimal correcto por Euler explícito; 2 decimales correctos por Heun, y 5 decimales correctos por RK4.

Ahora calculamos los errores $y(t_n) - y_n$. Con el cuadro 2 veamos que al último paso de tiempo tenemos, aproximadamente, un error de: $2{,}10^{-2}$ para Euler; $7{,}10^{-4}$ para Heun, y $3{,}10^{-7}$ para RK4.

Finalmente, dividimos por 2 el paso de tiempo ($\tau = 0,05$). El cuadro 3 y el cuadro 4 describen los resultados. Observamos ahora que en $t = 1$ encontramos la solución exacta con: 1 decimal correcto por Euler explícito; 2 decimales correctos por Heun, y 6 decimales correctos por RK4. Pero, sobre todo, veamos que al último paso de tiempo tenemos aproximadamente un error de:

- $1{,}10^{-2}$ para Euler, lo que corresponde a una reducción del error con un factor $1/2$ en comparación a la solución para $\tau = 0,1$;
- $1{,}6{,}10^{-4}$ para Heun, lo que corresponde a una reducción del error de $1/4$, aproximadamente;
- $2{,}10^{-8}$ para RK4, lo que corresponde a una reducción del error de $1/15$, aproximadamente.

Volvemos a encontrar los órdenes de convergencia teóricos: respectivamente, 1, 2 y 4.

t	Euler explícito	Heun	RK4	Solución analítica
0,0	1,000000	1,000000	1,000000	1,000000
0,1	1,000000	1,005000	1,004838	1,004837
0,2	1,010000	1,019025	1,018731	1,018731
0,3	1,029000	1,041218	1,040818	1,040818
0,4	1,056100	1,070802	1,070320	1,070320
0,5	1,090490	1,107076	1,106531	1,106531
0,6	1,131441	1,149404	1,148812	1,148812
0,7	1,178297	1,197210	1,196586	1,196585
0,8	1,230467	1,249975	1,249329	1,249329
0,9	1,287420	1,307228	1,306570	1,306570
1,0	1,348678	1,368541	1,367880	1,367879

Cuadro 1: Comparación de métodos numéricos con la solución analítica.

t	Euler explícito	Heun	RK4
0,0	0,000000	0,000000	0,000000e+00
0,1	-0,004837	0,000163	8,196404e-08
0,2	-0,008731	0,000294	1,483283e-07
0,3	-0,011818	0,000399	2,013195e-07
0,4	-0,014220	0,000482	2,428819e-07
0,5	-0,016041	0,000545	2,747107e-07
0,6	-0,017371	0,000592	2,982823e-07
0,7	-0,018288	0,000625	3,148798e-07
0,8	-0,018862	0,000646	3,256172e-07
0,9	-0,019149	0,000658	3,314595e-07
1,0	-0,019201	0,000662	3,332411e-07

Cuadro 2: Comparación de errores $y(t_n)$-y_n para diferentes métodos numéricos.

t	Euler explícito	Heun	RK4	Solución analítica
0,00	1,000000	1,000000	1,000000	1,000000
0,05	1,000000	1,001250	1,001229	1,001229
0,10	1,002500	1,004877	1,004837	1,004837
0,15	1,007375	1,010764	1,010708	1,010708
0,20	1,014506	1,018802	1,018731	1,018731
0,25	1,023781	1,028885	1,028801	1,028801
0,30	1,035092	1,040914	1,040818	1,040818
0,35	1,048337	1,054795	1,054688	1,054688
0,40	1,063420	1,070436	1,070320	1,070320
0,45	1,080249	1,087752	1,087628	1,087628
0,50	1,098737	1,106662	1,106531	1,106531
0,55	1,118800	1,127087	1,126950	1,126950
0,60	1,140360	1,148954	1,148812	1,148812
0,65	1,163342	1,172193	1,172046	1,172046
0,70	1,187675	1,196736	1,196585	1,196585
0,75	1,213291	1,222520	1,222367	1,222367
0,80	1,240127	1,249484	1,249329	1,249329
0,85	1,268120	1,277572	1,277415	1,277415
0,90	1,297214	1,306728	1,306570	1,306570
0,95	1,327354	1,336900	1,336741	1,336741
1,00	1,358486	1,368039	1,367879	1,367879

Cuadro 3: Comparación de métodos numéricos con la solución analítica, con $\tau = 0,05$.

t	Euler explícito	Heun	RK4
0,00	0,000000	0,000000	0,000000e+00
0,05	-0,001229	0,000021	2,582619e-09
0,10	-0,002337	0,000039	4,913327e-09
0,15	-0,003333	0,000056	7,010552e-09
0,20	-0,004225	0,000071	8,891524e-09
0,25	-0,005020	0,000084	1,057235e-08
0,30	-0,005726	0,000096	1,206808e-08
0,35	-0,006351	0,000107	1,339276e-08
0,40	-0,006900	0,000116	1,455953e-08
0,45	-0,007379	0,000124	1,558063e-08
0,50	-0,007794	0,000131	1,646751e-08
0,55	-0,008150	0,000137	1,723082e-08
0,60	-0,008452	0,000142	1,788050e-08
0,65	-0,008704	0,000147	1,842583e-08
0,70	-0,008910	0,000150	1,887544e-08
0,75	-0,009075	0,000153	1,923736e-08
0,80	-0,009202	0,000156	1,951909e-08
0,85	-0,009295	0,000157	1,972758e-08
0,90	-0,009355	0,000158	1,986930e-08
0,95	-0,009387	0,000159	1,995028e-08
1,00	-0,009394	0,000159	1,997610e-08

Cuadro 4: Comparación de errores $y(t_n)$-y_n para diferentes métodos numéricos, con $\tau = 0,05$.

3.5. Errores de redondeo

Las fórmulas

$$y_{n+1} = y_n + \tau\ F(t_n, y_n, \tau), \quad y_0 = y(0),$$

siguen siendo teóricas, en el sentido de que no toman en cuenta los errores de redondeo. En la práctica tenemos, de hecho:

$$\widetilde{y}_{n+1} = \widetilde{y}_n + \tau\,(F(t_n, \widetilde{y}_n, \tau) + \rho_n) + \sigma_n, \quad \widetilde{y}_0 = y(0) + \delta_0, \tag{3.31}$$

donde:

1. δ_0 representa el error de redondeo en la condición inicial;
2. ρ_n representa el error de redondeo en el cálculo de F;
3. σ_n representa el error de redondeo en el cálculo de $\widetilde{y}_{n+1}$.

Vamos a suponer que existen $\rho > 0$ y $\sigma > 0$ que verifican

$$|\rho_n| \leq \rho, \quad |\sigma_n| \leq \sigma,$$

para $0 \leq n < N$. El estándar desde los años 2000 para representar los números reales de forma aproximada es el *IEEE 754 binary floating-point arithmetic* [45]. La magnitud de los errores de redondeo en el *IEEE 754* es de 10^{-7} en precisión simple (*IEEE single precision*) y de 10^{-16} en precisión doble (*IEEE double precision*).

Supongamos nuestro método de un paso estable, en otros términos

$$\max_{n=0,\ldots,N} |\widetilde{y}_n - y_n| \leq C_S \left(|\widetilde{y}_0 - y_0| + \sum_{n=0}^{N-1} |\epsilon_n| \right),$$

con

$$\epsilon_n = \tau\rho_n + \sigma_n.$$

Entonces, tenemos

$$\sum_{n=0}^{N-1} |\epsilon_n| \leq \sum_{n=0}^{N-1} \left(\tau|\rho_n| + |\sigma_n|\right).$$

Luego acotamos de forma separada

$$\sum_{n=0}^{N-1} \tau|\rho_n| \leq \rho \sum_{n=0}^{N-1} \tau = T\rho$$

y también

$$\sum_{n=0}^{N-1} |\sigma_n| \leq N\sigma.$$

Esto lleva, finalmente, a

$$\max_{n=0,\ldots,N} |\widetilde{y}_n - y_n| \leq C_S \left(|\delta_0| + T\rho + N\sigma\right).$$

Combinamos lo anterior con el error de discretización del teorema 9 para un método de un paso estable de orden p:

$$\max_{n=0,\ldots,N} |y_n - y(t_n)| \leq CT\tau^p.$$

Obtenemos, mediante desigualdad triangular:

$$\max_{n=0,\ldots,N} |\widetilde{y}_n - y(t_n)| \leq CT\tau^p + C_S \left(|\delta_0| + T\rho + N\sigma\right).$$

Utilizamos la relación $T = N\tau$ y obtenemos:

$$\max_{n=0,\ldots,N} |\widetilde{y}_n - y(t_n)| \leq CT\tau^p + C_S \left(|\delta_0| + T\rho + \frac{T}{\tau}\sigma\right).$$

Deducimos el efecto siguiente: si τ es demasiado pequeño, el error total aumenta, dado que el término de error de redondeo $\frac{T}{\tau}\sigma$ termina por ser predominante. Lo ilustramos con el programa que sigue (en precisión simple).

```
import numpy as np

# Definimos la función para el método de Euler
def euler_method(f, y0, t0, tf, h):
    t_values = np.arange(t0, tf + h, h, dtype=np.float32)
    y_values = np.zeros(len(t_values), dtype=np.float32)
    y_values[0] = np.float32(y0)

```

```
    for i in range(1, len(t_values)):
        y_values[i] = y_values[i-1] + h * f(t_values[i-1],
            y_values[i-1])

    return t_values, y_values

# Definimos la función para la ecuación diferencial
def f(t, y):
    return np.float32(-y)

# Condiciones iniciales
y0 = np.float32(1)
t0 = np.float32(0)
tf = np.float32(1)

# Solución exacta
def exact_solution(t):
    return np.exp(np.float32(-t))

# Tamaños de paso a evaluar
step_sizes = np.logspace(-1, -7, 10, dtype=np.float32)

# Iteramos sobre los diferentes tamaños de paso
for h in step_sizes:
    t_values, y_values = euler_method(f, y0, t0, tf, h)
    # Calculamos el error en t = 1
    error = abs(y_values[-1] - exact_solution(tf))
    print("Tamaño del paso (h): {h:.10f}, Error en t = 1:
       {error:.10f}")
```

Listado 3: Ilustración del efecto del error de redondeo.

Obtenemos el resultado presentado en el cuadro 5. Se observa claramente el efecto del redondeo: a partir de un paso de tiempo $0,00001$ el error vuelve a crecer.

Tamaño del paso τ	**Error en $t = 1$**
0,1000000015	0,0192009807
0,0215443466	0,0085987151
0,0046415888	0,0018043816
0,0010000000	0,0001839995
0,0002154435	0,0000718236
0,0000464159	0,0000199676
0,0000100000	0,0000004470
0,0000021544	0,0000506639
0,0000004642	0,0016169846
0,0000001000	0,0311321318

Cuadro 5: Efecto del error de redondeo: error $\widetilde{y}_n - y(t_n)$ por varios valores del paso de tiempo τ.

3.6. Control del paso

Podemos hacer variar fácilmente el paso de tiempo a cada iterada en un método de un paso explícito. Cabe hacer lo siguiente: sea $t_0 < t_1 < t_2 < \ldots < t_N$ una subdivisión arbitraria del intervalo $[0, T]$. Definimos:

$$\tau_n = t_{n+1} - t_n, \tag{3.32}$$

para todo $0 \leq n < N$. Luego cambiamos (3.3) como sigue:

$$y_{n+1} = y_n + \tau_n F(t_n, y_n, \tau_n), \tag{3.33}$$

donde simplemente τ fue sustituido por τ_n. Se puede definir τ_n *a priori*, por ejemplo, si estamos interesados en tener más precisión en un intervalo de tiempo específico. También se puede pensar en ajustar τ_n de forma automática. Vamos a ver una forma de hacerlo.

3.6.1. Indicador *a posteriori* del error

En el caso de Euler explícito, no es difícil tener un indicador local η_n del error de discretización siguiendo [17]. La idea sería utilizarlo para disminuir el paso de tiempo τ_n en las regiones donde η_n es demasiado grande. Nos acordamos de que para controlar el error de discretización global es suficiente controlar $\sum_{0 \leq n < N} |e_n|$, que es la suma de los errores de consistencia. Para Euler explícito, hemos visto que el error de consistencia local es

$$e_n = \frac{1}{2}\tau_n^2 f^{[1]}(t_n, y(t_n)) + \mathcal{O}(\tau_n^3),$$

donde $y(t_n)$ es la solución exacta. Vamos a determinar una estimación η_n de e_n. En efecto, con Taylor tenemos, por otro lado,

$$\begin{aligned} p_{n+1} - p_n &= f(t_{n+1}, y_{n+1}) - f(t_n, y_n) \\ &= \tau_n f^{[1]}(t_n, y_n) + \mathcal{O}(\tau_n^2), \end{aligned}$$

donde y_n es la solución aproximada por el método de Euler. Entonces, definimos

$$\eta_n := \frac{1}{2}\tau_n(p_{n+1} - p_n).$$

Mencionemos que el indicador *a posteriori* η_n se calcula únicamente a partir de la solución aproximada y_n, y el cálculo es fácil, dado que solo involucra la evaluación de f. De esta forma, tenemos $\eta_n \simeq e_n$. Más precisamente:

$$\begin{aligned}\eta_n &= \frac{1}{2}\tau_n^2 f^{[1]}(t_n, y_n) + \mathcal{O}(\tau_n^3) \\ &= \frac{1}{2}\tau_n^2 f^{[1]}(t_n, y(t_n) + \delta y) + \mathcal{O}(\tau_n^3),\end{aligned}$$

donde $\delta y := y_n - y(t_n)$. Ahora utilizamos la cuota *a priori* (teorema 9) que nos entrega

$$|\delta y| \leq CT\tau_n,$$

donde CT es independiente de τ_n. Utilizando Taylor:

$$f^{[1]}(t_n, y(t_n) + \delta y) = f^{[1]}(t_n, y(t_n)) + f_y^{[1]'}(t_n, y(t_n))\delta y + \mathcal{O}(\delta y^2).$$

Finalmente, con lo anterior y la desigualdad triangular, viene

$$\begin{aligned}|\eta_n - e_n| &\leq \frac{1}{2}\tau_n^2 \left| f_y^{[1]'}(t_n, y(t_n)) \right| |\delta y| + \frac{1}{2}\tau_n^2 \left| \mathcal{O}(\delta y^2) \right| + \left| \mathcal{O}(\tau_n^3) \right| \\ &\leq \frac{1}{2}\tau_n^2 (C'CT\tau_n) + \frac{1}{2}\tau_n^2 \left| \mathcal{O}((CT\tau_n)^2) \right| + \left| \mathcal{O}(\tau_n^3) \right|,\end{aligned}$$

con

$$C' := \sup_{0 \leq t \leq T} \left| f_y^{[1]'}(t_n, y(t_n)) \right|.$$

Asumimos esta constante finita, simplificamos y obtenemos

$$|e_n| \leq |\eta_n| + \mathcal{O}(\tau_n^3).$$

Esto se puede considerar como un resultado de fiabilidad: si no se toman en cuenta los términos de alto orden en τ_n, el estimador domina el error de consistencia. Cabe mencionar que no es fácil generalizar para métodos de más alto orden [17]. Para estimadores más generales y con mejores propiedades ver, por ejemplo, el artículo de D. Estep [23], las publicaciones relacionadas [18], [19], [24], además de los trabajos recientes de B. Kehlet y A. Logg [47] y de M. Feischl y D. Niederkofler [27].

3.6.2. Aplicación a Van der Pol

Tomamos el mismo ejemplo del capítulo de M. Feischl y D. Niederkofler [27]: el oscilador de Van der Pol[5]. Hallamos $x, y : [0, T] \to \mathbb{R}$ $(T = 10)$ soluciones de

$$x' = f_1(x, y), \quad y' = f_2(x, y), \tag{3.34}$$

con condiciones iniciales

$$x(0) = y(0) = 1$$

y las siguientes expresiones de f_1 y f_2:

$$f_1(x, y) = y, \quad f_2(x, y) = 5(1 - x^2)y - x.$$

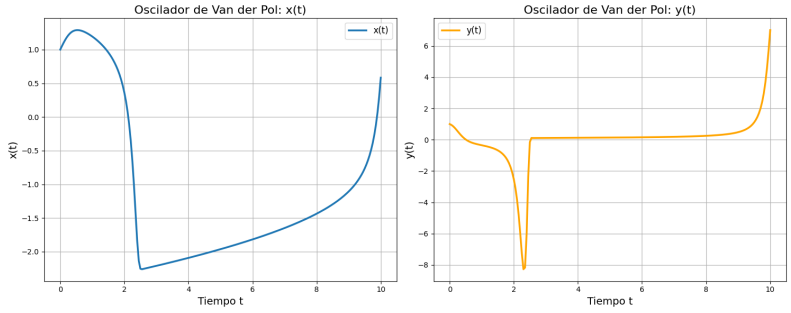
Es un sistema autónomo de dos ecuaciones no lineales de primer orden.

Aquí no seguimos los métodos tradicionales que adaptan directamente y dinámicamente el paso de tiempo τ_n, tal como están presentados en [17], [37], por ejemplo. De hecho, estos métodos carecen, por lo general, de robusteza y de respaldo teórico fuerte (ver la discusión en la introducción de [27]). Preferimos seguir la tecnología presentada en [27], que sigue métodos adaptativos clásicos para ecuaciones en derivadas parciales (ver, entre otros, [5], [8]). Esta tecnología sigue la idea siguiente:

SOLVE → ESTIMATE → MARK → REFINE

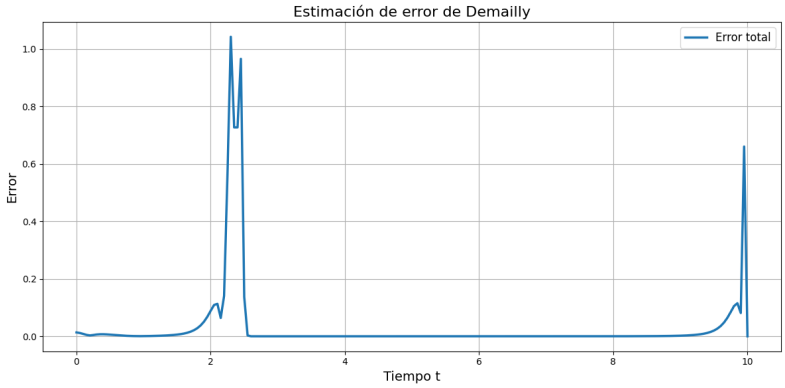
Explicamos e ilustramos a continuación los 4 pasos.

Etapa 1: SOLVE Solucionamos (3.34) con Euler explícito. En la etapa inicial, tomamos un paso de tiempo τ uniforme lo más grande posible. Obtenemos lo siguiente:



[5]Ver también, por ejemplo, [35].

Etapa 2: ESTIMATE Calculamos η_n para todo n. Como el sistema de Van der Pol (3.34) tiene dos componentes, hacemos un cálculo por componente y tomamos la norma euclidiana de los dos. Obtenemos:

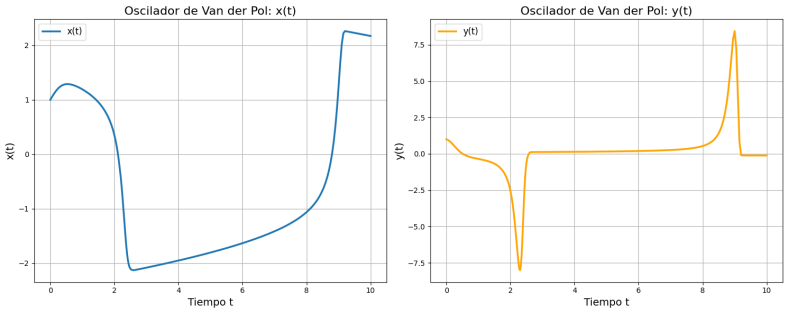


Hemos representado la norma global del estimador *a posteriori* η_n. Notamos que el estimador indica errores de importante magnitud un poco después de $t = 2$ y un poco antes de $t = 10$.

Etapa 3: MARK Identificamos los intervalos discretos $[t_n; t_{n+1}]$, donde $|\eta_n|$ tiene los valores más altos y lo marcamos para refinar. Los intervalos marcados se localizan un poco después de $t = 2$ y un poco antes de $t = 10$, de manera conforme al comportamiento del estimador.

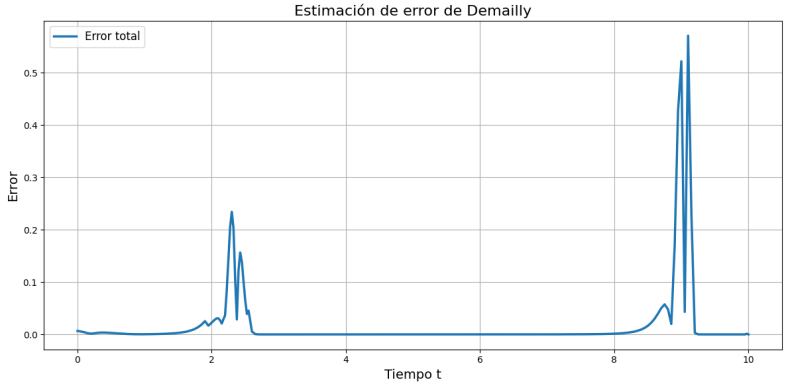
Etapa 4: REFINE Dividimos en dos únicamente los intervalos discretos marcados en la etapa anterior, donde $|\eta_n|$ es de magnitud importante.

Iteración siguiente: SOLVE (2) Hemos obtenido, al final, una nueva subdivisión $t_0 < t_1 < \ldots < t_{N_1}$ del intervalo de tiempo, con $N_1 > N$. La subdivisión es más fina y refinada donde se necesita más precisión. Calculamos de nuevo una solución con Euler explícito y esta subdivisión más fina. Obtenemos:

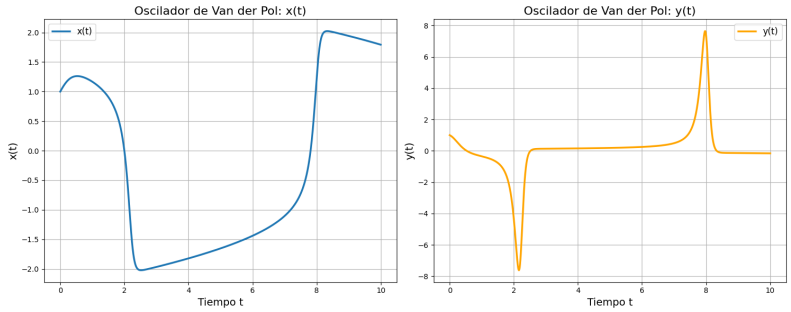


Observamos que la solución cambió en comparación a la iteración anterior, sobre todo, entre $t = 8$ y $t = 10$.

Iteración siguiente: ESTIMATE (2) Luego se puede volver a calcular el estimador η_n. Ahora tenemos la siguiente distribución de error.



Se observa que refinar donde el estimador fue más grande, cerca de $t = 2$, fue efectivo y permitió reducir el error en esta región. No obstante, aparece un nuevo error cerca de $t = 8$. Veamos ahora una solución de referencia calculada con un paso de tiempo muy pequeño ($\tau = 0{,}0005$).



Ahora tenemos la explicación: la predicción a la primera iteración era muy gruesa y no hacía aparecer la oscilación cerca de $t = 8$. La primera iteración adaptativa lo corrigió, pero no completamente. Con unas iteraciones más, llegamos muy cerca de la solución de referencia.

3.7. Referencias complementarias

Si se quiere saber más sobre métodos de un paso, aconsejamos, particularmente, [16], [17], [37], [46], [60] y también [4], [11], [28], [44]. Unos clásicos sobre métodos de Runge-Kutta son [6], [7].

Capítulo 4
Métodos explícitos multipasos

Seguimos con el problema de Cauchy

$$y' = f(t, y)$$

e $y(0) = y_0$ por $0 \leq t \leq T$. Sea τ el paso de tiempo y seguimos con las mismas notaciones relativas a lo anterior. Un método multipasos a $r+1$ pasos es un método de la forma

$$y_{n+1} = \Psi(t_n, y_n; \ldots; t_{n-r}, y_{n-r}).$$

Se puede, de esta forma, obtener un orden de convergencia elevado sin tener tantos cálculos como en los métodos de Runge-Kutta, aprovechando los cálculos de los pasos anteriores para mejorar la precisión. Esta estrategia tiene una contraparte: se pierde algo de estabilidad.

4.1. Una clase general de métodos

Notamos $f_n = f(t_n, y_n)$ e introducimos

$$y_{n+1} = \sum_{k=0}^{r} \alpha_k y_{n-k} + \tau \sum_{k=0}^{r} \beta_k f_{n-k}, \tag{4.1}$$

donde vamos a especificar los valores de los coeficientes reales $(\alpha_k)_{0 \leq k \leq r}$ y $(\beta_k)_{0 \leq k \leq r}$. Se inicia el algoritmo de la siguiente forma: se calculan $(y_1, f_1), \ldots, (y_r, f_r)$, por ejemplo, gracias a un método de un paso como Runge-Kutta, eligiendo con cuidado su orden.

4.1.1. Ejemplo: el método de Nyström

El método de Nyström se define como

$$y_{n+1} = y_{n-1} + 2\tau f_n.$$

Es parecido al método del punto medio explícito visto en el capítulo anterior, pero utiliza dos pasos de tiempo en vez de uno.

4.1.2. Error de consistencia

La definición del error de consistencia vista en el capítulo anterior se extiende naturalmente a métodos multipasos.

Definición 7. *El error de consistencia asociada al método* (4.1) *se define como*

$$e_n := y(t_{n+1}) - \left(\sum_{k=0}^{r} \alpha_k y(t_{n-k}) + \tau \sum_{k=0}^{r} \beta_k f(t_{n-k}, y(t_{n-k})) \right),$$

donde y es la solución exacta del problema de Cauchy (3.1).

De nuevo, si el error de consistencia es de orden $p+1$, el método multipasos es de orden p.

Ejemplo 12. *Veamos el error de consistencia para el método de Nyström. Tenemos:*

$$\begin{aligned}
e_n &= y(t_{n+1}) - \left(\sum_{k=0}^{r} \alpha_k y(t_{n-k}) + \tau \sum_{k=0}^{r} \beta_k f(t_{n-k}, y(t_{n-k})) \right) \\
&= y(t_{n+1}) - (y(t_{n-1}) + 2\tau f(t_n, y(t_n))) \\
&= y(t_{n+1}) - (y(t_{n-1}) + 2\tau y'(t_n)) \\
&= y(t_n) + \tau y'(t_n) + \frac{1}{2}\tau^2 y''(t_n) + \frac{1}{6}\tau^3 y^{(3)}(t_n) + o(\tau^3) \\
&\quad -(y(t_n) - \tau y'(t_n) + \frac{1}{2}\tau^2 y''(t_n) - \frac{1}{6}\tau^3 y^{(3)}(t_n) + o(\tau^3) + 2\tau y'(t_n)) \\
&= \frac{1}{3}\tau^3 y^{(3)}(t_n) + o(\tau^3).
\end{aligned}$$

Obtenemos un método de orden 2.

4.1.3. Métodos de Adams-Bashforth

Veamos ahora una clase muy conocida de métodos multipasos explícitos [17], [60]. Sea y la solución exacta del problema de Cauchy (3.3).

Podemos escribir

$$y(t_{n+1}) = y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\mathrm{d}t.$$

Tomamos $r \geq 0$. Conocemos los valores siguientes:

$$y(t_{n-k}), \quad f_{n-k} := f(t_{n-k}, y(t_{n-k})),$$

para $0 \leq k \leq r$. El método se basa en la idea siguiente: calcular el polinomio de interpolación de Lagrange p que aproxima $t \mapsto f(t, y(t))$ en los puntos $(t_n, y_n), \ldots, (t_{n-r}, y_{n-r})$ [17], [60]. Este polinomio se escribe

$$p(t) = \sum_{k=0}^{r} f_{n-k}\mathcal{L}_k(t),$$

con $(\mathcal{L}_k)_{k=0,\ldots,r}$ los polinomios de la base de Lagrange asociada, de grado r y definidos mediante:

$$\mathcal{L}_k(t_{n-i}) = \begin{cases} 1 & i = k, \\ 0 & i \neq k. \end{cases}$$

El polinomio p es un polinomio de grado r. Ahora escribimos

$$\begin{aligned} y(t_{n+1}) &= y(t_n) + \int_{t_n}^{t_{n+1}} f(t, y(t))\,\mathrm{d}t \\ &\simeq y(t_n) + \int_{t_n}^{t_{n+1}} p(t)\,\mathrm{d}t \\ &= y(t_n) + \sum_{k=0}^{r} f_{n-k} \int_{t_n}^{t_{n+1}} \mathcal{L}_k(t)\,\mathrm{d}t. \end{aligned}$$

Hemos utilizado la propiedad $f \simeq p$, y luego la descomposición de p en la base de Lagrange. Inspirados por la fórmula anterior, los métodos de Adams-Bashforth se escriben como

$$y_{n+1} = y_n + \tau \sum_{k=0}^{r} \beta_{n-k} f_{n-k} \tag{4.2}$$

con

$$\beta_{n-k} = \frac{1}{\tau}\int_{t_n}^{t_{n+1}} \mathcal{L}_k(t)\mathrm{d}t. \tag{4.3}$$

El éxito de estos métodos viene de que son sencillos y que permiten alcanzar un orden alto sin evaluaciones extras de la función f. Veamos ahora algunos casos particulares.

- Cuando $r = 0$, tenemos $\mathcal{L}_0 = 1$, y $\beta_n = 1$. La ecuación (4.2) se reduce a
$$y_{n+1} = y_n + \tau f_n \tag{4.4}$$
y volvemos a encontrar Euler explícito.

- Cuando $r = 1$, un simple cálculo permite obtener
$$y_{n+1} = y_n + \tau\left(\frac{3}{2}f_n - \frac{1}{2}f_{n-1}\right). \tag{4.5}$$

- Cuando $r = 3$, encontramos el método clásico de Adams-Bashforth que se escribe
$$y_{n+1} = y_n + \frac{\tau}{24}\left(55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}\right).$$
Ver, por ejemplo, [60]. Es un método de orden 4 como lo vamos a ver después.

- Se puede también escribir el método para cualquier orden $r \geq 4$. No obstante, los coeficientes β_{n-k} crecen con r, lo que disminuye la estabilidad del método [17].

4.1.4. Implementación y un ejemplo

Veamos cómo se implementa el método de Adams-Bashforth en Python. Notar la etapa de inicialización con Runge-Kutta. Lo ilustramos con el sistema de Lorenz:

$$\begin{cases} x' = \sigma(y - x), \\ y' = x(\rho - z) - y, \\ z' = xy - \beta z, \end{cases}$$

donde σ, ρ y β son parámetros del sistema con condiciones iniciales $x(0) = y(0) = z(0) = 1$. Tenemos $T = 50$ y $\tau = 0{,}005$. El gráfico resultante está en la figura 3.

```
import numpy as np
import matplotlib.pyplot as plt

# Definimos el sistema de Lorenz
def lorenz(t, y, sigma=10, beta=8/3, rho=28):
    dy = np.zeros_like(y)
    dy[0] = sigma * (y[1] - y[0])
    dy[1] = y[0] * (rho - y[2]) - y[1]
    dy[2] = y[0] * y[1] - beta * y[2]
    return dy

# Implementación del método de Adams-Bashforth de 4 pasos
def adams_bashforth(f, t_span, y0, steps):
    t0, tf = t_span
    h = (tf - t0) / steps
    t = np.linspace(t0, tf, steps + 1)
    y = np.zeros((steps + 1, len(y0)))
    y[0] = y0

    # Usamos RK4 para los primeros 4 pasos
    for i in range(3):
        k1 = f(t[i], y[i])
        k2 = f(t[i] + h/2, y[i] + h/2 * k1)
        k3 = f(t[i] + h/2, y[i] + h/2 * k2)
        k4 = f(t[i] + h, y[i] + h * k3)
        y[i+1] = y[i] + h/6 * (k1 + 2*k2 + 2*k3 + k4)

    # Aplicamos Adams-Bashforth para los pasos restantes
    for i in range(3, steps):
        y[i+1] = y[i] + h/24 * (55*f(t[i], y[i]) - 59*f(t[
            i-1], y[i-1]) + 37*f(t[i-2], y[i-2]) - 9*f(t[i
            -3], y[i-3]))

    return t, y

# Cond. iniciales
y0 = np.array([1.0, 1.0, 1.0])
t_span = (0, 50)
steps = 10000

```

```
# Resolvemos el sistema
t, y = adams_bashforth(lorenz, t_span, y0, steps)

# Graficamos el plano de fase
fig = plt.figure(figsize=(10, 8))
ax = fig.add_subplot(111, projection='3d')
ax.plot(y[:, 0], y[:, 1], y[:, 2], lw=0.5)
ax.set_xlabel("X")
ax.set_ylabel("Y")
ax.set_zlabel("Z")
ax.set_title("Plano de fase del sistema de Lorenz")
plt.show()
```

Listado 4: Implementación de Adams-Bashforth en Python.

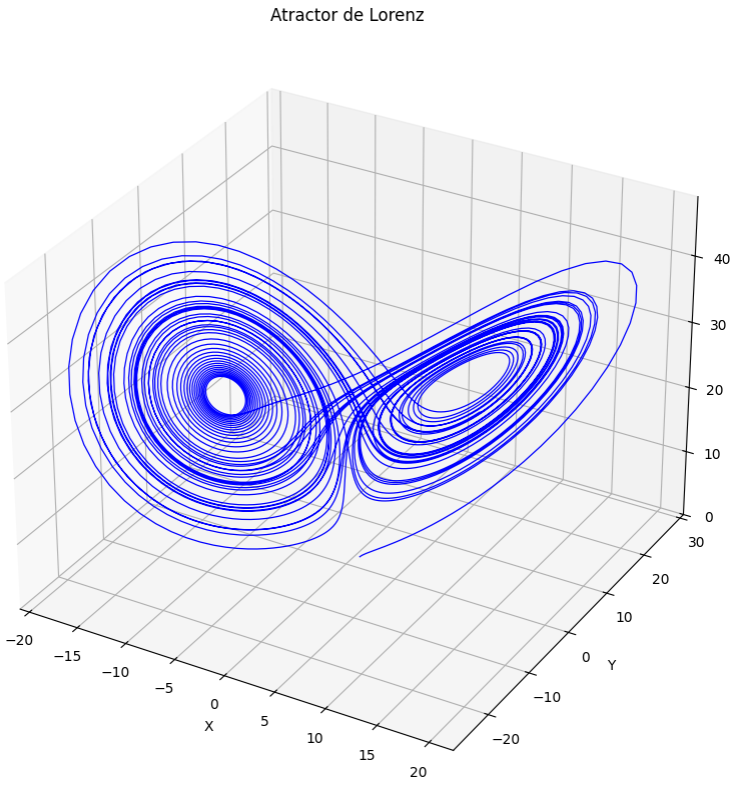


Figura 3: Sistema de Lorenz resuelto con Adams-Bashforth (dibujo en el plano de fase).

4.1.5. Consistencia de Adams-Bashforth

Necesitamos el resultado clásico siguiente sobre la interpolación de Lagrange [60]. Suponemos que f es una función real definida y continua en un intervalo $[a, b]$, que admite derivadas continuas en $[a, b]$ hasta el orden $r+1$. Tomamos $r+1$ puntos de interpolación distintos $x_0, x_1, \ldots, x_r$ y denotamos p el polinomio de interpolación asociado ($p(x_k) = f(x_k)$, $k = 0, \ldots, r$). Introducimos el polinomio de grado $r + 1$ siguiente:

$$\pi_{r+1}(x) = (x - x_0)(x - x_1) \ldots (x - x_r).$$

Se puede establecer [60]:

$$|f(x) - p(x)| \leq \frac{M_{r+1}}{(r+1)!}|\pi_{r+1}(x)|,$$

donde

$$M_{r+1} = \max_{a \leq \xi \leq b} |f^{(r+1)}(\xi)|.$$

Ahora volvemos al error de consistencia, que es dado por

$$e_n = y(t_{n+1}) - y(t_n) - \int_{t_n}^{t_{n+1}} p(t)\mathrm{d}t = \int_{t_n}^{t_{n+1}} (y'(t) - p(t))\mathrm{d}t,$$

donde p es el polinomio de interpolación de $t \mapsto f(t, y(t)) = y'(t)$ introducido anteriormente. Utilizamos el teorema del valor medio: existe $t_n \leq \theta \leq t_{n+1}$ que cumple

$$e_n = (t_{n+1} - t_n)(y'(\theta) - p(\theta)).$$

Utilizamos ahora la cuota superior:

$$|y'(\theta) - p(\theta)| \leq \frac{M_{r+1}}{(r+1)!}|\pi_{r+1}(\theta)|.$$

Para $0 \leq k \leq r$ utilizamos $t_n \leq \theta \leq t_{n+1}$ y acotamos

$$|\theta - t_{n-k}| \leq (k+1)\tau \leq (r+1)\tau.$$

Obtenemos:

$$|\pi_{r+1}(\theta)| = |\theta - t_n| \ldots |\theta - t_{n-r}| \leq ((r+1)\tau)^{r+1}.$$

Finalmente, deducimos:

$$|e_n| \leq \tau \frac{M_{r+1}}{(r+1)!}((r+1)\tau)^{r+1} = \frac{M_{r+1}(r+1)^{r+1}}{(r+1)!}\tau^{r+2}.$$

Entonces, el método de Adams-Bashforth de $r+1$ pasos es de orden $r+1$.

4.2. Cero-estabilidad

Estudiamos la estabilidad de los métodos multipasos; es un asunto más delicado en comparación a los métodos de un paso.

4.2.1. Una definición

Primero extendemos la noción de estabilidad vista en el capítulo anterior.

Definición 8. *Un método multipasos es estable si verifica lo siguiente. Sea $\tau_M > 0$. Existe una constante de estabilidad $C_s > 0$ tal que, para todo valor de $\tau \leq \tau_M$ (o equivalentemente $N \geq T/\tau_M$), y para toda pareja de soluciones $(y_n)_{0\leq n\leq N}$ y $(\widetilde{y}_n)_{0\leq n\leq N}$ definidas de la siguiente forma:*

$$\begin{aligned} y_{n+1} &= \Psi(t_n, y_n; \ldots; t_{n-r}, y_{n-r}), \\ \widetilde{y}_{n+1} &= \Psi(t_n, \widetilde{y}_n; \ldots; t_{n-r}, \widetilde{y}_{n-r}) + \epsilon_n, \end{aligned}$$

para todo $r \leq n < N$, con $(\epsilon_n)_{0\leq n<N}$ una secuencia arbitraria de reales, se cumple la desigualdad:

$$\max_{0\leq n\leq N} |y_n - \widetilde{y}_n| \leq C_s \left(\max_{0\leq n\leq r} |y_n - \widetilde{y}_n| + \sum_{r\leq n<N} |\epsilon_n| \right). \tag{4.6}$$

4.2.2. Un lema importante

El análisis de estabilidad se basa en el resultado siguiente. Es enunciado en el libro de E. Süli y D. Mayers y demostrado en el libro de P. Henrici [40].

Lema 5 (Lema 12.1 de Süli & Mayers)**.** *Sea la relación de recurrencia lineal homogénea de orden r siguiente:*

$$\alpha_r y_{n+r} + \cdots + \alpha_1 y_{n+1} + \alpha_0 y_n = 0, \qquad n \in \mathbb{N}, \tag{4.7}$$

donde $\alpha_k \in \mathbb{R}$, $k = 0, \ldots, r$, $\alpha_r \neq 0$, $\alpha_0 \neq 0$. *Sea el polinomio característico correspondiente,*

$$p(z) = \alpha_r z^r + \cdots + \alpha_1 z + \alpha_0.$$

Sean

$$(z_1, m_1) \ldots (z_l, m_l)$$

las raíces distintas del polinomio p juntas con sus multiplicidades, con $l \leq r$ *y* $m_1 + \ldots + m_l = r$. *Entonces, toda secuencia* $(y_n)_{n\geq 0}$ *de complejos que verifica* (4.7) *se escribe de la forma*

$$y_n = \sum_{j=1}^{l} p_j(n) z_j^n, \tag{4.8}$$

donde $p_j(\cdot)$ *es un polinomio de orden* $m_j - 1$, $1 \leq j \leq l$.

4.2.3. Condición necesaria de estabilidad

Para tener una condición necesaria de estabilidad, consideramos la ecuación

$$y' = 0.$$

En este caso, el método de un paso (4.1) se escribe

$$y_{n+1} = \sum_{k=0}^{r} \alpha_k y_{n-k}, \tag{4.9}$$

para $n \geq r$. Sea el polinomio característico

$$\lambda^{r+1} - \alpha_0 \lambda^r - \alpha_1 \lambda^{r-1} - \ldots - \alpha_r = 0.$$

Utilizamos el lema 5. Vamos a notar $\lambda_j \in \mathbb{C}$ las raíces de este polinomio característico y m_j las multiplicidades correspondientes. Fijamos j y consideramos λ_j una de estas raíces. Ahora tomamos $\epsilon > 0$ y consideramos las secuencias y_n y $\widetilde{y}_n$ generadas por las iteraciones

$$y_{n+1} = \sum_{k=0}^{r} \alpha_k y_{n-k}, \quad \widetilde{y}_{n+1} = \sum_{k=0}^{r} \alpha_k \widetilde{y}_{n-k},$$

con $n \geq r$. Las fórmulas anteriores son idénticas. Entonces, para que las secuencias puedan ser distintas, las iniciamos de forma distinta, por $n = 0, \ldots, r$:

$$y_n = 0, \quad \widetilde{y}_n = \epsilon \lambda_j^n.$$

Dado que λ_j es solución del polinomio característico, tenemos, para todo $n \geq 0$:

$$y_n = 0, \quad \widetilde{y}_n = \epsilon \lambda_j^n.$$

Supongamos que el método multipasos sea estable. La fórmula (4.6) implica

$$|y_N - \widetilde{y}_N| = \epsilon |\lambda_j|^N \leq C_s \max_{0 \leq n \leq r} \epsilon |\lambda_j|^n.$$

Simplificamos:

$$|\lambda_j|^N \leq C_s \max_{0 \leq n \leq r} |\lambda_j|^n = C_s \max\{1, |\lambda_j|^r\}.$$

Finalmente, tomamos el límite cuando $\tau \to 0$, o con $N \to +\infty$. Obtenemos la condición $|\lambda_j| \leq 1$.

Ahora consideramos la situación donde λ_j es de módulo 1 ($|\lambda_j| = 1$) y de multiplicidad $m_j \geq 2$. En este caso, λ_j, como raíz múltiple, es también raíz del polinomio derivado. Tomamos $\widetilde{y}_n = \epsilon n \lambda_j^n$ y encontramos

$$|y_N - \widetilde{y}_N| = \epsilon N |\lambda_j|^N = \epsilon N$$

y, por otro lado,

$$\max_{0 \leq n \leq r} |y_n - \widetilde{y}_n| = \epsilon r,$$

lo que impide tener la condición (4.6): el método multipasos no puede ser estable. Hemos demostrado el resultado siguiente.

Teorema 11. *Si el método multipasos* (4.1) *es estable, entonces, todas las raíces* λ *de su polinomio característico tienen que cumplir* $|\lambda| \leq 1$. *Además, todas las raíces de módulo 1 tienen que ser simples.*

Observación 14. *Se puede establecer que la condición necesaria anterior es también suficente [17].*

4.2.4. Unos ejemplos

Veamos ahora algunos ejemplos.

Método de Nyström El polinomio característico es

$$\lambda^2 - 1 = 0,$$

que admite dos raíces simples: -1 y 1. Entonces, el método es estable. No obstante, dado que la otra raíz es -1, la estabilidad no es muy buena [17].

Método de Adams–Bashforth El polinomio característico es

$$\lambda^{r-1}(\lambda - 1) = 0.$$

El polinomio tiene una raíz múltiple, 0, y otra raíz simple, 1, entonces, el métodos es estable.

Se puede demostrar que la constante de estabilidad, que depende de los coeficientes del método, aumenta cuando r crece [17]. Entonces, se pierde estabilidad para métodos de orden alto. Por esta misma razón, no se utilizan mucho métodos de orden más grande que 4.

4.3. Convergencia

Como para los métodos de un paso la estabilidad implica el resultado siguiente:

$$\max_{0\leq n\leq N} |y_n - y(t_n)| \leq C_s \left(\max_{0\leq n\leq r} |y_n - y(t_n)| + \sum_{r\leq n<N} |e_n| \right),$$

donde y es la solución exacta del problema de Cauchy (3.1), y_n es la solución aproximada por el método multipasos (4.1) y e_n es el error de consistencia del método. Esta vez tenemos que acotar todos los pasos de inicialización del método. Dado esto, veamos que si el método es estable y consistente, aún tenemos convergencia.

Veamos ahora lo del orden de convergencia. Supongamos que la inicialización se hace con un método de orden $q \geq 1$:

$$\max_{0 \leq n \leq r} |y_n - y(t_n)| \leq C\tau^{q+1},$$

y que el método multipasos sea de orden p:

$$\sum_{r \leq n < N} |e_n| \leq C \sum_{r \leq n < N} \tau^{p+1} \leq CT\tau^p.$$

Obtenemos:

$$\max_{0 \leq n \leq N} |y_n - y(t_n)| \leq C_s C \left(\tau^{q+1} + T\tau^p\right).$$

Particularmente, si $q \geq p - 1$, tenemos

$$\max_{0 \leq n \leq N} |y_n - y(t_n)| \leq C\tau^p.$$

Podemos, entonces, hacer la inicialización con un método de orden $p-1$ sin perder la convergencia en orden p.

Capítulo 5
Métodos implícitos

Hasta ahora hemos visto métodos explícitos, que son fáciles de implementar y que funcionan muy bien para una clase importante de ecuaciones diferenciales. Sin embargo, existe otra clase de ecuaciones diferenciales, bastante comunes, para las cuales los métodos explícitos no dan buenos resultados en la práctica. Vamos a tratar de entender por qué y ver cómo los métodos implícitos de integración permiten solucionar este tema.

5.1. Sistemas rígidos

Hemos visto en el capítulo introductorio (capítulo 1) un ejemplo de problema de Cauchy que no se resuelve bien con el método de Euler explícito. Los problemas rígidos (*stiff*, en inglés) son todos los problemas por los cuales los métodos explícitos no dan buenos resultados en la práctica. Veamos otro ejemplo propuesto en el libro de J. P. Demailly [17] (varios otros ejemplos muy ilustrativos se pueden encontrar en otros libros, particularmente en [39], [46]). Sea el problema de Cauchy

$$y' = -150y + 30, \quad y(0) = 1/5.$$

La solución exacta es una constante:

$$y(t) = 1/5.$$

Si tomamos como dato inicial

$$\tilde{y}(0) = 1/5 + \epsilon,$$

con *epsilon* cualquier real, tenemos

$$\tilde{y}(t) = 1/5 + \epsilon e^{-150t}$$

El problema está matemáticamente bien puesto: hay existencia y unicidad de solución (Cauchy-Lipschitz aplica sin problema). Está también numéricamente bien puesto: la sensibilidad a la perturbación es compensada por el término exponencial decreciente.

Tratamos de aplicar Euler explícito con un paso de tiempo τ. Una recurrencia directa nos da

$$y_n - 1/5 = (1 - 150\tau)^n (y_0 - 1/5).$$

Para darnos una idea, supongamos $\tau = 0{,}02$. Si tenemos un error inicial $y_0 = 1/5 + \epsilon$, esto nos da

$$y_n = 1/5 + (-2)^n \epsilon,$$

y al tiempo $t = 1$

$$y_{50} \simeq 1/50 + 10^{15}\epsilon.$$

La solución tiene oscilaciones crecientes y no se parece en nada a la solución exacta. Para tener estabilidad, y volver a tener una solución monótona y decreciente, necesitamos $\tau < 1/75$. Esto es demasiado restrictivo.

Otros ejemplos vienen de la discretización de ciertas ecuaciones parciales en tiempo y espacio (parabólicas), como la ecuación del calor o la de Stokes.

5.2. A-estabilidad

Veamos ahora una nueva noción de estabilidad, más fuerte que la anterior. Esta noción permite entender mejor algunas propiedades de los métodos de integración en tiempo, particularmente en tiempo largo y para sistemas rígidos. Seguimos la presentación del libro de E. Süli y D. F. Mayers [60].

Consideremos el siguiente método multipasos:

$$\sum_{k=0}^{r} \alpha_k y_{n+k} = \tau \sum_{k=0}^{r} \beta_k f_{n+k}. \tag{5.1}$$

Observamos que la fórmula es más general que la del capítulo anterior. Notamos también que, por un tema de comodidad, hemos cambiado el rol de los índices. Aplicamos el método (5.1) para aproximar la ecuación lineal $y' = \lambda y$, $\lambda \in \mathbb{C}$, y obtenemos la siguiente relación de recurrencia:

$$\sum_{k=0}^{r} (\alpha_k - \tau\lambda\beta_k) y_{n+k} = 0.$$

El polinomio característico de esta relación es:

$$\pi(z; \tau\lambda) = \sum_{k=0}^{r} (\alpha_k - \tau\lambda\beta_k) z^k.$$

Se llama *polinomio de estabilidad*. Es un polinomio en z. Fijamos $\tau\lambda$. Según el lema 5 del capítulo anterior, la solución de la relación de recurrencia se puede expresar de la forma:

$$y_n = \sum_{j=1}^{l} p_j(n) z_j^n,$$

con z_j, $j = 1, \ldots, l$, las raíces complejas distintas de $\pi(\cdot; \tau\lambda)$, m_j sus multiplicidades, $l \leq r$ y $m_1 + \ldots + m_l = r$. Cada $p_j(\cdot)$ es un polinomio de orden $m_j - 1$, $1 \leq j \leq l$.

Si el parámetro λ es un complejo con parte real negativa, la solución de la ecuación diferencial $y' = \lambda y$ tiende a cero para t grande:

$$\lim_{t \to +\infty} y(t) = 0.$$

Esperamos el mismo comportamiento para la solución y_n del método multipasos (5.1). Esta propiedad se cumple si, y solamente si, todas las raíces z_j son de módulo inferior estricto a 1. La discusión anterior motiva la serie de definiciones siguientes.

Definición 9. *El método* (5.1) *es absolutamente estable en el punto*

$$\tau\lambda \in \mathbb{C},$$

si todas las raíces z_j del polinomio de estabilidad $\pi(\cdot; \tau\lambda)$ son de módulo inferior estricto a 1.

Definición 10. *La región $\mathcal{A}$ de estabilidad absoluta del método* (5.1) *es constituida por todos los puntos $\tau\lambda \in \mathbb{C}$, donde el método* (5.1) *es absolutamente estable.*

Definición 11. *El método* (5.1) *es A-estable, si su región de estabilidad $\mathcal{A}$ contiene la parte izquierda del plano complejo:*

$$\mathcal{A} \subset \{z \in \mathbb{C} | \Im z < 0\}.$$

5.3. Euler y A-estabilidad

La fórmula (5.1) permite tener Euler explícito como un caso particular:

$$y_{n+1} - y_n = \tau f_n.$$

El polinomio de estabilidad se escribe como

$$\pi(z; \lambda\tau) = z - 1 - \lambda\tau.$$

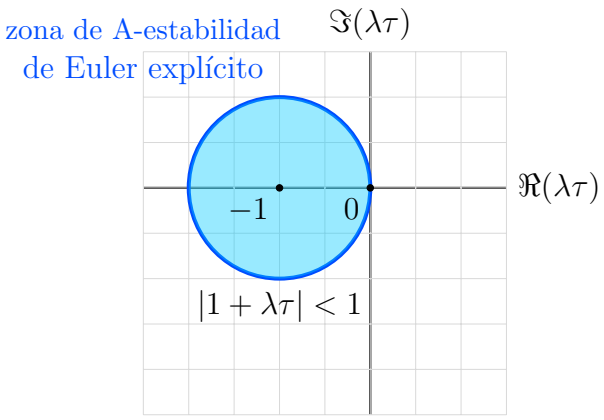
Tiene una única raíz simple:

$$z_1 = 1 + \lambda\tau.$$

Esta raíz z_1 es de módulo inferior estricto a 1 cuando

$$|1 + \lambda\tau| < 1.$$

Dibujamos, a continuación, la región de estabilidad. Volvemos a encontrar el resultado del primer capítulo. Comprobamos, entonces, que Euler explícito no es A-estable.



Volvemos ahora a introducir Euler implícito. En el formalismo de (5.1) se escribe como

$$y_{n+1} - y_n = \tau f_{n+1}.$$

El polinomio de estabilidad asociado es

$$\pi(z; \lambda\tau) = (1 - \lambda\tau)z - 1.$$

Tiene una única raíz simple:

$$z_1 = \frac{1}{1 - \lambda\tau}.$$

Esta raíz es de módulo inferior estricto a 1 cuando

$$\left| \frac{1}{1 - \lambda\tau} \right| < 1,$$

o, dicho de otra manera,

$$|1 - \lambda\tau| > 1.$$

Veamos que para cualquier valor de λ de parte real negativa, esta condición se cumple. Entonces, Euler implícito es A-estable.

5.4. Otros métodos implícitos

El método de Euler implícito, por muy estable que sea, es solamente de orden 1. Veamos en esta sección otros dos métodos implícitos conocidos que son de orden 2.

5.4.1. Método de trapecios (o Crank-Nicolson)

Este método es la variante implícita del método de Heun. Se escribe de la forma siguiente:

$$y_{n+1} - y_n = \frac{1}{2}\tau(f_n + f_{n+1}).$$

La idea es la misma que para el método de Heun. Se basa en la formulación integral de la ecuación diferencial, más la aproximación de la integral por la fórmula de trapecios. Se puede comprobar sin dificultad que es un método A-estable y de orden 2.

5.4.2. Método de punto medio implícito

Igualmente, este método es la variante implícita del punto medio explícito. Se escribe como

$$y_{n+1} - y_n = \tau f\left(\frac{t_n + t_{n+1}}{2}, \frac{y_n + y_{n+1}}{2}\right).$$

También se puede derivar de la fórmula de integración numérica del punto medio. Es un método A-estable y de orden 2. Cuando f es lineal, coincide con el método de Crank-Nicolson.

5.4.3. Métodos más sofisticados

Existen muchísimos más métodos implícitos que los muy básicos mencionados anteriormente. Algunos permiten obtener órdenes más elevados que uno o dos. Particularmente, existen variantes implícitas de los métodos de Runge-Kutta, y también métodos multipasos implícitos como Adams-Moulton o *Backward Finite Differences* (BDF). Además, se pueden combinar métodos explícitos e implícitos para tener métodos de predicción-corrección. Para más detalles ver, por ejemplo, [17], [37], [39], [46], [60].

5.5. Método de Newton

Los métodos implícitos se pueden escribir de la forma

$$\Psi(y_{n+1}) = 0,$$

donde Ψ es una función posiblemente no lineal que depende también de f, de τ, de los coeficientes del método y de los valores anteriores $y_n, \ldots, y_{n-r}$, $r \geq 0$. Por ejemplo, con Euler implícito tenemos a cada paso de tiempo la fórmula siguiente:

$$y_{n+1} - \tau f(t_{n+1}, y_{n+1}) - y_n = 0.$$

A nivel de implementación se necesita, entonces, una herramienta nueva para solucionar a cada paso de tiempo este problema. De hecho, cualquier método para calcular el cero de funciones sirve [60]. Lo más natural es el método de Newton.

5.5.1. Algoritmo

Supongamos que queremos solucionar

$$\Psi(y) = 0.$$

Se parte de una estimación y^0 de la solución y luego se calcula una secuencia $(y^k)_{k\geq 0}$ con la iteración siguiente:

$$\Psi(y^k) + \Psi'(y^k)(y^{k+1} - y^k) = 0,$$

que se puede volver a escribir así:

$$y^{k+1} = y^k - \frac{\Psi(y^k)}{\Psi'(y^k)}. \tag{5.2}$$

Asumimos que, para todo $k \geq 0$, $\Psi'(x_k) \neq 0$. Este método tiene una interpretación geométrica en términos de intersección de la tangente con la recta de ecuación $y = 0$. Paramos cuando el residuo es más pequeño que una tolerancia ϵ elegida:

$$|\Psi(y^K)| \leq \epsilon.$$

5.5.2. Convergencia cuadrática

El teorema de Kantorovich asegura que el método de Newton aproxima el cero con convergencia cuadrática, bajo algunas hipótesis de suavidad en Ψ y si y^0 esta cerca de y. Esta propiedad de convergencia cuadrática hace muy atractivo el algoritmo: entrega una aproximación con alta precisión en muy pocas iteraciones. No obstante, el algoritmo puede no entregar ninguna solución si lo iniciamos lejos del cero. Además, calcular la derivada exacta de Ψ puede ser desafiante para problemas complejos.

Enunciamos primero el teorema de Kantorovich (para la demostración ver, por ejemplo, el libro de E. Süli y D. F. Mayers [60]):

Teorema 12 (Kantorovich)**.** *Sea Ψ una función real de derivada segunda continua en un intervalo cerrado $[\xi - \delta; \xi + \delta]$ con $\delta > 0$. Asumimos*

$\Psi(\xi) = 0$ *y* $\Psi"(\xi) \neq 0$. *Asumimos, también, que existe una constante* $A > 0$ *por la cual tenemos la relación:*

$$\left| \frac{\Psi"(x)}{\Psi'(y)} \right| \leq A, \quad \forall x, y \in [\xi - \delta; \xi + \delta].$$

Entonces, para todo $x_0 \in \mathbb{R}$ *con*

$$|x_0 - \xi| \leq \min(\delta, 1/A),$$

la secuencia (x_k) *que corresponde al método de Newton* (5.2) *tiende hacia* ξ *con convergencia cuadrática: existe* $\mu \in \mathbb{R}$, $\mu > 0$, *que verifica*

$$\lim_{k \to +\infty} \frac{|x_{k+1} - \xi|}{|x_k - \xi|^2} = \mu.$$

5.5.3. Implicación práctica

Sea ahora la siguiente implementación en Python del método de Newton, que soluciona $f(x) = x^2 - 2$ para aproximar la raíz de 2.

```
import numpy as np
import matplotlib.pyplot as plt

# Cero de
def f(x, a):
    return x**2 - a

# Derivada
def df(x):
    return 2 * x

# Método de Newton
def newton_method(a, x0, tol=1e-10, max_iter=10):
    x = x0
    errors = []
    for _ in range(max_iter):
        x_new = x - f(x, a) / df(x)
        errors.append(abs(x_new - np.sqrt(a)))
        if abs(x_new - x) < tol:
            break
        x = x_new
```

```
    return errors

# Parámetros
a = 2.0
x0 = 1.5

# Utilizamos Newton
errors = newton_method(a, x0)
```

Listado 5: Newton en Python.

Esto nos entrega los errores siguientes:

Iteración k	$x_k - \xi$
0	0,5
1	0,0024531042935715 95
2	$2{,}1239014147411694{,}10^{-06}$
3	$1{,}5947243525715749{,}10^{-12}$
4	0,0

Observamos claramente la influencia de la convergencia cuadrática. Aproximadamente, el método permite tener el doble de cifras significativas de una iteración a la otra. A la cuarta iteración, el error esta debajo de la precisión aritmética usual.

5.5.4. Aplicación a Euler implícito

Mostramos aquí cómo el método sirve para Euler implícito. La función *fsolve* de Python se encarga automáticamente de aplicar el método de Newton.

```
import numpy as np
from scipy.optimize import fsolve

def implicit_euler(f, y0, t0, tf, tau):
    # Euler implícito

    # Parámetros:
    # f (function): EDO.
    # y0 (float): valor inicial.
    # t0 (float): tiempo inicial.
```

```
    # tf (float): tiempo final.
    # tau (float): paso de tiempo.

    # Devuelve:
    tuple: tn y yn.

    # Pasos de tiempo
    num_steps = int((tf - t0) / tau)

    # Inicialización
    t_values = np.linspace(t0, tf, num_steps + 1)
    y_values = np.zeros(num_steps + 1)
    y_values[0] = y0

    # Euler implícito
    for i in range(num_steps):
        # Define la ecuación implícita
        def implicit_equation(y_new):
            return y_new - y_values[i] - tau * f(t_values[
                i + 1], y_new)

        # Solución con Newton
        y_new_guess = y_values[i]
        y_new = fsolve(implicit_equation, y_new_guess)[0]

        # Guarda el valor
        y_values[i + 1] = y_new

    return t_values, y_values

# ejemplo de ecuación diferencial
def f(t, y):
    return -150 * y + 30

# Condición inicial
y0 = 0.2

# Tiempo
t0 = 0
tf = 1

# Paso de tiempo
tau = 0.02

```

```
# Solucionamos
t_values, y_values = implicit_euler(f, y0, t0, tf, tau)

# Escribimos los resultados
for t, y in zip(t_values, y_values):
    print(f"t = {t:.2f}, y = {y:.6f}")
```

Listado 6: **Euler implicito en Python.**

Obtenemos el resultado siguiente, que podemos comparar con el ejemplo inicial del capítulo.

t_n	y_n
0,00	0,200000
0,02	0,200000
0,04	0,200000
...	...
0,26	0,200000
0,28	0,200000
0,30	0,200000

5.6. Métodos paralelos en tiempo

Vimos que un método como Euler implícito permite, para una clase importante de ecuaciones o sistemas diferenciales, elegir un paso de tiempo τ grande sin perder estabilidad. Además, este método es de fácil implementación y permite obtener una solución sin muchos cálculos. Explicamos ahora cómo se puede sacar provecho de esto para hacer paralelización en tiempo.

5.6.1. Propagadores e integración en tiempo

Todavía para el problema de Cauchy

$$y' = f(t, y)$$

e $y(0) = y_0$ por $0 \leq t \leq T$, introducimos M pasos de tiempo largos ΔT: $\Delta T = T/M$, con $\tau \ll \Delta T$ y $M \ll N$. Introducimos la notación

$T_n = n\Delta T$, $n = 0, \ldots, M$. Primero, con un método poco preciso pero económico, construimos un propagador grueso $\mathcal{G}$ que sirve para predecir un esquema de la evolución temporal de la cantidad $y(t)$:

$$Y_{n+1} = \mathcal{G}(T_n, T_{n+1}; Y_n), \quad \forall n = 0, \ldots, M-1\,; Y_0 = y_0. \tag{5.3}$$

Ejemplo 13. *Tomamos $f(t, y) = \alpha y$, $\alpha \in \mathbb{R}$ ($y' = \alpha y$) y Euler implícito con paso de tiempo ΔT. En este caso:*

$$\mathcal{G}(T_n, T_{n+1}; Y_n) = \frac{Y_n}{1 - \alpha\Delta T}.$$

En segundo lugar, introducimos un propagador fino $\mathcal{F}$ mucho más preciso que $\mathcal{G}$ pero más costoso:

$$\widehat{Y}_{n+1} = \mathcal{F}(T_n, T_{n+1}; \widehat{Y}_n), \quad \forall n = 0, \ldots, M-1,\ \widehat{Y}^0 = Y(0). \tag{5.4}$$

Ejemplo 14. *Tomamos aún $f(t, y) = \alpha y$, $\alpha \in \mathbb{R}$ y la solución analítica de $y' = \alpha y$ en un intervalo de tamaño ΔT. En este caso:*

$$\mathcal{F}(T_n, T_{n+1}; Y_n) = \exp(\alpha\Delta T) Y_n.$$

En el caso general, se puede elegir un método de orden alto.

Ejemplo 15. *De nuevo, en el caso particular $f(t, y) = \alpha y$, $\alpha \in \mathbb{R}$ ($y' = \alpha y$), elegimos Euler implícito pero con paso de tiempo $\tau = \Delta T/p$, con p grande. En este caso:*

$$\mathcal{F}(T_n, T_{n+1}; Y_n) = \frac{Y_n}{(1 - \alpha\tau)^p}.$$

Aclaramos que los ejemplos anteriores son puramente ilustrativos. En casos un poco más generales no es fácil dar una expresión explícita de estos propagadores, a pesar de que siguen siendo bien definidos. Se pueden elegir métodos explícitos también a condición de que sean estables. Por el momento tenemos dos fórmulas de integración numérica en tiempo que son ambas secuenciales: el método (5.3) toma poco tiempo de cálculo pero no da una solución muy precisa, y el método (5.4) da una solución precisa pero con más costo computacional. Teniendo una máquina con M procesadores paralelos, se pueden combinar de forma astuta los dos métodos antagonistas y terminamos con un método preciso y rápido.

5.6.2. Idea *pararéelle*

La idea de paralelizar la resolución en tiempo de ecuaciones diferenciales parece haber nacido en 1964 con un trabajo de J. Nievergelt [51]. Para una perspectiva histórica sobre el tema y una descripción de diversos métodos ver, en particular, [29], [30]. Nos vamos a enfocar en un método sencillo, *pararéel*, que fue propuesto en 2001 por J. L. Lions, Y. Maday y G. Turinici [50]. Ideas similares se encuentran en un trabajo anterior (1997) de P. Saha, J. Stadel y S. Tremaine [55], y en un artículo publicado por C. Farhat y M. Chandesris en 2003 [26].

Seguiremos la presentación dada en el libro de M. Gander y T. Lunet [30] (o, alternativamente, en [3], [31]). Empezamos como en (5.3) para recorrer todo el intervalo de tiempo $[0, T]$ con el propagador grueso:

$$Y^0_{n+1} = \mathcal{G}(T_n, T_{n+1}; Y^0_n), \quad \forall n = 0, \ldots, M-1\,; \quad Y^0_0 = y(0).$$

Obtenemos una primera secuencia $Y^0_0, \ldots, Y^0_M$ que puede servir para predecir en cada intervalo de tiempo $(T^n; T^{n+1})$ una solución precisa con el propagador fino $\mathcal{F}$:

$$\widehat{Y}^0_{n+1} = \mathcal{F}(T_n, T_{n+1}; Y^0_n), \quad \forall n = 0, \ldots, N-1\,; \quad \widehat{Y}^0_0 = y(0).$$

Dado que toda la secuencia $Y^0_0, \ldots, Y^0_M$ es conocida, este paso puede ser completamente paralelizado en M procesadores. Sin embargo, debido a la diferencia de precisión entre el propagador fino $\mathcal{F}$ y el propagador grueso $\mathcal{G}$, es esperable que las soluciones no coincidan en los tiempos $T_1, \ldots, T_M$. Sea

$$\delta Y_{n+1} = \widehat{Y}^0_{n+1} - Y^0_{n+1}$$

el salto de solución en T_{n+1} que puede ser diferente de cero. Reducimos estos saltos con una nueva etapa secuencial, con la cual utilizamos únicamente el propagador grueso (para seguir siendo rápido):

$$Y^1_{n+1} = \underbrace{\mathcal{G}(T_n, T_{n+1}; Y^1_n)}_{\text{predicción}} + \underbrace{\delta Y_{n+1}}_{\text{corrección}}, \quad \forall n = 0, \ldots, M-1.$$

Corresponde a un paso de predicción-corrección, o corrección de defectos [58], donde la predicción gruesa (o el defecto) está corregida utilizando

la solución fina. Podemos escribirlo también como

$$Y_{n+1}^1 = \underbrace{\mathcal{G}(T_n, T_{n+1}; Y_n^1)}_{\text{predicción}} + \underbrace{\mathcal{F}(T_n, T_{n+1}; Y_n^0) - \mathcal{G}(T_n, T_{n+1}; Y_n^0)}_{\text{corrección}},$$

para $\forall n = 0, \ldots, M-1$. Como resultado obtenemos una nueva secuencia,

$$Y_0^1, \ldots, Y_M^1,$$

que, bajo algunas hipótesis, aproxima más precisamente la solución del problema de Cauchy inicial. Luego iteramos y volvemos a calcular nuevas soluciones finas en los M procesadores, utilizando $Y_0^1, \ldots, Y_M^1$ como nuevos valores iniciales.

5.6.3. Algoritmo *parareel*

Detallamos el método *parareel* de forma completa. Los componentes del algoritmo son:

- **Propagador fino ($\mathcal{F}$)**: Método de integración preciso (ejemplo: Runge-Kutta de orden alto).
- **Propagador grueso ($\mathcal{G}$)**: Método de integración rápido pero menos preciso (ejemplo: Euler).
- **Intervalo temporal**: $[0, T]$ dividido en M subintervalos $[T_n, T_{n+1}]$, con $T_0 = 0$ y $T_M = T$, para M procesadores.

El método está descrito en el algoritmo 1. Comentamos los pasos.

1. **Inicialización:** Se propaga la solución inicial y_0 usando solo el propagador grueso $\mathcal{G}$ en todo el intervalo temporal.
2. **Iteraciones *parareelles* en k:**
   - **Paso en paralelo:** Para cada subintervalo $[T_n, T_{n+1}]$ se calcula la solución usando el propagador fino $\mathcal{F}$ con la condición inicial Y_n. Este paso se paraleliza, ya que cada subintervalo es independiente en esta fase.

- **Corrección secuencial:** Se actualiza la solución en cada subintervalo usando la fórmula de corrección:

$$Y_{n+1}^{k+1} = \mathcal{G}(T_n, T_{n+1}; Y_{n+1}^k) + \mathcal{F}(T_n, T_{n+1}; Y_n^k) - \mathcal{G}(T_n, T_{n+1}; Y_n^k). \tag{5.5}$$

Este paso es secuencial, porque Y_{n+1}^{k+1} depende de Y_n^k.

3. **Convergencia:** Tras K iteraciones, la solución Y_n^K aproxima a la solución que se obtendría usando solo el propagador fino $\mathcal{F}$ en todo el intervalo.

Algoritmo 1 (*pararéel*)

Requerido:
- $f(t, y)$: Función del sistema de EDO;
- y_0: Condición inicial;
- $\mathcal{F}$: Propagador fino;
- $\mathcal{G}$: Propagador grueso;
- M: Número de subintervalos;
- K: Número máximo de iteraciones.

Garantizar: Solución aproximada Y_n en los puntos T_n.

Inicialización: $Y_0 = y(0)$
para desde $n = 0$ hasta $M - 1$ **hacer**
  $Y_{n+1} = \mathcal{G}(T_n, T_{n+1}; Y_n)$ ▷ Propagación gruesa secuencial
fin para
para desde $k = 0$ hasta $K - 1$ **hacer** ▷ Iteraciones *pararéelles*
  para desde $n = 0$ hasta $M - 1$ **hacer en paralelo**
    $\widehat{Y}_{n+1} = \mathcal{F}(T_n, T_{n+1}; Y_n)$ ▷ Propagación fina en paralelo
  fin para
  para desde $n = 0$ hasta $M - 1$ **hacer** ▷ Corrección secuencial
    $Y_{n+1} = \mathcal{G}(T_n, T_{n+1}; Y_n) + \widehat{Y}_{n+1} - Y_{n+1}$
  fin para
fin para
Devolver Y_n para $n = 0, \ldots, M$.

Hacemos $k \to +\infty$ en la fórmula (5.5). Suponiendo que haya convergencia hasta una secuencia $(Y_0^\infty, \ldots, Y_M^\infty)$ que verifica para cualquier

$0 \leq n < M$:

$$\begin{aligned} & Y_{n+1}^{\infty} \\ = \; & \mathcal{G}(T_n, T_{n+1}; Y_n^{\infty}) + \mathcal{F}(T_n, T_{n+1}; Y_n^{\infty}) \\ & -\mathcal{G}(T_n, T_{n+1}; Y_n^{\infty}) \\ = \; & \mathcal{F}(T_n, T_{n+1}; Y_n^{\infty}). \end{aligned}$$

Por lo tanto, $(Y_0^{\infty}, \ldots, Y_M^{\infty})$ es la solución obtenida a través de cálculos secuenciales (costosos) con el método numérico preciso. Notar que la aceleración en paralelo es efectiva siempre que $K \ll M$, donde K es el número de iteraciones *pararéelles* necesarias para la convergencia, y siempre que el cálculo grueso sea mucho más económico que el fino. De hecho, se muestra sin dificultad que el algoritmo converge en un máximo de $K = M$ iteraciones [30]. El estudio preciso de estabilidad y convergencia del algoritmo se puede encontrar en [30], [31]. El tema de estabilidad es particularmente delicado: de cierta forma, *pararéel* es un método multipasos y se enfrenta a las barreras de Dahlquist, que son una serie de teoremas generales sobre los límites de orden de métodos multipasos estables (ver [34], [37], [39], [60] para más detalles).

5.6.4. *Pararéel* y disparos múltiples

Ahora trataremos de entender mejor cómo está construida la etapa de corrección en *pararéel*. Para este propósito, es interesante hacer el vínculo con el método de disparos múltiples (*multiple shooting*) de P. Chartier y B. Philippe [9], siguiendo la presentación de [30], [31]. Partamos de la iteración (5.5) y hacemos lo siguiente:

$$\mathcal{G}(T_n, T_{n+1}, Y_n^{k+1}) - \mathcal{G}(T_n, T_{n+1}, Y_n^{k}) \simeq y(T_{n+1}; Y_n^{k+1}) - y(T_{n+1}; Y_n^{k}),$$

donde $y(T_{n+1}; Y)$ denota la solución exacta en T_{n+1} del problema de Cauchy $y' = f(t, y)$, $y(T_n) = Y$. Lo combinamos con una fórmula de Taylor para obtener:

$$\mathcal{G}(T_n, T_{n+1}, Y_n^{k+1}) - \mathcal{G}(T_n, T_{n+1}, Y_n^{k}) \simeq \frac{\partial y}{\partial Y}(T_{n+1}; Y_n^{k+1} - Y_n^{k}).$$

Nos da la fórmula siguiente, alternativa a (5.5):

$$Y_{n+1}^{k+1} = \mathcal{F}(T_n, T_{n+1}; Y_n^{k}) + \frac{\partial y}{\partial Y}(T_{n+1}; Y_n^{k+1} - Y_n^{k}).$$

Esta es la fórmula a la base del método de disparos múltiples (*multiple shooting*) de P. Chartier y B. Philippe [9]. La podemos entender mejor de la forma siguiente.

5.6.5. Formulaciones globales

Introducimos primero el vector columna $\mathbf{Y}$ que contiene las soluciones en cada subintervalo temporal:

$$\mathbf{Y} = \begin{bmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_M \end{bmatrix},$$

donde Y_n es la solución aproximada en el tiempo T_n. Sea la siguiente función $\mathbf{F}(\mathbf{Y})$ definida como:

$$\mathbf{F}(\mathbf{Y}) = \begin{bmatrix} F_1(\mathbf{Y}) \\ F_2(\mathbf{Y}) \\ \vdots \\ F_M(\mathbf{Y}) \end{bmatrix} = \begin{bmatrix} Y_1 - \mathcal{F}(T_0, T_1; Y_0) \\ Y_2 - \mathcal{F}(T_1, T_2, Y_1) \\ \vdots \\ Y_M - \mathcal{F}(T_{M-1}, T_M; Y_{M-1}) \end{bmatrix},$$

donde $\mathcal{F}(T_{n-1}, T_n; Y_{n-1})$ es la solución obtenida con el propagador fino en el subintervalo $[T_{n-1}, T_n]$. Para simplificar, podemos suponer que el propagador corresponde a la solución exacta. La ecuación

$$\mathbf{F}(\mathbf{Y}) = \mathbf{0}$$

representa el sistema de ecuaciones que debe resolverse para encontrar la solución fina/exacta del problema de Cauchy. La idea de [9] es utilizar el método de Newton visto previamente para solucionar estas ecuaciones. La iteración k del método de Newton se expresa en forma compacta como:

$$\mathbf{Y}^{k+1} = \mathbf{Y}^k - [\mathbf{J}(\mathbf{Y}^k)]^{-1}\mathbf{F}(\mathbf{Y}^k),$$

donde $\mathbf{J}(\mathbf{Y}^k)$ es la matriz Jacobiana de $\mathbf{F}$ evaluada en $\mathbf{Y}^k$.

Matriz jacobiana $\mathbf{J}(\mathbf{Y})$

Dado que $F_n(\mathbf{Y}) = Y_n - \mathcal{F}(T_{n-1}, T_n; Y_{n-1})$, la matriz jacobiana tiene la siguiente estructura:

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -\dfrac{\partial \mathcal{F}(T_1, T_2; Y_1)}{\partial Y_1} & 1 & 0 & \cdots & 0 \\ 0 & -\dfrac{\partial \mathcal{F}(T_2, T_3; Y_2)}{\partial Y_2} & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -\dfrac{\partial \mathcal{F}(T_{M-1}, T_M; Y_{M-1})}{\partial Y_{M-1}} & 1 \end{bmatrix}}_{=\mathbf{J}(\mathbf{Y})}.$$

Método de Newton componente por componente

Entonces, la iteración k

$$[\mathbf{J}(\mathbf{Y}^k)](\mathbf{Y}^{k+1} - \mathbf{Y}^k) = -\mathbf{F}(\mathbf{Y}^k)$$

del método de Newton puede escribirse componente por componente como:

$$\begin{cases} Y_1^{k+1} - Y_1^k = -Y_1^k + \mathcal{F}(T_0, T_1; Y_0^k), \\ -\dfrac{\partial \mathcal{F}(T_1, T_2; Y_1)}{\partial Y_1}(Y_1^{k+1} - Y_1^k) + Y_2^{k+1} - Y_2^k \\ = -Y_2^k + \mathcal{F}(T_1, T_2; Y_1^k), \\ \vdots \\ -\dfrac{\partial \mathcal{F}(T_{M-1}, T_M; Y_{M-1})}{\partial Y_{M-1}}(Y_{M-1}^{k+1} - Y_{M-1}^k) + Y_M^{k+1} - Y_M^k \\ = -Y_M^k + \mathcal{F}(T_{M-1}, T_M; Y_{M-1}^k). \end{cases}$$

Simplificamos:

$$\begin{cases} Y_1^{k+1} = \mathcal{F}(T_0, T_1; Y_0^k), \\ Y_2^{k+1} = \mathcal{F}(T_1, T_2; Y_1^k) + \dfrac{\partial \mathcal{F}(T_1, T_2; Y_1)}{\partial Y_1}(Y_1^{k+1} - Y_1^k), \\ \vdots \\ Y_M^{k+1} = \mathcal{F}(T_{M-1}, T_M; Y_{M-1}^k) \\ + \dfrac{\partial \mathcal{F}(T_{M-1}, T_M; Y_{M-1})}{\partial Y_{M-1}}(Y_{M-1}^{k+1} - Y_{M-1}^k). \end{cases}$$

Volvemos a encontrar el método de [9]. Es paralelizable también y tiene las ventajas de los métodos de Newton, particularmente, la convergencia cuadrática. No obstante, tiene desventajas: se tienen que calcular de las jacobianas y la convergencia ocurre solo si estamos cerca de la solución (ver [30] para más detalles).

Pararéel en formulación matricial

En el caso de que el propagador grueso $\mathcal{G}$ sea una aplicación lineal, veamos también que la iteración k *pararéelle* se formula así:

$$[\mathbf{J}_{\mathcal{G}}](\mathbf{Y}^{k+1} - \mathbf{Y}^k) = -\mathbf{F}(\mathbf{Y}^k),$$

donde

$$\underbrace{\begin{bmatrix} 1 & 0 & 0 & \cdots & 0 \\ -\mathcal{G}(T_1, T_2; \cdot) & 1 & 0 & \cdots & 0 \\ 0 & -\mathcal{G}(T_2, T_3; \cdot) & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & -\mathcal{G}(T_{M-1}, T_M; \cdot) & 1 \end{bmatrix}}_{=\mathbf{J}_{\mathcal{G}}}.$$

En este caso, $\mathbf{J}_{\mathcal{G}}$ se puede ver como una aproximación gruesa de la jacobiana exacta $\mathbf{J}(\mathbf{Y})$ y el método *pararéel* un método de cuasi Newton o de Newton inexacto para solucionar el sistema $\mathbf{F}(\mathbf{Y}) = \mathbf{0}$.

5.6.6. Ejemplo en Python

Retomamos el ejemplo del capítulo 3 con Van der Pol y los mismos parámetros. Hallamos $x, y : [0, T] \to \mathbb{R}$ $(T = 10)$ soluciones de

$$x' = f_1(x, y), \quad y' = f_2(x, y),$$

con condiciones iniciales

$$x(0) = y(0) = 1$$

y las siguientes expresiones de f_1 y f_2:

$$f_1(x, y) = y, \quad f_2(x, y) = 5(1 - x^2)y - x.$$

Para paralelizar, empleamos la librería *multiprocessing* de Python.

```
import numpy as np
from scipy.integrate import solve_ivp
import matplotlib.pyplot as plt
from multiprocessing import Pool

# Definición del sistema de Van der Pol
def f(t, u):
    x, y = u
    dxdt = y
    dydt = 5 * (1 - x**2) * y - x
    return [dxdt, dydt]

# Propagador fino (F): solve_ivp (Runge-Kutta 4-5)
def F(args):
    t0, t1, u0 = args
    sol = solve_ivp(f, [t0, t1], u0, t_eval=[t1], method='RK45')
    return sol.y[:, 0]  # Devuelve el valor final como arreglo

# Propagador grueso (G): Euler explícito
def G(t0, t1, u0, dt=0.05):
    t = t0
    u = u0.copy()  # u0 es un arreglo de dimensión 2
    while t < t1:
        u += dt * np.array(f(t, u))
```

```
25          t += dt
26      return u  # Devuelve un arreglo de dimensión 2
27
28  # Parámetros de simulación
29  T = 20.0  # Tiempo final
30  N = 40  # Número de subintervalos
31  T_n = np.linspace(0, T, N+1)
32  u0 = np.array([1.0, 1.0])  # Condiciones iniciales: x(0) =
        y(0) = 1
33  max_iter = 6  # Número de iteraciones
34
35  # Inicialización (solo G)
36  U = np.zeros((N+1, 2))
37  U[0] = u0
38  for n in range(N):
39      U[n+1] = G(T_n[n], T_n[n+1], U[n])
40
41  # Lista para guardar soluciones intermedias
42  soluciones_intermedias = [U.copy()]
43
44  # Iteraciones
45  for k in range(max_iter):
46      U_new = U.copy()
47      # Preparamos los argumentos para el propagador fino F
48      args_F = [(T_n[n], T_n[n+1], U[n]) for n in range(N)]
49
50      # Usamos multiprocessing para paralelizar el cálculo
            de F
51      with Pool() as pool:
52          F_vals = np.array(pool.map(F, args_F))
53
54      # Paso 2: Actualización secuencial
55      for n in range(N):
56          U_new[n+1] = G(T_n[n], T_n[n+1], U_new[n]) +
                F_vals[n] - G(T_n[n], T_n[n+1], U[n])
57      U = U_new
58      soluciones_intermedias.append(U.copy())
59
60  # Solución fina (referencia) usando solo el propagador
        fino
61  sol_fina = solve_ivp(f, [0, T], u0, t_eval=T_n, method='
        RK45')
62
63  # Graficar las soluciones para x(t) e y(t)
```

```
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 8))

# Graficar x(t)
ax1.plot(sol_fina.t, sol_fina.y[0], 'k--', label='Solución fina (RK45)', linewidth=2)
for i, sol in enumerate(soluciones_intermedias):
    ax1.plot(T_n, sol[:, 0], 'o-', label=f'Iteración {i}')
ax1.set_xlabel('Tiempo (t)')
ax1.set_ylabel('x(t)')
ax1.set_title('Convergencia del método pararéel para x(t) en Van der Pol')
ax1.legend()
ax1.grid()
# Graficar y(t)
ax2.plot(sol_fina.t, sol_fina.y[1], 'k--', label='Solución fina (RK45)', linewidth=2)
for i, sol in enumerate(soluciones_intermedias):
    ax2.plot(T_n, sol[:, 1], 'o-', label=f'Iteración {i}')
ax2.set_xlabel('Tiempo (t)')
ax2.set_ylabel('y(t)')
ax2.set_title('Convergencia del método pararéel para y(t) en Van der Pol')
ax2.legend()
ax2.grid()
plt.tight_layout()
plt.show()
```

Listado 7: ***Pararéel* en Python.**

Los resultados se pueden observar en la figura 4. Se nota que el resultado final se puede encontrar en pocas iteraciones (aproximadamente $K = 4 \ll 40 = M$), en un problema donde no es fácil predecir la solución. No obstante, es más difícil lograr tener convergencia entre los tiempos 8 y 10: las dos primeras iteraciones proporcionan una solución lejana de la solución fina. Otras implementaciones de *pararéel* y variantes están disponibles en el libro de M. J. Gander y T. Lunet [30] (en Matlab) y en la web[6].

[6]En Python: en https://github.com/KVuillemot/Project_M1_Parallelisation_en_temps y en https://parallelwindfarms.github.io/parareal/; en Julia: en https://arxiv.org/abs/1706.08569, y en C++ en: https://github.com/NLESC-JCER/parareal.

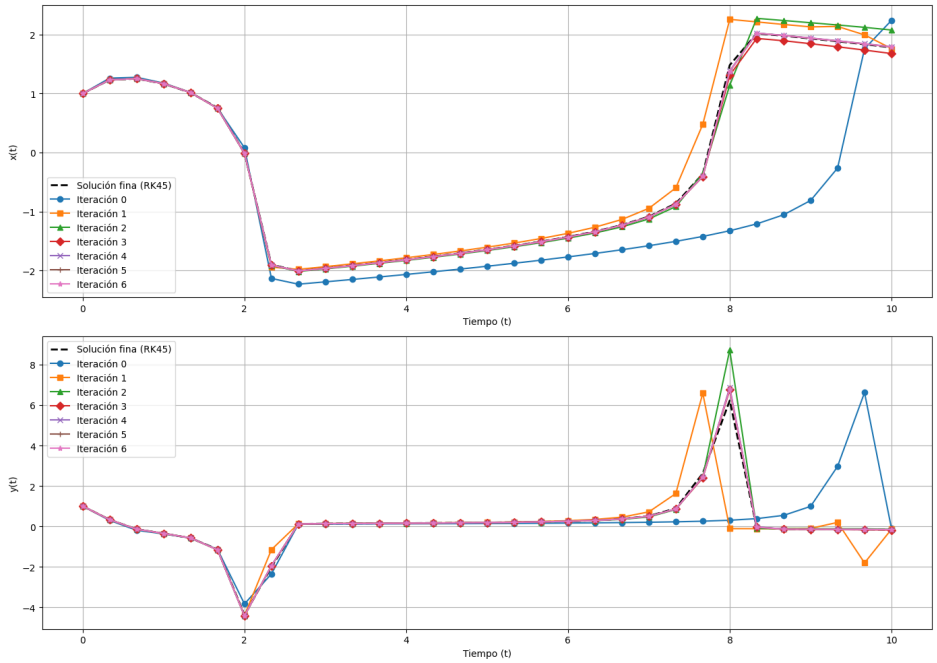


Figura 4: Resolución de Van der Pol con *pararéel*.

Capítulo 6
Métodos conservativos

Este capítulo presenta métodos numéricos que permiten conservar cantidades, o invariantes, tras el tiempo. Un invariante típico puede ser la energía de un sistema. Veamos primero una clase importante de sistemas físicos, en un sentido amplio, que tienen propiedades notables.

6.1. Sistemas hamiltonianos

Sea $d \geq 1$. Un sistema hamiltoniano es un sistema dinámico que puede describirse mediante las ecuaciones de Hamilton [2], [36]. Estas ecuaciones son de la forma:

$$\frac{dq_i}{dt} = \frac{\partial H}{\partial p_i}, \quad \frac{dp_i}{dt} = -\frac{\partial H}{\partial q_i}, \quad i = 1, \ldots, d,$$

donde

- $q = (q_1, \ldots, q_d)$ son las coordenadas generalizadas;
- $p = (p_1, \ldots, p_d)$ son los momentos conjugados;
- $H(\cdot, \cdot)$ es la función hamiltoniana:

  $$H : \mathcal{D} \ni (p, q) \mapsto H(p, q) \in \mathbb{R},$$

  donde $\mathcal{D}$ es un subconjunto de $\mathbb{R}^{2d}$.

Ejemplo 16. *Consideremos un sistema físico simple: el oscilador armónico. En este caso, $d = 1$ y notamos simplemente $p = p_1$ y $q = q_1$. El hamiltoniano para este sistema es:*

$$H(p, q) = \frac{\omega^2}{2} p^2 + \frac{1}{2} q^2,$$

donde $\omega > 0$ es un parámetro físico. Este modelo puede representar un resorte, un circuito electrónico, etc. Las ecuaciones de Hamilton para este sistema son:

$$\frac{dq}{dt} = \frac{\partial H}{\partial p} = \omega^2 p, \quad \frac{dp}{dt} = -\frac{\partial H}{\partial q} = -q.$$

Deducimos, finalmente,

$$\frac{d^2q}{dt^2} = \omega^2 \frac{dp}{dt} = -\omega^2 q,$$

que podemos volver a escribir

$$\frac{d^2q}{dt^2} + \omega^2 q = 0,$$

que es la ecuación diferencial clásica de segundo orden.

A partir de ahora, por comodidad, notaremos a veces en este capítulo

$$\dot{y} = \frac{dy}{dt}, \quad \ddot{y} = \frac{d^2y}{dt^2}$$

para las derivadas en tiempo de orden uno y dos, respectivamente, donde y es una función del tiempo. El hamiltoniano tiene una propiedad notable: se conserva en el transcurso del tiempo. Se dice que es un invariante o una primera integral del sistema. Precisamente:

Teorema 13. *Sea $d \geq 1$ y $H(\cdot)$ un hamiltoniano. Para todo $T \geq 0$ se cumple*

$$H(T) = H(0). \tag{6.1}$$

Demostración. Sea $d \geq 1$, $0 \leq t \leq T$ y H un hamiltoniano. Aplicamos la regla de la cadena

$$\frac{d}{dt} H(p(t), q(t)) = \sum_{i=1}^{d} \frac{\partial H}{\partial p_i} \dot{p}_i(t) + \sum_{i=1}^{d} \frac{\partial H}{\partial q_i} \dot{q}_i(t).$$

Las ecuaciones de Hamilton permiten tener

$$\frac{d}{dt} H(p(t), q(t)) = \sum_{i=1}^{d} \dot{q}_i(t)\dot{p}_i(t) - \sum_{i=1}^{d} \dot{p}_i(t)\dot{q}_i(t) = 0.$$

Terminamos integrando la ecuación anterior entre 0 y T. □

Veamos ahora varios ejemplos.

Ejemplo 17 (El péndulo)**.** *En este caso tenemos*

$$H(p,q) = \frac{1}{2}\omega^2 p^2 - \cos(q), \tag{6.2}$$

donde $\omega > 0$ es un parámetro físico y q representa un ángulo. Entonces, el sistema diferencial asociado es

$$\dot{p} = -\operatorname{sen}(q), \quad \dot{q} = \omega^2 p.$$

Se puede, de nuevo, escribir como una ecuación diferencial de segundo orden:

$$\ddot{q} + \omega^2 \operatorname{sen}(q) = 0.$$

Si las oscilaciones son muy pequeñas, podemos aproximar $\operatorname{sen}(q) \simeq q$ *y volvemos a encontrar el oscilador armónico.*

Ejemplo 18 (El problema de dos cuerpos)**.** *Se trata de dos cuerpos puntuales (p_1, q_1) y (p_2, q_2) en el espacio con la fuerza gravitacional [36]. El hamiltoniano en este caso es*

$$H(p,q) = \frac{1}{2}(p_1^2 + p_2^2) - (q_1^2 + q_2^2)^{-\frac{1}{2}}.$$

Ejemplo 19 (El problema de Hénon-Heiles)**.** *Es otro problema en astrofísica relacionado con dinámica estelar [36]. El hamiltoniano se define como:*

$$H(p,q) = \frac{1}{2}(p_1^2 + p_2^2) + U(q),$$

con

$$U(q) = \frac{1}{2}(q_1^2 + q_2^2) + q_1^2 q_2 - \frac{1}{3}q_2^3.$$

6.2. Flujos simplécticos

La estructura simpléctica es una propiedad geométrica fundamental de los sistemas hamiltonianos. En el espacio de fase, que es el espacio de todas las posibles coordenadas y momentos, la estructura simpléctica

se define mediante una forma diferencial cerrada y no degenerada llamada forma simpléctica. La preservación de esta estructura es crucial porque garantiza que ciertas propiedades físicas, como la conservación de la energía, se mantengan a lo largo del tiempo. Nos vamos a enfocar en el caso $d = 1$, donde podemos simplificar bastante la presentación. Notamos, entonces, $p = p_1$ y $q = q_1$. Para el caso general ver, por ejemplo, [36] y otras referencias clásicas [2].

Primero definimos una aplicación simpléctica como una aplicación que preserva localmente el área orientada.

Definición 12. *Sea $\mathcal{D}$ un conjunto abierto en $\mathbb{R}^2$. Una aplicación diferencial $g : \mathcal{D} \to \mathbb{R}^2$ es simpléctica si su matriz jacobiana J satisface*

$$\det(J(p, q)) = 1,$$

para todo $(p, q) \in \mathcal{D}$.

Típicamente, una rotación del plano es simpléctica, pero una homotecia de razón diferente de 1 no lo es. Veamos ahora lo que pasa con el flujo de un sistema hamiltoniano.

Sea (p_0, q_0) una coordenada inicial cualquiera en $\mathcal{D}$. Para todo $t \geq 0$, definimos el flujo de un sistema hamiltoniano como:

$$\varphi_t(p_0, q_0) = \begin{pmatrix} p(t; p_0, q_0) \\ q(t; p_0, q_0) \end{pmatrix}.$$

Observamos que

$$\varphi_t : \mathcal{D} \to \mathbb{R}^2$$

y permite saber, al tiempo t, cómo el sistema hamiltoniano ha transformado el punto (p_0, q_0). Consideramos t como un parámetro de φ. Notamos J_t la matriz jacobiana de φ_t. Tenemos

$$J_t(p_0, q_0) = \begin{pmatrix} \partial_{p_0} p & \partial_{q_0} p \\ \partial_{p_0} q & \partial_{q_0} q \end{pmatrix}.$$

Para aliviar un poco las notaciones hemos omitido la dependencia en t a la derecha y notado

$$\partial_{p_0} p = \frac{\partial p}{\partial p_0}, \quad \partial_{q_0} p = \frac{\partial p}{\partial q_0}, \quad \partial_{p_0} q = \frac{\partial q}{\partial p_0}, \quad \partial_{q_0} q = \frac{\partial q}{\partial q_0}.$$

La derivada en tiempo de J_t es dada por:

$$\dot{J}_t(p_0, q_0) = \begin{pmatrix} \partial_{p_0}\dot{p} & \partial_{q_0}\dot{p} \\ \partial_{p_0}\dot{q} & \partial_{q_0}\dot{q} \end{pmatrix}.$$

Demostramos el teorema siguiente, que va a tener un rol fundamental [36] y que asegura que el flujo de un sistema hamiltoniano es simpléctico.

Teorema 14 (Poincaré, 1899)**.** *Supongamos que el hamiltoniano H sea de clase C^2 en su dominio $\mathcal{D}$. Entonces, el jacobiano J_t del flujo φ_t verifica*

$$\det(J_t) = 1,$$

para todo $t \geq 0$.

Demostración. Consideremos la ecuación diferencial que proviene del hamiltoniano:

$$\begin{pmatrix} \dot{p} \\ \dot{q} \end{pmatrix} = \begin{pmatrix} -\partial_q H \\ \partial_p H \end{pmatrix}.$$

Hemos utilizado la notación

$$\partial_p H = \frac{\partial H}{\partial p}, \quad \partial_q H = \frac{\partial H}{\partial q}.$$

Derivamos con respecto a (p_0, q_0):

$$\begin{pmatrix} \partial_{p_0}\dot{p} & \partial_{q_0}\dot{p} \\ \partial_{p_0}\dot{q} & \partial_{q_0}\dot{q} \end{pmatrix} = \begin{pmatrix} -\partial_{p_0}(\partial_q H) & -\partial_{q_0}(\partial_q H) \\ \partial_{p_0}(\partial_p H) & \partial_{q_0}(\partial_p H) \end{pmatrix}.$$

Aplicamos la regla de la cadena:

$$\partial_{p_0}(\partial_q H) = (\partial_{pq} H)\partial_{p_0} p + (\partial_{qq} H)\partial_{p_0} q.$$

De la misma forma:

$$\begin{aligned} \partial_{q_0}(\partial_q H) &= (\partial_{pq} H)\partial_{q_0} p + (\partial_{qq} H)\partial_{q_0} q, \\ \partial_{p_0}(\partial_p H) &= (\partial_{pp} H)\partial_{p_0} p + (\partial_{qp} H)\partial_{p_0} q, \\ \partial_{q_0}(\partial_p H) &= (\partial_{pp} H)\partial_{q_0} p + (\partial_{qp} H)\partial_{q_0} q. \end{aligned}$$

Llegamos, entonces, a

$$\begin{pmatrix} \partial_{p_0}\dot{p} & \partial_{q_0}\dot{p} \\ \partial_{p_0}\dot{q} & \partial_{q_0}\dot{q} \end{pmatrix} = \begin{pmatrix} -\partial_{pq}H & -\partial_{qq}H \\ \partial_{pp}H & \partial_{qp}H \end{pmatrix} \begin{pmatrix} \partial_{p_0}p & \partial_{q_0}p \\ \partial_{p_0}q & \partial_{q_0}q \end{pmatrix},$$

que podemos formular como

$$\dot{J}_t = \Xi_H J_t,$$

donde

$$\Xi_H := \begin{pmatrix} -\partial_{pq}H & -\partial_{qq}H \\ \partial_{pp}H & \partial_{qp}H \end{pmatrix}.$$

Ahora, utilizando la fórmula de Jacobi, podemos calcular:

$$\frac{d}{dt}(\det J_t) = \mathrm{tr}(\mathrm{Adj}(J_t)\dot{J}_t) = \mathrm{tr}(\mathrm{Adj}(J_t)\Xi_H J_t),$$

donde $\mathrm{Adj}(J_t)$ es la matriz adjunta de J_t (traspuesta de la matriz de cofactores [57]). Ahora, utilizando las propiedades de la traza y la relación

$$\mathrm{Adj}(J_t)J_t = (\det J_t)I,$$

con I la identidad en $\mathbb{R}^{2\times 2}$ [57], llegamos a

$$\mathrm{tr}(\mathrm{Adj}(J_t)\Xi_H J_t) = \mathrm{tr}(\mathrm{Adj}(J_t)J_t\Xi_H) = (\det J_t)\mathrm{tr}(\Xi_H).$$

Asumiendo que H es $\mathcal{C}^2$, el teorema de Schwarz implica

$$\mathrm{tr}(\Xi_H) = -H_{qp} + H_{pq} = -H_{pq} + H_{pq} = 0.$$

Finalmente:

$$\frac{d}{dt}(\det J_t) = 0.$$

Además,

$$\det(J_0) = 1.$$

Entonces, deducimos de lo anterior:

$$\det(J_t) = 1,$$

para todo $t \geq 0$. □

Observación 15. *Se verifica que*

$$\Xi_H = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \partial_{pp} H & \partial_{qp} H \\ \partial_{pq} H & \partial_{qq} H \end{pmatrix}.$$

Entonces, Ξ_H es el producto de una matriz de rotación por la hessiana de H. La misma matriz de rotación tiene un rol importante para sistemas hamiltonianos más generales [36].

De forma recíproca, se puede demostrar, bajo unas hipótesis mínimas, que toda ecuación diferencial autónoma cuyo flujo es simpléctico es localmente hamiltoniana [36].

6.3. Métodos simplécticos

Los métodos simplécticos son técnicas numéricas que preservan la estructura simpléctica de los sistemas hamiltonianos; esto significa que conservan el área en el espacio de fase, lo cual es una propiedad fundamental de los sistemas hamiltonianos tal como lo hemos visto anteriormente (ver también [36] para una exposición mucho más detallada).

6.3.1. El método de Euler simpléctico

Para un sistema hamiltoniano, el método de Euler simpléctico se puede definir como:

$$\begin{aligned} p_{n+1} &= p_n - \tau \partial_q H(p_{n+1}, q_n), \\ q_{n+1} &= q_n + \tau \partial_p H(p_{n+1}, q_n), \end{aligned} \tag{6.3}$$

con $\tau > 0$ el paso de tiempo y con $(t_n)_{0 \leq n \leq N}$ una subdivisión del intervalo $[0, T]$, que verifica:

$$\tau = t_{n+1} - t_n, \tag{6.4}$$

para todo $0 \leq n < N$. Es una forma de combinar Euler explícito y Euler implícito, vistos anteriormente. Se puede comprobar, sin mayor dificultad, que es convergente de orden 1 bajo las mismas hipótesis que en los casos anteriores. Ahora probamos que Euler simpléctico es simpléctico.

Teorema 15. *Supongamos que el hamiltoniano es C^2. El método* (6.3) *verifica, para todo $\tau > 0$ y para todo $0 \leq n < N$:*

$$\det\left(\frac{\partial(p_{n+1}, q_{n+1})}{\partial(p_n, q_n)}\right) = 1, \tag{6.5}$$

donde

$$\frac{\partial(p_{n+1}, q_{n+1})}{\partial(p_n, q_n)}$$

es la jacobiana de la aplicación $(p_n, q_n) \to (p_{n+1}, q_{n+1})$.

Demostración. Utilizamos la regla de la cadena y derivamos la primera ecuación de (6.3) en p_n:

$$\frac{\partial p_{n+1}}{\partial p_n} = 1 - \tau \partial_{pq} H \frac{\partial p_{n+1}}{\partial p_n},$$

donde $\partial_{qp} H = \partial_{qp} H(p_{n+1}, q_n)$. Esto nos da

$$\frac{\partial p_{n+1}}{\partial p_n} = \frac{1}{1 + \tau \partial_{pq} H}.$$

De la misma forma, tenemos

$$\frac{\partial p_{n+1}}{\partial q_n} = -\tau \partial_{pq} H \frac{\partial p_{n+1}}{\partial q_n} - \tau \partial_{qq} H,$$

y deducimos

$$\frac{\partial p_{n+1}}{\partial q_n} = \frac{-\tau \partial_{qq} H}{1 + \tau \partial H_{pq}}.$$

Ahora, derivando la segunda ecuación, obtenemos:

$$\frac{\partial q_{n+1}}{\partial p_n} = \tau \partial_{pp} H \frac{\partial p_{n+1}}{\partial p_n}.$$

Luego, obtenemos

$$\frac{\partial q_{n+1}}{\partial p_n} = \frac{\tau \partial_{pp} H}{1 + \tau \partial_{pq} H}.$$

Finalmente, calculamos:

$$\frac{\partial q_{n+1}}{\partial q_n} = 1 + \tau \partial_{pp} H \frac{\partial p_{n+1}}{\partial q_n} + \tau \partial_{qp} H,$$

y luego, utilizando de nuevo lo anterior:

$$\frac{\partial q_{n+1}}{\partial q_n} = 1 + \tau\partial_{pp}H\left(\frac{-\tau\partial_{qq}H}{1+\tau\partial_{pq}H}\right) + \tau\partial_{qp}H.$$

Lo volvemos a escribir así:

$$\frac{\partial q_{n+1}}{\partial q_n} = 1 + \tau\partial_{qp}H - \frac{\tau^2\partial_{pp}H\partial_{qq}H}{1+\tau\partial_{pq}H}.$$

Tenemos, entonces, la siguiente expresión para el flujo discreto:

$$\left(\frac{\partial(p_{n+1},q_{n+1})}{\partial(p_n,q_n)}\right) = \begin{pmatrix} \dfrac{1}{1+\tau\partial_{pq}H} & \dfrac{-\tau\partial_{qq}H}{1+\tau\partial_{pq}H} \\ \dfrac{\tau\partial_{pp}H}{1+\tau\partial_{pq}H} & 1+\tau\partial_{qp}H - \dfrac{\tau^2\partial_{pp}H\partial H_{qq}}{1+\tau\partial H_{pq}} \end{pmatrix}.$$

Y ahora calculamos

$$\begin{aligned} & \det\left(\frac{\partial(p_{n+1},q_{n+1})}{\partial(p_n,q_n)}\right) \\ = \;& 1 - \frac{\tau^2\partial_{pp}H\partial_{qq}H}{(1+\tau\partial_{qp}H)(1+\tau\partial_{qp}H)} + \frac{\tau^2\partial_{pp}H\partial_{qq}H}{(1+\tau\partial_{qp}H)(1+\tau\partial_{qp}H)} = 1, \end{aligned}$$

donde hemos utilizado la simetría de la hessiana de H, dada por el teorema de Schwarz, y la hipótesis H pertenece a C^2. Concluimos que el método (6.3) es simpléctico. □

Observación 16. *Mostrar la extensión del teorema anterior en el caso $d \geq 2$ resulta ser delicado. Se necesitan, en este caso, técnicas bastante más sofisticadas que las que se brindan en este capítulo introductorio (ver [36] por los detalles).*

Observación 17. *Por simetría, se puede considerar también el otro método de Euler simpléctico*

$$\begin{aligned} p_{n+1} &= p_n - \tau\frac{\partial H}{\partial q}(q_n, p_{n+1}), \\ q_{n+1} &= q_n + \tau\frac{\partial H}{\partial p}(q_n, p_{n+1}), \end{aligned}$$

y mostrar que tiene las mismas propiedades que el método anterior (es también de orden 1).

Observación 18. *Cuando el hamiltoniano se descompone en una suma de la forma*

$$H(p,q) = T(p) + U(q)$$

como en los ejemplos anteriores (en varios casos se puede interpretar como la suma de dos energías distintas), Euler simpléctico se vuelve un método explícito:

$$\begin{aligned} p_{n+1} &= p_n - \tau U'(q_n), \\ q_{n+1} &= q_n + \tau T'(p_{n+1}). \end{aligned}$$

En efecto, el primer paso no necesita más la solución de una ecuación implícita en p_{n+1}, y el valor p_{n+1} se inyecta en la segunda para obtener q_{n+1}.

6.3.2. Unos contraejemplos

Veamos ahora que Euler explícito y Euler implícito no son simplécticos. Para un sistema hamiltoniano, Euler explícito se formula así:

$$\begin{aligned} p_{n+1} &= p_n - \tau \partial_q H(p_n, q_n), \\ q_{n+1} &= q_n + \tau \partial_p H(p_n, q_n). \end{aligned}$$

Asumiendo de nuevo H de regularidad C^2, la matriz de flujo discreto que corresponde es la siguiente:

$$\begin{pmatrix} \dfrac{\partial p_{n+1}}{\partial p_n} & \dfrac{\partial p_{n+1}}{\partial q_n} \\ \dfrac{\partial q_{n+1}}{\partial p_n} & \dfrac{\partial q_{n+1}}{\partial q_n} \end{pmatrix} = \begin{pmatrix} 1 - \tau \partial_{pq} H & -\tau \partial_{qq} H \\ \tau \partial_{pp} H & 1 + \tau \partial_{pq} H \end{pmatrix}$$

y el determinante es, entonces,

$$\det \begin{pmatrix} \dfrac{\partial p_{n+1}}{\partial p_n} & \dfrac{\partial p_{n+1}}{\partial q_n} \\ \dfrac{\partial q_{n+1}}{\partial p_n} & \dfrac{\partial q_{n+1}}{\partial q_n} \end{pmatrix} = 1 - \tau^2 \partial_{pq} H + \tau^2 \partial_{pp} H \partial_{qq} H.$$

Típicamente, para el oscilador harmónico, obtenemos

$$\det \begin{pmatrix} \frac{\partial p_{n+1}}{\partial p_n} & \frac{\partial p_{n+1}}{\partial q_n} \\ \frac{\partial q_{n+1}}{\partial p_n} & \frac{\partial q_{n+1}}{\partial q_n} \end{pmatrix} = 1 + \tau^2.$$

El área del flujo discreto aumenta con el tiempo. Veremos que esto se observa claramente en experimentos numéricos.

Ahora veamos lo que podemos decir sobre la conservación del hamiltoniano discreto. Notamos

$$\delta p = p_{n+1} - p_n, \quad \delta q = q_{n+1} - q_n,$$

y con una fórmula de Taylor al orden 2, obtenemos:

$$\begin{aligned} & H(p_{n+1}, q_{n+1}) - H(p_n, q_n) \\ = \; & \underbrace{\partial_q H(p_n, q_n)(q_{n+1} - q_n) + \partial_p H(p_n, q_n)(p_{n+1} - p_n)}_{0} \\ & + \frac{1}{2}\partial_{qq} H(\delta q)^2 + \frac{1}{2}\partial H_{pp}(\delta p)^2 + \partial_{pq} H(\delta p)(\delta q) + \mathcal{O}((\delta p)^3, (\delta q)^3). \end{aligned}$$

Para el oscilador harmónico, con $\omega = 1$, tenemos, por ejemplo:

$$H(p_{n+1}, q_{n+1}) - H(p_n, q_n) = \frac{1}{2}(\delta q)^2 + \frac{1}{2}(\delta p)^2.$$

Utilizando las ecuaciones de Euler explícito

$$\delta p = -\tau q_n, \quad \delta q = \tau p_n,$$

llegamos a

$$H(p_{n+1}, q_{n+1}) - H(p_n, q_n) = \frac{\tau}{2}(p_n^2 + q_n^2).$$

Observamos que el hamiltoniano a nivel discreto no se conserva sino que aumenta en el transcurso del tiempo. Este efecto se puede reducir con τ muy pequeño. Otro punto de vista: este fenómeno se puede interpretar como una creación de energía artificial (numérica) a cada paso de tiempo. Para simulaciones en tiempo corto, este crecimiento del hamiltoniano es controlado por Gronwall (ver el capítulo 3). No obstante, puede ser un problema para simulaciones en grandes intervalos de tiempo, como en astrofísica.

Observación 19. *Cálculos similares muestran que el método de Euler implícito produce efectos contrarios: el determinante del jacobiano del flujo se reduce a cada paso de tiempo y el hamiltoniano decrece a cada paso de tiempo.*

6.3.3. Método de punto medio

El método de punto medio implícito visto en el capítulo 5 es otro ejemplo de un método simpléctico. Para un sistema hamiltoniano, el método se puede definir como:

$$\begin{aligned} p_{n+1} &= p_n - \tau \partial_q H(p_{n+\frac{1}{2}}, q_{n+\frac{1}{2}}), \\ q_{n+1} &= q_n + \tau \partial_p H(p_{n+\frac{1}{2}}, q_{n+\frac{1}{2}}), \end{aligned} \tag{6.6}$$

con $\tau > 0$ el paso de tiempo, y las notaciones:

$$p_{n+\frac{1}{2}} = \frac{p_n + p_{n+1}}{2}, \quad q_{n+\frac{1}{2}} = \frac{q_n + q_{n+1}}{2}.$$

Demostramos ahora que es simpléctico.

Teorema 16. *Supongamos que el hamiltoniano es C^2. El método* (6.6) *verifica, para todo $\tau > 0$ y para todo $0 \leq n < N$:*

$$\det\left(\frac{\partial(p_{n+1}, q_{n+1})}{\partial(p_n, q_n)}\right) = 1, \tag{6.7}$$

donde

$$\frac{\partial(p_{n+1}, q_{n+1})}{\partial(p_n, q_n)}$$

es la jacobiana de la aplicación $(p_n, q_n) \to (p_{n+1}, q_{n+1})$.

Demostración. Derivamos la primera ecuación de (6.6) en p_n:

$$\frac{\partial p_{n+1}}{\partial p_n} = 1 - \frac{\tau}{2}\partial_{pq} H\left(\frac{\partial p_{n+1}}{\partial p_n} + 1\right) - \frac{\tau}{2}\partial_{qq} H \frac{\partial q_{n+1}}{\partial p_n}.$$

De la misma forma, tenemos

$$\frac{\partial p_{n+1}}{\partial q_n} = -\frac{\tau}{2}\partial_{pq} H \frac{\partial p_{n+1}}{\partial q_n} - \frac{\tau}{2}\partial_{qq} H\left(\frac{\partial q_{n+1}}{\partial q_n} + 1\right).$$

Ahora, derivando la segunda ecuación, obtenemos:

$$\frac{\partial q_{n+1}}{\partial p_n} = \frac{\tau}{2}\partial_{pp}(\frac{\partial p_{n+1}}{\partial p_n} + 1) + \frac{\tau}{2}\partial_{qp}\frac{\partial q_{n+1}}{\partial p_n}$$

y

$$\frac{\partial q_{n+1}}{\partial q_n} = 1 + \frac{\tau}{2}\partial_{pp}H\frac{\partial p_{n+1}}{\partial q_n} + \frac{\tau}{2}\partial_{qp}H\left(\frac{\partial q_{n+1}}{\partial q_n} + 1\right).$$

Aplicamos de nuevo el teorema de Schwarz y tenemos, entonces, la siguiente expresión para el flujo discreto:

$$\begin{pmatrix} 1 + \frac{\tau}{2}\partial_{pq}H & \frac{\tau}{2}\partial_{qq}H \\ -\frac{\tau}{2}\partial_{pp}H & 1 - \frac{\tau}{2}\partial_{pq}H \end{pmatrix} \begin{pmatrix} \frac{\partial p_{n+1}}{\partial p_n} & \frac{\partial p_{n+1}}{\partial q_n} \\ \frac{\partial q_{n+1}}{\partial p_n} & \frac{\partial q_{n+1}}{\partial q_n} \end{pmatrix}$$

$$= \begin{pmatrix} 1 - \frac{\tau}{2}\partial_{pq}H & -\frac{\tau}{2}\partial_{qq}H \\ \frac{\tau}{2}\partial_{pp}H & 1 + \frac{\tau}{2}\partial_{pq}H \end{pmatrix}.$$

Dado que las matrices de izquierda y derecha tienen el mismo determinante, obtenemos:

$$\det\left(\frac{\partial(p_{n+1}, q_{n+1})}{\partial(p_n, q_n)}\right) = 1,$$

lo que termina la demostración. □

Ahora veamos lo que podemos decir sobre la conservación del hamiltoniano discreto. Primero hacemos:

$$\begin{aligned} & H(p_{n+1}, q_{n+1}) - H(p_n, q_n) \\ = \; & H(p_{n+1}, q_{n+1}) - H(p_{n+1/2}, q_{n+1/2}) + H(p_{n+1/2}, q_{n+1/2}) - H(p_n, q_n). \end{aligned}$$

Después aplicamos Taylor en el punto $(p_{n+1/2}, q_{n+1/2})$ al orden 2:

$$\begin{aligned} H(p_{n+1}, q_{n+1}) - H(p_n, q_n) &= (p_{n+1} - p_{n+1/2})\partial_p H(p_{n+1/2}, q_{n+1/2}) \\ &\quad + (q_{n+1} - q_{n+1/2})\partial_q H(p_{n+1/2}, q_{n+1/2}) \\ &\quad - (p_n - p_{n+1/2})\partial_p H(p_{n+1/2}, q_{n+1/2}) \\ &\quad - (q_n - q_{n+1/2})\partial_q H(p_{n+1/2}, q_{n+1/2}) \\ &= \delta p\partial_p H + \delta q\partial_q H + \mathcal{O}((\delta p)^3, (\delta q)^3) \\ &= \mathcal{O}((\delta p)^3, (\delta q)^3). \end{aligned}$$

Hemos utilizado el hecho de que los términos de orden dos se anulan, y luego las ecuaciones (6.6) del método. De forma notable, para el oscilador armónico, tenemos una conservación exacta del hamiltoniano (si omitimos los errores de redondeo). Finalmente, nos acordamos de que es un método implícito y de orden 2 (capítulo 5).

6.3.4. Método de Störmer-Verlet

El método de Störmer-Verlet fue concebido, entre otros, por L. Verlet y C. Störmer (ver [36] para un histórico más completo). Su simplicidad y sus propiedades lo hicieron muy popular en la comunidad de cálculo científico, particularmente, de dinámica molecular y de computación gráfica. Las evoluciones recientes en las arquitecturas computacionales (paralelización masiva con *Graphical Processing Units, GPU*) han aumentado su interés en los últimos años (ver, por ejemplo, [13] y las referencias citadas).

Una forma de obtenerlo es componiendo dos métodos de Euler simpléctico [36]. Más precisamente, notamos

$$\phi_\tau : (p_n, q_n) \mapsto (p_{n+1}, q_{n+1})$$

el mapeo asociado a Euler simpléctico en su primera versión (6.3):

$$\begin{aligned} p_{n+1} &= p_n - \tau \partial_q H(p_{n+1}, q_n), \\ q_{n+1} &= q_n + \tau \partial_p H(p_{n+1}, q_n), \end{aligned}$$

y

$$\phi^\star_\tau : (p_n, q_n) \mapsto (p_{n+1}, q_{n+1})$$

el mapeo asociado a la segunda versión

$$\begin{aligned} p_{n+1} &= p_n - \tau \partial_q H(p_n, q_{n+1}), \\ q_{n+1} &= q_n + \tau \partial_p H(p_n, q_{n+1}). \end{aligned}$$

El método de Störmer-Verlet se puede obtener como

$$\Psi_\tau = \phi^\star_{\frac{\tau}{2}} \circ \phi_{\frac{\tau}{2}}.$$

Ahora explicitamos su fórmula:

$$\begin{aligned} p_{n+\frac{1}{2}} &= p_n - \frac{\tau}{2}\partial_q H(p_{n+\frac{1}{2}}, q_n), \\ q_{n+\frac{1}{2}} &= q_n + \frac{\tau}{2}\partial_p H(p_{n+\frac{1}{2}}, q_n), \\ p_{n+1} &= p_{n+\frac{1}{2}} - \frac{\tau}{2}\partial_q H(p_{n+\frac{1}{2}}, q_{n+1}), \\ q_{n+1} &= q_{n+\frac{1}{2}} + \frac{\tau}{2}\partial_p H(p_{n+\frac{1}{2}}, q_{n+1}), \end{aligned}$$

lo que se escribe también como

$$\begin{aligned} p_{n+\frac{1}{2}} &= p_n - \frac{\tau}{2}\partial_q H(p_{n+\frac{1}{2}}, q_n), \\ q_{n+1} &= q_n + \frac{\tau}{2}\partial_p H(p_{n+\frac{1}{2}}, q_n) + \frac{\tau}{2}\partial_p H(p_{n+\frac{1}{2}}, q_{n+1}), \\ p_{n+1} &= p_{n+\frac{1}{2}} - \frac{\tau}{2}\partial_q H(p_{n+\frac{1}{2}}, q_{n+1}). \end{aligned}$$

Otra versión de Störmer-Verlet se obtiene haciendo

$$\Psi_\tau^\star = \phi_{\frac{\tau}{2}} \circ \phi_{\frac{\tau}{2}}^\star.$$

Para un hamiltoniano de la forma $H(p,q) = H(p) + H(q)$, obtenemos un método completamente explícito. Se comprueba, sin problema, que es un método de orden 2, entonces, mucho más preciso que Euler simpléctico [36]. El método de Störmer-Verlet es simpléctico dado que está compuesto por dos metódos simplécticos [36]. Se puede escribir de varias otras formas [13], [36].

6.4. Ejemplo con el oscilador harmónico

Programamos los métodos vistos anteriormente en Python y comparamos los resultados en el caso del oscilador armónico.

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 # Parámetros del oscilador armónico
5 omega = 1.0  # frecuencia angular
```

```
t_start, t_end = 0, 10  # intervalo de tiempo
step_size = 0.1  # tamaño del paso
num_steps = int((t_end - t_start) / step_size)  # número de pasos

# Condiciones iniciales
initial_position = 1.0
initial_momentum = 0.0

# Definición de la función para el oscilador armónico
def harmonic_oscillator(state, t, omega):
    x, p = state
    dxdt = p
    dydt = -omega**2 * x
    return np.array([dxdt, dydt])

# Método de Euler explícito
def explicit_euler(method_func, initial_state, step_size, num_steps, *method_args):
    state = np.copy(initial_state)
    results = np.zeros((num_steps, len(initial_state)))
    for i in range(num_steps):
        results[i] = state
        state += step_size * method_func(state, i * step_size, *method_args)
    return results

# Método de Euler implícito
def implicit_euler(method_func, initial_state, step_size, num_steps, *method_args):
    state = np.copy(initial_state)
    results = np.zeros((num_steps, len(initial_state)))
    for i in range(num_steps):
        results[i] = state
        x_new = state[0] + step_size * (state[1] - step_size * omega**2 * (state[0] + step_size * state[1]))
        p_new = state[1] - step_size * omega**2 * (state[0] + step_size * state[1])
        state = np.array([x_new, p_new])
    return results

# Método de Euler simpléctico
def symplectic_euler(method_func, initial_state, step_size
```

```
    , num_steps, *method_args):
43      state = np.copy(initial_state)
44      results = np.zeros((num_steps, len(initial_state)))
45      for i in range(num_steps):
46          results[i] = state
47          x_new = state[0] + step_size * state[1]
48          p_new = state[1] - step_size * omega**2 * x_new
49          state = np.array([x_new, p_new])
50      return results
51
52  # Método del punto medio
53  def midpoint_method(method_func, initial_state, step_size,
        num_steps, *method_args):
54      state = np.copy(initial_state)
55      results = np.zeros((num_steps, len(initial_state)))
56      for i in range(num_steps):
57          results[i] = state
58          k1 = method_func(state, i * step_size, *
                method_args)
59          k2 = method_func(state + 0.5 * step_size * k1, i *
                 step_size + 0.5 * step_size, *method_args)
60          state += step_size * k2
61      return results
62
63  # Resolver usando los cuatro métodos
64  initial_state = np.array([initial_position,
        initial_momentum])
65  explicit_results = explicit_euler(harmonic_oscillator,
        initial_state, step_size, num_steps, omega)
66  implicit_results = implicit_euler(harmonic_oscillator,
        initial_state, step_size, num_steps, omega)
67  symplectic_results = symplectic_euler(harmonic_oscillator,
         initial_state, step_size, num_steps, omega)
68  midpoint_results = midpoint_method(harmonic_oscillator,
        initial_state, step_size, num_steps, omega)
69
70  # Calcular el hamiltoniano para cada método
71  def hamiltonian(x, p, omega):
72      return 0.5 * p**2 + 0.5 * omega**2 * x**2
73
74  h_explicit = np.array([hamiltonian(explicit_results[i, 0],
         explicit_results[i, 1], omega) for i in range(
        num_steps)])
75  h_implicit = np.array([hamiltonian(implicit_results[i, 0],
```

```
    implicit_results[i, 1], omega) for i in range(
    num_steps)])
h_symplectic = np.array([hamiltonian(symplectic_results[i,
    0], symplectic_results[i, 1], omega) for i in range(
    num_steps)])
h_midpoint = np.array([hamiltonian(midpoint_results[i, 0],
    midpoint_results[i, 1], omega) for i in range(
    num_steps)])

# Graficar los resultados en el plano de fase
plt.figure(figsize=(14, 6))

plt.subplot(1, 2, 1)
plt.plot(explicit_results[:, 0], explicit_results[:, 1],
    label='Euler explícito')
plt.plot(implicit_results[:, 0], implicit_results[:, 1],
    label='Euler implícito')
plt.plot(symplectic_results[:, 0], symplectic_results[:,
    1], label='Euler simpléctico')
plt.plot(midpoint_results[:, 0], midpoint_results[:, 1],
    label='punto medio')
plt.title('Plano de fase')
plt.xlabel('posición')
plt.ylabel('momento')
plt.legend()
plt.grid()

# Graficar la evolución del hamiltoniano
plt.subplot(1, 2, 2)
plt.plot(np.linspace(t_start, t_end, num_steps),
    h_explicit, label='Euler explícito')
plt.plot(np.linspace(t_start, t_end, num_steps),
    h_implicit, label='Euler implícito')
plt.plot(np.linspace(t_start, t_end, num_steps),
    h_symplectic, label='Euler simpléctico')
plt.plot(np.linspace(t_start, t_end, num_steps),
    h_midpoint, label='punto medio')
plt.title('Evolución del hamiltoniano')
plt.xlabel('tiempo')
plt.ylabel('hamiltoniano')
plt.legend()
plt.grid()

plt.tight_layout()
```

```
106 plt.show()
```

Listado 8: Implementación en Python para solucionar el oscilador armónico.

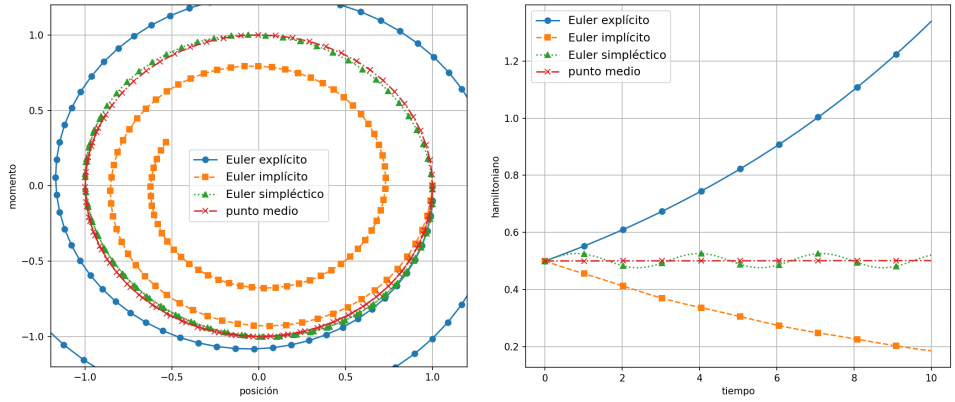


Figura 5: Aproximación del oscilador armónico con diferentes métodos. Órbitas en el plano de fase y hamiltoniano en función del tiempo.

Los resultados se pueden observar en la figura 5. Las conclusiones son conformes a la teoría:

- El método del punto medio implícito conserva perfectamente la trayectoria en el plano de fase así como el hamiltoniano.
- El método de Euler simpléctico conserva casi perfectamente la trayectoria en el plano de fase (se nota una leve diferencia). No conserva exactamente el hamiltoniano y se notan oscilaciones alrededor del valor exacto. Esas oscilaciones se atenúan cuando disminuimos el paso de tiempo τ.
- El método de Euler explícito calcula una trayectoria divergente, que se aleja al infinito, y el hamiltoniano es una función creciente del tiempo.
- El método de Euler implícito calcula una trayectoria divergente, que se acerca al punto $(0, 0)$, y el hamiltoniano disminuye en el transcurso del tiempo.

6.5. Ejemplo con Lotka-Volterra

Tomamos el ejemplo inicial de [36] y estudiamos cómo se comporta Euler simpléctico en la práctica para las ecuaciones de Lotka-Volterra. Las ecuaciones de Lotka-Volterra, también conocidas como el modelo depredador-presa, son un par de ecuaciones diferenciales no lineales de primer orden que se utilizan frecuentemente para describir la dinámica de sistemas biológicos en los que dos especies interactúan, una como depredador y la otra como presa. El modelo está dado por las siguientes ecuaciones:

$$\frac{dx}{dt} = \alpha x - \beta xy,$$
$$\frac{dy}{dt} = \delta xy - \gamma y,$$

donde x representa el número de presas, y el número de depredadores, α, β, δ, y γ son parámetros que representan las tasas de interacción y crecimiento de las dos poblaciones. Este modelo simple ilustra cómo la coexistencia de dos especies puede surgir de la interacción entre ellas[7].

El código Python que corresponde es:

```
import numpy as np
import matplotlib.pyplot as plt
def lotka_volterra(state, t, alpha, beta, delta, gamma):
    x, y = state
    dxdt = alpha * x - beta * x * y
    dydt = delta * x * y - gamma * y
    return np.array([dxdt, dydt])

# Parámetros
alpha, beta, delta, gamma = 0.1, 0.02, 0.01, 0.3
initial_state = np.array([40.0, 9.0])
t_start, t_end = 0, 200
step_size = 0.1
num_steps = int((t_end - t_start) / step_size)

# Euler explícito
```

[7] Ver, por ejemplo, [10].

```
17 def explicit_euler(method_func, initial_state, step_size,
       num_steps, *method_args):
18     state = np.copy(initial_state)
19     results = np.zeros((num_steps, len(initial_state)),
           dtype=float)
20     for i in range(num_steps):
21         results[i] = state
22         state += step_size * method_func(state, i *
               step_size, *method_args)
23     return results
24
25 # Euler simpléctico
26 def symplectic_euler(method_func, initial_state, step_size
       , num_steps, *method_args):
27     state = np.copy(initial_state)
28     results = np.zeros((num_steps, len(initial_state)),
           dtype=float)
29     for i in range(num_steps):
30         results[i] = state
31         k1 = method_func(state, i * step_size, *
               method_args)
32         state += step_size * np.array([k1[0], method_func(
               state + step_size * np.array([k1[0], 0.0]), i *
                step_size, *method_args)[1]])
33     return results
34
35 explicit_results = explicit_euler(lotka_volterra,
       initial_state, step_size, num_steps, alpha, beta, delta
       , gamma)
36 symplectic_results = symplectic_euler(lotka_volterra,
       initial_state, step_size, num_steps, alpha, beta, delta
       , gamma)
37
38 plt.figure(figsize=(12, 6))
39
40 plt.subplot(1, 2, 1)
41 plt.plot(explicit_results[:, 0], explicit_results[:, 1],
       label='explícito', color='b')
42 plt.grid()
43
44 plt.subplot(1, 2, 2)
45 plt.plot(symplectic_results[:, 0], symplectic_results[:,
       1], label='simpléctico', color='r')
46 plt.grid()
```

```
47
48 plt.tight_layout()
49 plt.show()
```

Listado 9: Implementación en Python: Euler para Lotka-Volterra.

La figura 6 muestra el resultado numérico.

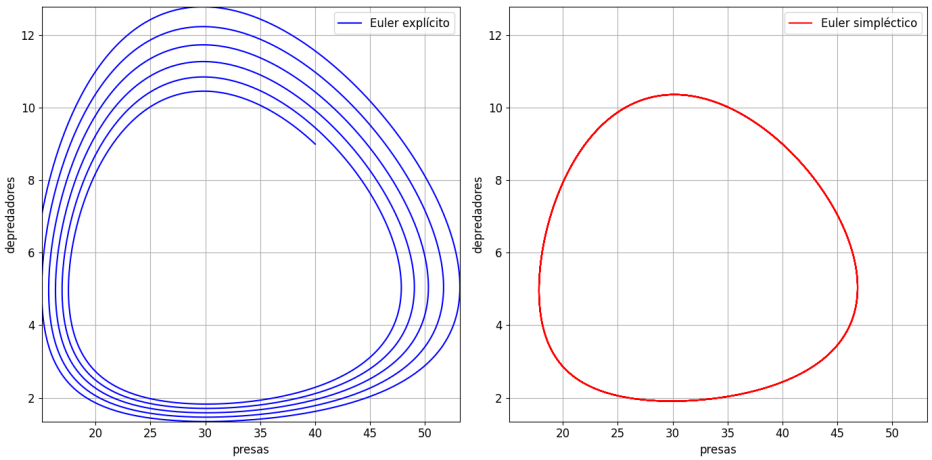


Figura 6: Lotka-Volterra con Euler explícito y Euler simpléctico.

El método Euler simpléctico actualiza las variables de estado de una manera que respeta la geometría simpléctica del sistema. Las reglas de actualización para el método Euler simpléctico aplicado a las ecuaciones de Lotka-Volterra son:

1. Actualizar la población de presas x usando la población actualizada de depredadores y:

$$x_{n+1} = x_n + \tau(\alpha x_n - \beta x_n y_n)$$

2. Actualizar la población de depredadores y usando la población actualizada de presas x_{n+1}:

$$y_{n+1} = y_n + \tau(\delta x_{n+1} y_n - \gamma y_n)$$

Aquí, τ es el tamaño del paso de tiempo y x_n, y_n son las poblaciones en el paso de tiempo n. La idea clave es que la actualización de x se realiza primero, y luego la actualización de y utiliza la x_{n+1} recién calculada. Esta actualización secuencial ayuda a preservar la estructura simpléctica del sistema, lo que puede llevar a simulaciones más precisas a largo plazo en comparación con métodos no simplécticos como el método Euler explícito.

Observación 20. *El hamiltoniano asociado a las ecuaciones de Lotka-Volterra es*

$$H(p, q) = \exp(p) - p + \exp(q) - q,$$

con $p = \ln(x)$ *y* $q = \ln(y)$.

Capítulo 7
Conclusión y perspectivas

Para concluir nuestra breve introducción al tema de la aproximación numérica de ecuaciones diferenciales ordinarias, damos unas sugerencias que permitirán al lector ir más allá en el tema y/o descubrir otros campos relacionados.

Primero, para tener una visión más general y más completa sobre el tema se puede consultar [37], [39]. Particularmente, allí se detallan los resultados generales obtenidos por Dahlquist relacionados a los límites de los métodos multipasos (ver también [60] a un nivel más introductorio). En segundo lugar, hubo mucho progreso recientemente en el tema de métodos paralelos en tiempo; para ello, ver el libro [30] y sus referencias asociadas. Otro tema aún abierto es la integración numérica geométrica. El libro [36] presenta una base sobre el tema (ver también [25] para una extensión a algunas ecuaciones en derivadas parciales de la física cuántica). Mencionamos, también, la aplicación de técnicas numéricas para asuntos en sistemas dinámicos; para ello, ver, por ejemplo, [59].

Además, las técnicas vistas en este libro son una base para aproximar ecuaciones en derivadas parciales con componente en tiempo, como la ecuación del calor o de onda. De hecho, una técnica clásica para solucionar estas ecuaciones consiste en aplicar primero una técnica de discretización espacial, como las diferencias finitas, los volúmenes finitos o los elementos finitos; esto permite tratar las variables espaciales. Luego, se obtiene un sistema de ecuaciones diferenciales ordinarias y se puede, entonces, aplicar uno de los métodos vistos en este libro (ver, por ejemplo, [20], [21], [22] en el caso de los elementos finitos).

Referencias bibliográficas

[1] V. I. Arnol'd, *Ordinary differential equations*. Moskva: Nauka. Glavnaya Redaktsiya Fiziko-Matematicheskoj Literatury. 272 p. R. 0.90, 1984.

[2] ——, *Mathematical methods of classical mechanics*, Springer, 1989.

[3] M. Astorino, F. Chouly, and A. Quarteroni, *A time-parallel framework for coupling finite element and lattice Boltzmann methods*, AMRX, Appl. Math. Res. Express, 2016 (2016), pp. 24–67.

[4] P. Azerad, *Analyse numérique des équations différentielles*, 2023. Curso, año universitario 2022–2023. Facultad de Ciencias, Universidad de Montpellier. Disponible en: https://imag.umontpellier.fr/~azerad/coursequadiff.pdf.

[5] A. Bonito, C. Canuto, R. H. Nochetto, and A. Veeser, *Adaptive finite element methods*, Acta Numerica, 33 (2024), pp. 163–485.

[6] J. C. Butcher, *Numerical methods for ordinary differential equations*, John Wiley & Sons, 2016.

[7] J. C. Butcher, *B-series. Algebraic analysis of numerical methods*, Springer, 2021.

[8] C. Carstensen, M. Feischl, M. Page, and D. Praetorius, *Axioms of adaptivity*, Comput. Math. Appl., 67 (2014), pp. 1195–1253.

[9] P. Chartier and B. Philippe, *A parallel shooting technique for solving dissipative ODE's*, Computing, 51 (1993), pp. 209–236.

[10] J. R. Chasnov, *Mathematical Biology*, LibreTexts, 2024. Disponible en: https://math.libretexts.org/Bookshelves/Applied_Mathematics/Mathematical_Biology_(Chasnov).

[11] F. Chouly, *An introductive course to some numerical approximation methods for ordinary and partial differential equations*. Apuntes de clase para la maestría "Math4Phy". Instituto de Matemáticas de Borgoña, Francia. Disponible en: https://cel.hal.science/hal-03212748, 2021.

[12] ——, *Jupyter Notebooks for numerical approximation of ordinary differential equations*, 11 2025. DOI:10.6084/m9.figshare.30687245.v1.

[13] F. Chouly and Y. Renard, *Explicit Verlet time-integration for a Nitsche-based approximation of elastodynamic contact problems*, Adv. Model. Simul. Engng. Sci., 5 (2018), pp. 1–31.

[14] P. G. Ciarlet, *Linear and nonlinear functional analysis with applications*, SIAM, 2025.

[15] E. A. Coddington and N. Levinson, *Theory of ordinary differential equations*. McGill-Hill, 1955.

[16] M. Crouzeix and A. L. Mignot, *Analyse numérique des équations différentielles*, Masson, 1989.

[17] J. P. Demailly, *Analyse numérique et équations différentielles*, EDP Sciences, 2016.

[18] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson, *Introduction to adaptive methods for differential equations*, in Acta Numerica, Cambridge University Press, 1995, pp. 105–158.

[19] ——, *Computational differential equations*, Cambridge University Press, 1996.

[20] A. Ern and J. L. Guermond, *Finite elements I. Approximation and interpolation*, Springer, 2020.

[21] ——, *Finite elements II. Galerkin approximation, elliptic and mixed PDEs*, Springer, 2021.

[22] ——, *Finite elements III. First-order and time-dependent PDEs*, Springer, 2021.

[23] D. Estep, *A posteriori error bounds and global error control for approximation of ordinary differential equations*, SIAM J. Numer. Anal., 32 (1995), pp. 1–48.

[24] D. Estep and D. French, *Global error control for the continuous Galerkin finite element method for ordinary differential equations*, RAIRO, Modélisation Math. Anal. Numér., 28 (1994), pp. 815–852.

[25] E. Faou, *Geometric numerical integration and Schrödinger equations*, European Mathematical Society (EMS), 2012.

[26] C. Farhat and M. Chandesris, *Time-decomposed parallel time-integrators: Theory and feasibility studies for fluid, structure, and fluid-structure applications*, Int. J. Numer. Methods Eng., 58 (2003), pp. 1397–1434.

[27] M. Feischl and D. Niederkofler, *Optimal adaptive implicit time stepping*, Advances in Applied Mechanics, 61 (2025), pp. 407–448.

[28] A. Fortin, *Analyse numérique pour ingénieurs*, Presses inter Polytechnique, 2008.

[29] M. J. Gander, *50 years of time parallel time integration*, in "Multiple shooting and time domain decomposition methods", Springer, 2015, pp. 69–113.

[30] M. J. Gander and T. Lunet, *Time parallel time integration*, SIAM, 2024.

[31] M. J. Gander and S. Vandewalle, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput., 29 (2007), pp. 556–578.

[32] M. J. Gander and G. Wanner, *From Euler, Ritz, and Galerkin to modern computing*, SIAM Rev., 54 (2012), pp. 627–666.

[33] W. Gander, M. J. Gander, and F. Kwok, *Scientific computing. An introduction using Maple and MATLAB*, Springer, 2014.

[34] W. Gautschi, *Numerical analysis. An introduction*, Birkhäuser, 1997.

[35] M. Girotti, *The Van der Pol oscillator*, 2024. Course MATH 3406: Differential Equations I, Saint Mary's University, Canada. Disponible en: https://mathemanu.github.io/VanderPol.pdf. Jupyter Notebook disponible en: https://github.com/mathemanu/MATH3406-Diff.Eq.2.

[36] E. Hairer, C. Lubich, and G. Wanner, *Geometric numerical integration. Structure-preserving algorithms for ordinary differential equations*, Springer, 2006.

[37] E. Hairer, S. P. Nørsett, and G. Wanner, *Solving ordinary differential equations. I: Nonstiff problems*, Springer, 2010.

[38] E. Hairer and G. Wanner, *Analysis by its history*, Springer, 2008.

[39] ——, *Solving ordinary differential equations. II: Stiff and differential-algebraic problems*, Springer, 2010.

[40] P. Henrici, *Discrete variable methods in ordinary differential equations.* John Wiley and Sons, 1962.

[41] K. Heun, *Neue Methode zur approximativen Integration der Differentialgleichungen einer unabhängigen Variable*, Schlömilch Z., 45 (1900), pp. 23–38.

[42] M. W. Hirsch and S. Smale, *Differential equations, dynamical systems, and linear algebra*, Academic Press, 1974.

[43] M. W. Hirsch, S. Smale, and R. L. Devaney, *Differential equations, dynamical systems, and an introduction to chaos*, Academic Press, 2013.

[44] J. H. Hubbard and F. Hubert, *Calcul scientifique. Volume 2. Équations différentielles et équations aux dérivées partielles*, Vuibert, 2006.

[45] IEEE, *IEEE standard for floating-point arithmetic*, Tech. Rep. 754-2019, Institute of Electrical and Electronics Engineers, 2019. Disponible en: https://ieeexplore.ieee.org/document/8766229.

[46] A. Iserles, *A first course in the numerical analysis of differential equations*, Cambridge University Press, 2009.

[47] B. Kehlet and A. Logg, *A posteriori error analysis of round-off errors in the numerical solution of ordinary differential equations*, Numer. Algorithms, 76 (2017), pp. 191–210.

[48] W. Kutta, *Beitrag zur näherungsweisen Integration totaler Differentialgleichungen*, Schlömilch Z., 46 (1901), pp. 435–452.

[49] J. D. Lambert, *Numerical methods for ordinary differential systems: the initial value problem*, John Wiley & Sons, 1991.

[50] J. L. Lions, Y. Maday, and G. Turinici, *A "parareal" in time discretization of PDE's*, C. R. Acad. Sci., Paris, Sér. I, Math., 332 (2001), pp. 661–668.

[51] J. Nievergelt, *Parallel methods for integrating ordinary differential equations*, Commun. ACM, 7 (1964), pp. 731–733.

[52] J. T. Oden and L. F. Demkowicz, *Applied functional analysis*, CRC Press, 2018.

[53] J. Rappaz and M. Picasso, *Introduction à l'analyse numérique*, Presses Polytechniques et Universitaires Romandes, 1998.

[54] C. Runge, *Ueber die numerische Auflösung von Differentialgleichungen*, Math. Ann., 46 (1895), pp. 167–178.

[55] P. Saha, J. Stadel, and S. Tremaine, *A parallel integration method for solar system dynamics*, The Astronomical Journal, 114 (1997), pp. 409–415.

[56] M. Schatzman, *Numerical analysis. A mathematical introduction*, Clarendon Press, 2002.

[57] D. Serre, *Matrices*, Springer, 2010.

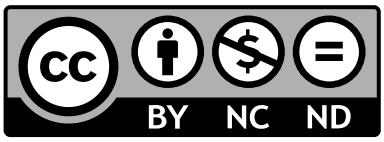
[58] H. J. Stetter, *The defect correction principle and discretization methods*, Numer. Math., 29 (1978), pp. 425–443.

[59] A. M. Stuart and A. R. Humphries, *Dynamical systems and numerical analysis*, Cambridge University Press, 1996.

[60] E. Süli and D. F. Mayers, *An introduction to numerical analysis*, Cambridge University Press, 2003.

[61] K. Yosida, *Functional analysis*, Springer, 1965.

Índice temático



Esta obra está bajo una Licencia Creative Commons Atribución-NoComercial-SinDerivadas 4.0 Internacional.

Este libro surge de un curso impartido en las licenciaturas de Matemática y Física de la Facultad de Ciencias de la Universidad de la República, con el objetivo de introducir los conceptos esenciales para la aproximación numérica de ecuaciones diferenciales ordinarias. A través de un enfoque riguroso pero accesible, se presentan algoritmos que permiten obtener secuencias numéricas aproximadas a las soluciones exactas, analizando su precisión y estabilidad. El análisis numérico, como disciplina central, garantiza la fiabilidad de estos métodos o identifica sus limitaciones.

Dirigido a estudiantes avanzados, investigadores y profesionales en matemática aplicada, física e ingeniería, este texto no solo aborda ecuaciones diferenciales ordinarias, sino que también sienta las bases para extender estos métodos a ecuaciones en derivadas parciales. El libro combina teoría con implementaciones prácticas en Python disponibles para experimentación.

Una obra introductoria pero rigurosa, ideal para quienes buscan dominar los fundamentos del análisis numérico y su aplicación en la resolución de problemas científicos.

DIRAC

9 789974 023741

RF[DS]