Universidad de Valladolid

Facultad de Ciencias

TRABAJO FIN DE GRADO

GRADO EN MATEMÁTICAS

Estudio de una familia de métodos Runge-Kutta implícitos: Los Métodos de Gauss-Legendre

Autor: Manuel Jiménez del Río

Tutor: María Paz Calvo Cabrero

Índice general

Introducción

En el Trabajo Fin de Grado que presentamos se aborda el estudio de los métodos de Gauss-Legendre para la integración numérica de sistemas de ecuaciones diferenciales ordinarias.

En el Capítulo 1 se revisan algunos resultados generales sobre métodos Runge-Kutta implícitos, que servirán como base para demostrar las demás propiedades que se abordan en esta memoria.

El Capítulo 2 incluye diversos resultados teóricos que permiten concluir que los métodos de Gauss tienen orden máximo. Se estudian, entre otros, las hipótesis simplificadoras para las condiciones de orden, y se interpretan los métodos de Gauss como métodos de colocación en los nodos asociados a los polinomios de Legendre.

En el Capítulo 3 se estudian las diferentes propiedades de estabilidad de los métodos de Gauss.

El Capítulo 4 considera, en primer lugar, la implementación eficiente de los métodos Runge-Kutta implícitos prestando especial atención a la resolución de las ecuaciones no lineales que se generan. A continuación se presentan resultados numéricos obtenidos al integrar el problema de Kepler con los métodos de Gauss de 2 y 4 etapas, y se analiza la eficiencia de ambos métodos.

En el Apéndice A se incluye la demostración de algunas propiedades de los polinomios de Legendre y en el Apéndice B se recogen los programas en Matlab que se han utilizado durante la experimentación numérica.

Por último, me gustaría agradecer a la Dra. María Paz Calvo Cabrero por acogerme bajo su tutela cuando le pedí que fuese mi tutora para este Trabajo Fin de Grado, por todo el tiempo y apoyo que me ha brindado de ese momento en adelante, y por darme la oportunidad de seguir aprendiendo junto a ella todo este tiempo.

Capítulo 1

Nociones básicas sobre métodos Runge-Kutta

1.1. Definición

Se considera un problema de valores iniciales

$$\left\{ \begin{array}{rcl} \mathbf{x}'(t) & = & \mathbf{f}(t, \mathbf{x}(t)), \quad t_0 \leq t \leq T, \\ \mathbf{x}(t_0) & = & \mathbf{x}_0, \end{array} \right. \tag{1.1}$$

con $\mathbf{x}_0 \in \mathbb{R}^D$ y $\mathbf{f} : \mathbb{R} \times \mathbb{R}^D \longrightarrow \mathbb{R}^D$ suficientemente regular.

Dada una aproximación $\mathbf{x}_n$ a la solución de (1.1) en tiempo $t = t_n$, se puede calcular una aproximación $\mathbf{x}_{n+1}$ a la solución de (1.1) en $t_{n+1} = t_n + h$ aplicando un método Runge-Kutta de s etapas con coeficientes

$$\mathbf{b} = (b_1, ..., b_s)^T, \quad \mathcal{A} = (a_{ij})_{1 \leq i,j \leq s}, \quad \mathbf{c} = (c_1, \ldots, c_s)^T, \tag{1.2}$$

del siguiente modo:

$$\mathbf{k}_i = \mathbf{f}(t_n + c_i h, \mathbf{x}_n + h \sum_{j=1}^{s} a_{ij} \mathbf{k}_j), \quad 1 \leq i \leq s, \tag{1.3}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h \sum_{i=1}^{s} b_i \mathbf{k}_i, \tag{1.4}$$

Una forma alternativa de definir estos métodos es la siguiente

$$\mathbf{X}_i = \mathbf{x}_n + h \sum_{j=1}^{s} a_{ij} \mathbf{f}(t_n + c_j h, \mathbf{X}_j), \quad 1 \leq i \leq s, \tag{1.5}$$

$$\mathbf{x}_{n+1} = \mathbf{x}_n + h \sum_{i=1}^{s} b_i \mathbf{f}(t_n + c_i h, \mathbf{X}_i), \tag{1.6}$$

donde los vectores $\mathbf{X}_i$ son aproximaciones a la solución de (1.1) en los tiempos $t_n + c_i h$, $1 \leq i \leq s$, que se denominan *etapas intermedias*.

Los coeficientes que definen el método se pueden representar en el llamado tablero de Butcher

$$\begin{array}{c|c} \mathbf{c} & \mathcal{A} \\ \hline & \mathbf{b}^T \end{array}.$$

Si los coeficientes c_i satisfacen $c_i = \sum_{j=1}^{s} a_{ij}$, $1 \leq i \leq s$, es posible reducir el estudio a problemas de valores iniciales en los que $\mathbf{f}$ solo depende de su segundo argumento, es decir, problemas de la forma

$$\left\{ \begin{array}{rcll} \mathbf{x}'(t) & = & \mathbf{f}(\mathbf{x}(t)), & t_0 \leq t \leq T, \\ \mathbf{x}(t_0) & = & \mathbf{x}_0. & \end{array} \right.$$

La solución numérica que encontramos con un método Runge-Kutta puede acercarse en mayor o menor medida a la solución exacta, y uno de los factores que mide la calidad de la aproximación calculada es el orden del método. Pero antes de revisar la teoría del orden para métodos Runge-Kutta, veamos si las ecuaciones (1.3)-(1.4) tienen solución.

1.2. Existencia de solución numérica

Según el formato de la matriz $\mathcal{A}$, distinguimos dos grandes tipos de métodos. Si se cumple $a_{ij} = 0$ para $i \leq j$, esto es, $\mathcal{A}$ es estrictamente triangular inferior, se dice que el método asociado es *explícito*. Por contra, si $\mathcal{A}$ no está sujeta a ninguna restricción, el método se dice *implícito*.

En los métodos explícitos, $\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_n)$ y para $i = 2, \ldots, s$, $\mathbf{k}_i$ se calcula en términos de $\mathbf{k}_1, \ldots, \mathbf{k}_{i-1}$. En los métodos implícitos, cada $\mathbf{k}_i$ involucra a todos los demás y nos encontramos con un sistema de $s \times D$ ecuaciones no lineales que definen a los vectores $\mathbf{k}_1, \ldots, \mathbf{k}_s$ de forma implícita. Es natural, entonces, preguntarse si existe una solución para (1.3).

Teorema 1.1 *Sea $\mathbf{f} : \mathbb{R} \times \mathbb{R}^D \longrightarrow \mathbb{R}^D$ una función continua y lipschitziana respecto de su segundo argumento, $\mathbf{x}$, con constante L. Si*

$$h < \frac{1}{L \cdot \max\limits_{1 \leq i \leq s} \sum\limits_{j=1}^{s} |a_{ij}|}, \tag{1.7}$$

existe una solución única de (1.3), que se puede obtener mediante iteración de punto fijo. Además, si $\mathbf{f}(t, \mathbf{x})$ es p veces diferenciable con continuidad, las $\mathbf{k}_i$ (como funciones de h) son también p veces diferenciables.

Demostración. Lo probaremos, sin pérdida de generalidad, para el primer paso, planteando la iteración de punto fijo

$$\mathbf{k}_i^{[m+1]} = \mathbf{f}(t_0 + c_i h, \mathbf{x}_0 + h \sum_{j=1}^{s} a_{ij} \mathbf{k}_j^{[m]}), \qquad 1 \leq i \leq s, \tag{1.8}$$

para $m = 0, 1, 2, \ldots$, tomando $\mathbf{k}_i^{[0]} = 0$, $1 \leq i \leq s$. Consideramos $\mathbf{K} = (\mathbf{k}_1^T, ..., \mathbf{k}_s^T)^T \in \mathbb{R}^{sD}$ y la norma $||\mathbf{K}|| = \max\limits_{1\leq i\leq s} ||\mathbf{k}_i||$. Reescribimos (1.8) como la iteración de punto fijo $\mathbf{K}^{[m+1]} = \mathbf{F}(\mathbf{K}^{[m]})$, con $\mathbf{K}^{[0]}$ dado, donde la columna i-ésima de $\mathbf{F}(\mathbf{K}^{[m]})$ viene dada por el lado derecho de (1.8).

Utilizando la condición de Lipschitz y la desigualdad triangular, se tiene que

$$\begin{aligned}
||\mathbf{K}^{[m+1]} - \mathbf{K}^{[m]}|| &= ||\mathbf{F}(\mathbf{K}^{[m]}) - \mathbf{F}(\mathbf{K}^{[m-1]})|| = \max_{1\leq i\leq s} ||\mathbf{F}_i(\mathbf{K}^{[m]}) - \mathbf{F}_i(\mathbf{K}^{[m-1]})|| \\
&\leq L \cdot \max_{1\leq i\leq s} ||(\mathbf{x}_0 + h\sum_{j=1}^{s} a_{ij}\mathbf{k}_j^{[m]}) - (\mathbf{x}_0 + h\sum_{j=1}^{s} a_{ij}\mathbf{k}_j^{[m-1]})|| \\
&= L \cdot \max_{1\leq i\leq s} ||h\sum_{j=1}^{s} a_{ij}(\mathbf{k}_j^{[m]} - \mathbf{k}_j^{[m-1]})|| \\
&\leq L \cdot h \left(\max_{1\leq i\leq s} \sum_{j=1}^{s} |a_{ij}| \right) \cdot ||\mathbf{K}^{[m]} - \mathbf{K}^{[m-1]}||.
\end{aligned}$$

Si se satisface (1.7), se concluye $||\mathbf{K}^{[m+1]} - \mathbf{K}^{[m]}|| < ||\mathbf{K}^{[m]} - \mathbf{K}^{[m-1]}||$, o equivalentemente, $||\mathbf{K}^{[m+1]} - \mathbf{K}^{[m]}|| \leq r \cdot ||\mathbf{K}^{[m]} - \mathbf{K}^{[m-1]}||$, con $0 < r < 1$. Aplicando esta desigualdad sucesivamente, llegamos a

$$||\mathbf{K}^{[m+1]} - \mathbf{K}^{[m]}| \leq r \cdot ||\mathbf{K}^{[m]} - \mathbf{K}^{[m-1]}|| \leq \cdots \leq r^m \cdot ||\mathbf{K}^{[1]} - \mathbf{K}^{[0]}|| \xrightarrow[m\to\infty]{} 0.$$

Por lo tanto, la iteración de punto fijo converge, y el límite es la solución del sistema de ecuaciones no lineales (1.3).

El resultado de diferenciabilidad se deduce del teorema de la función implícita (ver [2]):

Escribimos (1.3) como $\Phi(h, \mathbf{K}) = \mathbf{K}(h) - \mathbf{F}(\mathbf{K}(h)) = 0$. En $h = 0$, $\Phi(0, \mathbf{K}(0)) = \mathbf{0}$, puesto que $\mathbf{K}(0) = [\mathbf{f}(t_0, \mathbf{x}_0), \ldots, \mathbf{f}(t_0, \mathbf{x}_0)]$, que claramente satisface (1.3) con $h = 0$. Por otro lado, la matriz $\frac{\partial \Phi}{\partial \mathbf{K}}$ en $h = 0$ es la identidad, por lo que en un entorno de $h = 0$ existe una única función $\mathbf{K}(h)$ con $\Phi(h, \mathbf{K}(h)) = 0$ (que es solución de (1.3)). Además, como $\mathbf{f}$ es p veces diferenciable, también lo es Φ y el teorema de la función implícita garantiza que también lo es $\mathbf{K}(h)$, y, en consecuencia, $\mathbf{k}_i(h)$, $1 \leq i \leq s$. □

1.3. Teoría del orden. Árboles con raíz

En esta sección vamos a definir el orden de un método Runge-Kutta y a revisar brevemente la construcción de las llamadas *condiciones de orden.*

Definición 1.1 *Un método Runge-Kutta tiene orden p si, para problemas (1.1) con $\mathbf{f}$ suficientemente regular, se tiene*

$$||\mathbf{x}(t_0 + h) - \mathbf{x}_1|| \leq K \cdot h^{p+1},$$

donde $\mathbf{x}_1$ es la solución numérica calculada mediante (1.3)-(1.4) con $n = 0$. Esto quiere decir que los desarrollos de Taylor de $\mathbf{x}(t_0 + h)$ y de $\mathbf{x}_1$ coinciden hasta los términos h^p incluidos.

Durante gran parte de esta memoria vamos a trabajar con el orden de los métodos Runge-Kutta, así que nos conviene manejar dicho concepto con soltura.

Ejemplo 1 *Vamos a escribir el desarrollo de Taylor de la aproximación* $\mathbf{x}(t_0+h)$ *como primer paso para encontrar las condiciones de orden que deben satisfacer los coeficientes del método.*

$$\begin{aligned} x^I(t_0+h) &= x_0^I + hf^I + \frac{h^2}{2}\left[\frac{\partial f^I}{\partial x^1}\cdot f^1 + \cdots + \frac{\partial f^I}{\partial x^D}\cdot f^D\right] \\ &\quad + \frac{h^3}{6}\left[\sum_{J=1}^{D}\sum_{K=1}^{D}\frac{\partial^2 f^I}{\partial x^J \partial x^K} f^J f^K + \sum_{J=1}^{D}\frac{\partial f^I}{\partial x^J}\sum_{K=1}^{D}\frac{\partial f^J}{\partial x^K} f^K\right] + \cdots \end{aligned}$$

que, en notación tensorial, obviando las evaluaciones, se puede escribir como:

$$x^I(t_0+h) = x_0^I + hf^I + f_J^I f^J + (f_{JK}^I f^J f^K + f_J^I f_K^J f^K) + \cdots,$$

donde cada subíndice se refiere a una derivada parcial, cada superíndice denota una componente, y debemos entender un sumatorio para cada índice que se repite. Solo con llegar a orden 3 ya se observa que los distintos sumandos que aparecen se van complicando progresivamente. Esto motiva la introducción de los árboles con raíz, como herramienta simplificadora para nuestros desarrollos.

Aunque formalmente los árboles con raíz se definen como clases de equivalencia dentro del conjunto de árboles etiquetados con raíz (ver [3]), para estudiar las condiciones de orden de los métodos Runge-Kutta y los resultados que se presentarán más adelante en esta memoria es suficiente considerar la representación gráfica de dichos árboles y definir recursivamente una serie de funciones asociadas a ellos, que es la tarea que desarrollamos a continuación.

Definición 1.2 *Un árbol con raíz de orden q es un grafo conexo y sin ciclos con q vértices dispuestos en distintos niveles de altura, y un cierto número de lados (las ramas) conectando cada vértice (los hijos) con un único vértice (el padre) del nivel inmediatamente anterior, de modo que hay un único vértice en el nivel inferior (la raíz).*

Denotaremos por $\boldsymbol{\tau}$ al árbol formado por un solo vértice; esto es, la raíz. Además, expresaremos mediante $[\boldsymbol{t}_1,\boldsymbol{t}_2,\ldots,\boldsymbol{t}_m]$ el árbol resultante de unir las raíces de $\boldsymbol{t}_1$ hasta $\boldsymbol{t}_m$ a una raíz común. De esta manera, recursivamente, definiremos varias funciones ligadas a los árboles que nos resultarán útiles para representar las condiciones de orden de los métodos Runge-Kutta más adelante.

Definición 1.3 *Dado un árbol* $\boldsymbol{t} = [\boldsymbol{t}_1, \boldsymbol{t}_2, \ldots, \boldsymbol{t}_m]$*, se definen de manera recursiva las siguientes funciones asociadas:*

- *Orden*, $\rho(\boldsymbol{t})$,
$$\rho(\boldsymbol{t}) = 1 + \rho(\boldsymbol{t}_1) + \cdots + \rho(\boldsymbol{t}_m), \qquad \rho(\boldsymbol{\tau}) = 1.$$
- *Densidad*, $\gamma(\boldsymbol{t})$,
$$\gamma(\boldsymbol{t}) = \rho(\boldsymbol{t})\cdot\gamma(\boldsymbol{t}_1)\cdots\gamma(\boldsymbol{t}_m), \qquad \gamma(\boldsymbol{\tau}) = 1.$$

- *Diferencial elemental*, $\boldsymbol{F}(\boldsymbol{t})(\mathbf{x}_0)$, *donde cada componente se define por*

$$F^I(\boldsymbol{t})(\mathbf{x}_0) = \sum_{I_1,\ldots,I_m=1}^{D} \frac{\partial^m f^I(\mathbf{x}_0)}{\partial x^{I_1}\cdots\partial x^{I_m}} F^{I_1}(\boldsymbol{t_1})(\mathbf{x}_0)\cdots F^{I_m}(\boldsymbol{t_m})(\mathbf{x}_0),$$

  y $\boldsymbol{F}(\boldsymbol{\tau})(\mathbf{x}_0) = \mathbf{f}(\mathbf{x}_0)$.

- *Peso elemental de la etapa i-ésima*, $\Phi_i(\boldsymbol{t})$,

$$\Phi_i(\boldsymbol{t}) = \sum_{j_1,\ldots,j_m=1}^{s} a_{ij_1}\Phi_{j_1}(\boldsymbol{t}_1)a_{ij_2}\Phi_{j_2}(\boldsymbol{t}_2)a_{ij_2}\cdots a_{ij_m}\Phi_{j_m}(\boldsymbol{t}_m), \qquad \Phi_i(\boldsymbol{\tau}) = 1.$$

- *Peso elemental*, $\Phi(\boldsymbol{t})$,

$$\Phi(\boldsymbol{t}) = \sum_{i=1}^{s} b_i\Phi_i(\boldsymbol{t}).$$

Algunas de estas funciones son más sencillas de lo que pueden parecer a simple vista. El orden del árbol es el número de nodos (o vértices) que tiene dicho árbol. La función densidad está relacionada con el hecho de que el árbol sea más estirado o más achatado. El peso elemental es una expresión polinómica de los coeficientes b_i y a_{ij} del método, que incorpora el factor b_i asociado a la raíz, y un factor a_{jk} si en el árbol existe un vértice con índice j que es "padre" de otro vértice con índice k.

Para dejar totalmente claros estos conceptos, procedemos a explicitarlos en un ejemplo sencillo.

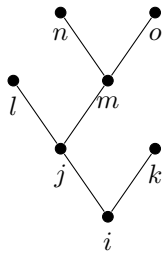


Figura 1.1: Árbol con raíz utilizado en el Ejemplo 2

Ejemplo 2 *Sea* $\boldsymbol{t}$ *el árbol de la Figura 1.1.*

- $\boldsymbol{t} = [[\tau,[\tau,\tau]],\tau]$,
- $\rho(\boldsymbol{t}) = 7$,
- $\gamma(\boldsymbol{t}) = 7\cdot 5\cdot 3 = 105$,
- $F^I(\boldsymbol{t})(\mathbf{x}) = \sum_{J,K,L,M,N,O=1}^{D} \frac{\partial^2 f^I(\mathbf{x})}{\partial x^J\partial x^K} f^K(\mathbf{x}) \frac{\partial^2 f^J(\mathbf{x})}{\partial x^L\partial x^M} f^L(\mathbf{x}) \frac{\partial^2 f^M(\mathbf{x})}{\partial x^N\partial x^O} f^N(\mathbf{x}) f^O(\mathbf{x}),$

- $\Phi_i(\boldsymbol{t}) = \sum_{j,m=1}^{s} c_i a_{ij} c_j a_{jm} c_m^2$.

Llegamos con esto al resultado que nos interesa, que justifica nuestro interés en introducir la teoría de los árboles.

Teorema 1.2 *Un método Runge-Kutta de s etapas y coeficientes (1.2) tiene orden p si, y sólo si,*

$$\Phi(\boldsymbol{t}) = \frac{1}{\gamma(\boldsymbol{t})}$$

para todos los árboles con raíz $\boldsymbol{t}$ *de orden menor o igual que p.*

Demostración. La demostración puede encontrarse en el Capítulo II.2 de [3]. □

Capítulo 2

Los métodos de Gauss: construcción y orden

2.1. Hipótesis simplificadoras

Un método Runge-Kutta de s etapas explícito solo puede llegar a tener orden s como máximo, por lo que nos interesarán los métodos implícitos. El primer método implícito fue usado por Cauchy en 1824; insertó el teorema del valor medio en la cuadratura

$$\mathbf{x}(t_1) = \mathbf{x}(t_0) + \int_{t_0}^{t_1} \mathbf{f}(t, \mathbf{x}(t))dt,$$

y obtuvo

$$\mathbf{x}_1 = \mathbf{x}_0 + h \cdot \mathbf{f}(t_0 + \theta h, \mathbf{x}_0 + \Theta(\mathbf{x}_1 - \mathbf{x}_0)),$$

con $0 \leq \theta, \Theta \leq 1$, $h = (t_1 - t_0)$. Escogiendo $\theta = \Theta = 1$ se obtiene el método de Euler implícito

$$\mathbf{x}_1 = \mathbf{x}_0 + h \cdot \mathbf{f}(t_1, \mathbf{x}_1),$$

que tiene una etapa y orden 1.

Si elegimos $\theta = \Theta = \frac{1}{2}$, llamando $\mathbf{k}_1 = \dfrac{\mathbf{x}_1 - \mathbf{x}_0}{h}$, obtenemos

$$\begin{aligned} \mathbf{k}_1 &= \mathbf{f}(t_0 + \frac{h}{2}, \mathbf{x}_0 + \frac{h}{2}\mathbf{k}_1), \\ \mathbf{x}_1 &= \mathbf{x}_0 + h\mathbf{k}_1. \end{aligned}$$

Este método se conoce como *la regla implícita del punto medio*, tiene una etapa y orden 2, ya que $b_1 = 1$ y $b_1 c_1 = \frac{1}{2}$ (ver Teorema 1.2), y es el primer método de la familia objeto de nuestro estudio, los métodos de Gauss.

Ejemplo 3 *Vamos a construir ahora un método de dos etapas y orden 4. De acuerdo con el Teorema 1.2 necesitamos imponer las siguientes condiciones de orden (recogidas en la Tabla 2.1):*

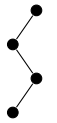
	$1 = \sum_{i=1}^{2} b_i$		$\frac{1}{4} = \sum_{i=1}^{2} b_i c_i^3$
	$\frac{1}{2} = \sum_{i=1}^{2} b_i c_i$		$\frac{1}{8} = \sum_{i,j=1}^{2} b_i c_i a_{ij} c_j$
	$\frac{1}{3} = \sum_{i=1}^{2} b_i c_i^2$		$\frac{1}{12} = \sum_{i,j=1}^{2} b_i a_{ij} c_j^2$
	$\frac{1}{6} = \sum_{i,j=1}^{2} b_i a_{ij} c_j$		$\frac{1}{24} = \sum_{i,j,k=1}^{2} b_i a_{ij} a_{jk} c_k$

Tabla 2.1: Condiciones de orden ligadas a los árboles de orden ≤ 4

Fijándonos primero en las condiciones asociadas a los arbustos (los árboles con $\gamma(t) = \rho(t)$) podemos hallar los valores de b_1, b_2, c_1, c_2 resolviendo el sistema de ecuaciones no lineales

$$\begin{cases} 1 = b_1 + b_2, \\ \frac{1}{2} = b_1 c_1 + b_2 c_2, \\ \frac{1}{3} = b_1 c_1^2 + b_2 c_2^2, \\ \frac{1}{4} = b_1 c_1^3 + b_2 c_2^3, \end{cases} \Leftrightarrow \begin{cases} 1 = b_1 + b_2, \\ \frac{1}{2} - c_1 = b_2(c_2 - c_1), \\ \frac{1}{3} - \frac{c_1}{2} = b_2 c_2 (c_2 - c_1), \\ \frac{1}{4} - \frac{c_1}{3} = b_2 c_2^2 (c_2 - c_1). \end{cases}$$

Dividiendo ahora la tercera ecuación entre la segunda, y la cuarta entre la tercera, obtenemos dos expresiones para c_2 e igualándolas, podemos despejar c_1

$$c_1 = \frac{1}{2} \pm \frac{\sqrt{3}}{6}.$$

Eligiendo $c_1 = \frac{1}{2} - \frac{\sqrt{3}}{6}$ (para conseguir $c_1 < c_2$) despejamos los valores restantes:

$$c_2 = \frac{1}{2} + \frac{\sqrt{3}}{6}, \qquad b_1 = \frac{1}{2}, \qquad b_2 = \frac{1}{2}.$$

Ahora, conocidos b_1, b_2, c_1, c_2, las cuatro siguientes ecuaciones son lineales en los a_{ij}:

$$\begin{cases} a_{11}+a_{12}=c_1, \\ a_{21}+a_{22}=c_2, \\ b_1(a_{11}c_1+a_{12}c_2)+b_2(a_{21}c_1+a_{22}c_2)=\dfrac{1}{6}, \\ b_1c_1(a_{11}c_1+a_{12}c_2)+b_2c_2(a_{21}c_1+a_{22}c_2)=\dfrac{1}{8}, \end{cases} \Leftrightarrow \begin{cases} a_{11}+a_{12}=c_1, \\ a_{11}c_1+a_{12}c_2=\dfrac{4c_2-3}{12(c_2-c_1)}, \\ a_{21}+a_{22}=c_2, \\ a_{21}c_1+a_{22}c_2=\dfrac{4c_1-3}{12(c_1-c_2)}. \end{cases}$$

Operando para resolver los dos subsistemas resultantes, obtenemos:

$$a_{11}=\frac{1}{4}, \quad a_{12}=\frac{1}{4}-\frac{\sqrt{3}}{6}, \quad a_{21}=\frac{1}{4}+\frac{\sqrt{3}}{6}, \quad a_{22}=\frac{1}{4}.$$

Es una sencilla comprobación que las dos ecuaciones de orden que no hemos utilizado se satisfacen para estos valores. Por lo tanto, ya tenemos un método de dos etapas y orden 4, que se conoce como el método de Gauss de dos etapas, cuyo tablero de Butcher aparece en la Tabla 2.2. .

$$\begin{array}{c|cc} \frac{1}{2}-\frac{\sqrt{3}}{6} & \frac{1}{4} & \frac{1}{4}-\frac{\sqrt{3}}{6} \\ \frac{1}{2}+\frac{\sqrt{3}}{6} & \frac{1}{4}+\frac{\sqrt{3}}{6} & \frac{1}{4} \\ \hline & 1/2 & 1/2 \end{array}.$$

Tabla 2.2: Método de Gauss de dos etapas

Si pretendemos construir métodos Runge-Kutta de s etapas y orden $2s$, con este sencillo ejemplo nos damos cuenta de que requerimos de alguna técnica de simplificación, o los cálculos se harán inabordables con solo aumentar un poco el número de etapas del método. Para ello, haremos uso de las siguientes familias de hipótesis simplificadoras:

$$B(p): \quad \sum_{i=1}^{s} b_i c_i^{q-1}=\frac{1}{q} \qquad q=1,\dots,p, \tag{2.1}$$

$$C(\eta): \quad \sum_{j=1}^{s} a_{ij} c_j^{q-1}=\frac{c_i^q}{q} \qquad 1\le i\le s, \;\; q=1,\dots,\eta, \tag{2.2}$$

$$D(\zeta): \quad \sum_{i=1}^{s} b_i c_i^{q-1} a_{ij}=\frac{b_j}{q}(1-c_j^q) \qquad 1\le j\le s, \;\; q=1,\dots,\zeta. \tag{2.3}$$

El cumplimiento de estas reducciones evitará tener que comprobar todas las condiciones de orden necesarias. Procedemos a explicar el porqué.

- La simplificación $B(p)$ significa que se cumplen las condiciones de orden del Teorema 1.2 para todos los arbustos $\boldsymbol{t}=[\boldsymbol{\tau},\dots,\boldsymbol{\tau}]$ hasta orden p.

- La suposición $C(\eta)$ implica que los pares de árboles como los de la Figura 2.1 con $q \leq \eta$ generan condiciones de orden equivalentes. Comprobémoslo:

  Sean $\boldsymbol{t}_1$ y $\boldsymbol{t}_2$ dos árboles con raíz con el mismo número de nodos que tienen una parte común (que puede ser arbitraria y que en la Figura 2.1 se ha rodeado por una línea discontinua y que denotaremos por $\boldsymbol{t}$) y que se diferencian en que uno de sus nodos, con índice j en la Figura 2.1, en el árbol $\boldsymbol{t}_1$ tiene un hijo (con índice k) que a su vez tiene $q-1$ hijos finales, mientras que en el árbol $\boldsymbol{t}_2$ dicho nodo con índice j tiene q hijos finales. Aplicando la reducción $C(q)$ se tiene que

$$\sum_{i=1}^{s} b_i \Phi_i(\boldsymbol{t_1}) = \sum_{i=1}^{s} b_i \Phi_i(\boldsymbol{t}) \left[\sum_{k=1}^{s} a_{jk} c_k^{q-1} \right] = \frac{1}{q} \sum_{i=1}^{s} b_i \Phi_i(\boldsymbol{t}) c_j^q = \frac{1}{q} \sum_{i=1}^{s} b_i \Phi_i(\boldsymbol{t_2}).$$

  El resultado se concluye del hecho de que $\gamma(\boldsymbol{t}_1) = q \cdot \gamma(\boldsymbol{t}_2)$.

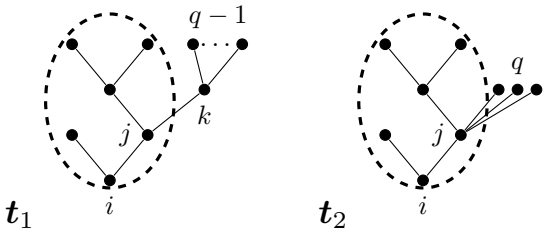


Figura 2.1: Hipótesis simplificadora $C(q)$

- Si se cumple $D(q)$, las condiciones de orden de los árboles $\boldsymbol{t}$ y $\boldsymbol{t}_1$ de la derecha en la Figura 2.2 implican la del árbol $\boldsymbol{T}$ de la izquierda, entendiendo que la parte rodeada por la línea discontinua (el árbol $\boldsymbol{t}$) puede ser arbitraria.
  Queremos ver que se cumple $\Phi(\boldsymbol{T}) = \dfrac{1}{\gamma(\boldsymbol{T})}$, si se tiene $\Phi(\mathbf{t}_1) = \dfrac{1}{\gamma(\mathbf{t}_1)}$ y $\Phi(\mathbf{t}) = \dfrac{1}{\gamma(\mathbf{t})}$:

$$\begin{aligned}
\Phi(\boldsymbol{T}) &= \sum_{i=1}^{s} b_i \Phi_i(\boldsymbol{T}) = \sum_{i,j=1}^{s} b_i c_i^{q-1} a_{ij} \Phi_j(\boldsymbol{t}) \\
&= \sum_{j=1}^{s} \Phi_j(\boldsymbol{t}) \left[\sum_{i=1}^{s} b_i c_i^{q-1} a_{ij} \right] = \frac{1}{q} \left[\sum_{j=1}^{s} b_j \Phi_j(\boldsymbol{t}) - \sum_{j=1}^{s} b_j c_j^q \Phi_j(\boldsymbol{t}) \right] \\
&= \frac{1}{q} \left[\frac{1}{\gamma(\boldsymbol{t})} - \frac{1}{\gamma(\boldsymbol{t}_1)} \right] = \frac{1}{q} \left[\frac{\rho(\boldsymbol{t}) + q}{\gamma(\boldsymbol{T})} - \frac{\rho(\boldsymbol{t})}{\gamma(\boldsymbol{T})} \right] = \frac{1}{\gamma(\boldsymbol{T})},
\end{aligned}$$

  donde hemos usado que se cumple $D(q)$ y que $\rho(\boldsymbol{T}) = \rho(\boldsymbol{t}) + q$, $\gamma(\boldsymbol{T}) = \rho(\boldsymbol{T}) \cdot \gamma(\boldsymbol{t})$ y $\gamma(\boldsymbol{t}_1) = \dfrac{\gamma(\boldsymbol{T})}{\rho(\boldsymbol{t})}$.

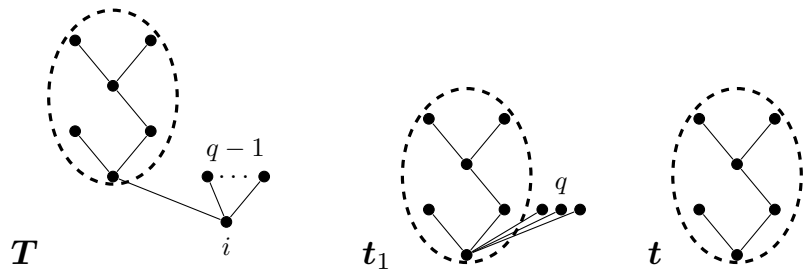


Figura 2.2: Hipótesis simplificadora $D(q)$

Una vez entendidas estas condiciones, podemos enunciar el siguiente teorema.

Teorema 2.1 *Si se cumplen $B(p)$, $C(\eta)$ y $D(\zeta)$ con $p \leq 2\eta+2$ y $p \leq \eta+\zeta+1$, entonces el método (1.3)-(1.4) es de orden p.*

Demostración. Para que el método tuviera orden p habrían de satisfacerse todas las condiciones asociadas a los árboles de orden menor o igual que p. Sin embargo, gracias a nuestras hipótesis simplificadoras, algunas son superfluas. La reducción $C(\eta)$ implica que basta considerar árboles $[\boldsymbol{t}_1, \ldots, \boldsymbol{t}_m]$ de orden menor o igual que p tales que todos los subárboles son iguales a $\boldsymbol{\tau}$ o hay exactamente un $\boldsymbol{t}_i$ con orden $\rho(\boldsymbol{t}_i) \geq \eta + 1$.

- Si hubiera dos o más subárboles de orden mayor o igual a $\eta + 1$, contradiría $p \leq 2\eta + 2$.
- Para los subárboles de orden menor que $\eta + 1$ se puede aplicar la reducción $C(\eta)$ hasta llevarlos a un arbusto, "bajando" ramas de nivel en nivel, como se puede apreciar en la Figura 2.3.

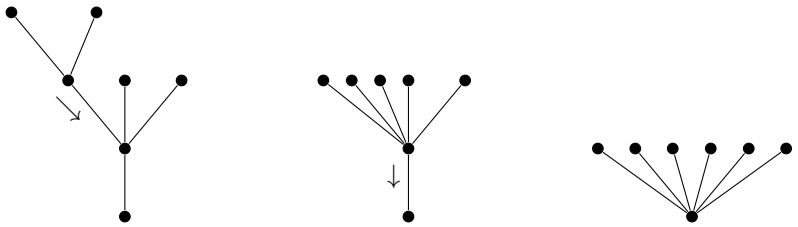


Figura 2.3: Aplicación sucesiva de la hipótesis simplificadora $C(\eta)$

- En el caso de que haya un subárbol de orden mayor o igual que $\eta + 1$, el resto de subárboles, que son iguales a $\boldsymbol{\tau}$, han de ser como mucho $\zeta - 1$ para respetar $p \leq \eta + \zeta + 1$. A dicho subárbol podemos entonces aplicarle la reducción $D(\zeta)$. Por lo tanto, solo restan los arbustos, cuyas condiciones de orden asociadas se satisfacen gracias a $B(p)$. □

Como estamos interesados en conseguir con s etapas métodos de orden $2s$, una forma de conseguirlo sería hallar coeficientes b_i, c_i, $1 \leq i \leq s$, que satisfagan $B(2s)$ y, a continuación, coeficientes a_{ij} que satisfagan $C(s)$ y $D(s)$. Pero antes, vamos a demostrar un resultado que muestra una consecuencia inmediata de la condición $C(\eta)$.

Lema 2.1 *Si se cumple $C(\eta)$, las etapas intermedias*

$$\mathbf{X}_i = \mathbf{x}_n + h\sum_{j=1}^{s} a_{ij}\mathbf{f}(t_n + c_j h, \mathbf{X}_j), \qquad 1 \le i \le s$$

de un método Runge-Kutta satisfacen

$$\mathbf{X}_i - \mathbf{x}(t_n + c_i h) = \mathcal{O}(h^{\eta+1}),$$

siendo $\mathbf{x}(t)$ la solución del sistema diferencial (1.1) con condición inicial $\mathbf{x}(t_n) = \mathbf{x}_n$.

Demostración. Haciendo el desarrollo de Taylor de la solución exacta y utilizando la reducción $C(\eta)$, obtenemos

$$\begin{aligned}
\mathbf{x}(t_n + c_i h) &= \mathbf{x}_n + c_i h\mathbf{x}'(t_n) + \frac{c_i^2}{2}h^2\mathbf{x}''(t_n) + \cdots + \frac{c_i^\eta}{\eta}\frac{h^\eta}{(\eta-1)!}\mathbf{x}^{(\eta)}(t_n) + \mathcal{O}(h^{\eta+1}) \\
&= \mathbf{x}_n + h\left[c_i\mathbf{x}'(t_n) + \frac{c_i^2}{2}h\mathbf{x}''(t_n) + \cdots + \frac{c_i^\eta}{\eta}\frac{h^{\eta-1}}{(\eta-1)!}\mathbf{x}^{(\eta)}(t_n)\right] + \mathcal{O}(h^{\eta+1}) \\
&= \mathbf{x}_n + h\left[\sum_{j=1}^{s} a_{ij}\mathbf{x}'(t_n) + \cdots + \frac{h^{\eta-1}}{(\eta-1)!}\sum_{j=1}^{s} a_{ij}c_j^{\eta-1}\mathbf{x}^{(\eta)}(t_n)\right] + \mathcal{O}(h^{\eta+1}) \\
&= \mathbf{x}_n + h\sum_{j=1}^{s} a_{ij}\mathbf{x}'(t_n + c_j h) + \mathcal{O}(h^{\eta+1}),
\end{aligned}$$

de donde se deduce que

$$\mathbf{X}_i - \mathbf{x}(t_n + c_i h) = h\sum_{j=1}^{s} a_{ij}[\mathbf{f}(t_n + c_j h, \mathbf{X_j}) - \mathbf{f}(t_n + c_j h, \mathbf{x}(t_n + c_j h))] + \mathcal{O}(h^{\eta+1}).$$

Utilizando ahora la condición de Lipschitz (con constante L) se tiene

$$||\mathbf{X}_i - \mathbf{x}(t_n + c_i h)|| \le hL\sum_{j=1}^{s} |a_{ij}| \cdot ||\mathbf{X}_j - \mathbf{x}(t_n + c_j h)|| + \mathcal{O}(h^{\eta+1}).$$

Si denotamos por $|\mathcal{A}|$ la matriz $s \times s$ con elementos $(|a_{ij}|)_{1\le i,j\le s}$ podemos escribir

$$(I - hL|\mathcal{A}|)\mathbf{E} = \mathcal{O}(h^{\eta+1}),$$

donde $\mathbf{E} = (||\mathbf{X}_1 - \mathbf{x}(t_n + c_1 h)||, \ldots, ||\mathbf{X}_s - \mathbf{x}(t_n + c_s h||)^T$. Si se satisface (1.7) se tiene que la matriz $I - hL|\mathcal{A}|$ es invertible (puesto que $||hL|\mathcal{A}|||_\infty < 1$) y su inversa tiene norma acotada, luego podemos concluir que $\mathbf{E} = \mathcal{O}(h^{\eta+1})$. □

2.2. Métodos de colocación

Continuando con nuestro objetivo de encontrar una aproximación numérica a la solución de (1.1) en tiempo $t_0 + h$, nos planteamos ahora una idea alternativa que da lugar a los llamados *métodos de colocación* que se definen de la siguiente manera:

Definición 2.1 *Sea s un entero positivo y $c_1, \ldots, c_s$ números reales distintos (usualmente entre 0 y 1). El correspondiente polinomio de colocación $\mathbf{u}(t)$ de grado s está definido por las condiciones*

$$\begin{cases} \mathbf{u}(t_0) = \mathbf{x_0}, \\ \mathbf{u}'(t_0 + c_i h) = \mathbf{f}(t_0 + c_i h, \mathbf{u}(t_0 + c_i h)), \quad 1 \le i \le s. \end{cases} \tag{2.4}$$

Se puede tomar entonces, como solución numérica de (1.1) en $t_0 + h$, el valor $\mathbf{x}_1 = \mathbf{u}(t_0 + h)$.

Observamos que el polinomio de colocación $\mathbf{u}(t)$, además de satisfacer la condición inicial $\mathbf{u}(t_0) = \mathbf{x}_0$, satisface también que su derivada en los tiempos $t_0 + c_i h$, $1 \le i \le s$, coincide con el campo vectorial de la ecuación diferencial (1.1) en dichos tiempos.

No es estrictamente necesario que los c_i sean todos diferentes, pero lo supondremos así para facilitar los cálculos.

Los siguientes resultados justifican nuestro interés en estos métodos.

Teorema 2.2 *El método de colocación (2.4) es equivalente al método Runge-Kutta implícito de s etapas (1.3)-(1.4) con coeficientes*

$$a_{ij} = \int_0^{c_i} l_j(\tau) d\tau, \quad b_j = \int_0^1 l_j(\tau) d\tau, \quad i, j = 1, \ldots, s, \tag{2.5}$$

donde $l_j(\tau)$ representa el polinomio de Lagrange asociado al nodo c_j

$$l_j(\tau) = \prod_{k \neq j} \frac{\tau - c_k}{c_j - c_k}.$$

Demostración. Llamamos $\mathbf{k}_i = \mathbf{u}'(t_0 + c_i h)$; de esta manera se tiene, por la fórmula interpolatoria de Lagrange y puesto que $\mathbf{u}'(t_0 + \tau h)$ es un polinomio de grado menor o igual que $s - 1$,

$$\mathbf{u}'(t_0 + \tau h) = \sum_{j=1}^{s} \mathbf{k}_j l_j(\tau).$$

Ahora, dado que

$$\mathbf{u}(t_0 + c_i h) = \mathbf{x_0} + h \int_0^{c_i} \mathbf{u}'(t_0 + \tau h) d\tau, \tag{2.6}$$

solo tenemos que introducirlo en (2.4) para obtener

$$\begin{aligned} \mathbf{k}_i &= \mathbf{u}'(t_0 + c_i h) = \mathbf{f}\big(t_0 + c_i h, \mathbf{u}(t_0 + c_i h)\big) = \mathbf{f}\big(t_0 + c_i h, \mathbf{x}_0 + h \int_0^{c_i} \mathbf{u}'(t_0 + \tau h) d\tau\big) \\ &= \mathbf{f}\big(t_0 + c_i h, \mathbf{x}_0 + h \sum_{j=1}^{s} \mathbf{k}_j \int_0^{c_i} l_j(\tau) d\tau\big) = \mathbf{f}\big(t_0 + c_i h, \mathbf{x}_0 + h \sum_{j=1}^{s} a_{ij} \mathbf{k}_j\big). \\ \mathbf{x}_1 &= \mathbf{u}(t_0 + h) = \mathbf{x}_0 + h \int_0^1 \mathbf{u}'(t_0 + \tau h) d\tau = \mathbf{x}_0 + h \sum_{j=1}^{s} b_j \mathbf{k}_j, \end{aligned}$$

expresiones que corresponden a un paso de longitud h con el método Runge-Kutta de s etapas con coeficientes dados por (2.5). □

Una consecuencia natural que se obtiene a partir de este resultado es la existencia y unicidad del polinomio de colocación (para h suficientemente pequeño), pues se sigue de lo demostrado para los métodos Runge-Kutta en la Sección 1.3. Además, se tiene el siguiente resultado:

Teorema 2.3 *Un método Runge-Kutta implícito con todos los coeficientes c_i distintos y orden al menos s es un método de colocación si, y solo si, se satisface $C(s)$.*

Demostración. El orden s garantiza que $\sum_{i=1}^{s} b_i c_i^{q-1} = \frac{1}{q}, \;\; q = 1, \ldots, s$, y la fórmula de cuadratura es exacta para los polinomios de la base de Lagrange, luego los b_i siempre satisfacen el segundo grupo de igualdades de (2.5). Entonces, solo falta ver que el que se satisfaga $C(s)$ es equivalente a las igualdades del lado izquierdo de (2.5).

⇐) Conocidos los valores c_i, $1 \leq i \leq s$, la condición $C(s)$ determina unívocamente los coeficientes a_{ij}, pues las condiciones

$$\sum_{j=1}^{s} a_{ij} c_j^{q-1} = \frac{c_i^q}{q}, \qquad 1 \leq i \leq s, \;\; q = 1, \ldots, s \tag{2.7}$$

forman s sistemas lineales de s ecuaciones y s incógnitas con matriz de Vandermonde invertible

$$\begin{pmatrix} 1 & 1 & \cdots & 1 \\ c_1 & c_2 & \cdots & c_s \\ c_1^2 & c_2^2 & \cdots & c_s^2 \\ \vdots & \vdots & \cdots & \vdots \\ c_1^{s-1} & c_2^{s-1} & \cdots & c_s^{s-1} \end{pmatrix} \begin{pmatrix} a_{i1} \\ a_{i2} \\ a_{i3} \\ \vdots \\ a_{is} \end{pmatrix} = \begin{pmatrix} c_i \\ c_i^2/2 \\ c_i^3/3 \\ \vdots \\ c_i^s/s \end{pmatrix}, \tag{2.8}$$

uno para cada índice i con $1 \leq i \leq s$.

Gracias al cumplimiento de (2.7), también se satisface

$$\sum_{j=1}^{s} a_{ij} p(c_j) = \int_0^{c_i} p(\tau) d\tau,$$

para todo polinomio p de grado menor o igual que $s-1$. En particular, tomando en la igualdad anterior como p los polinomios de la base de Lagrange, se obtiene la primera expresión de (2.5).

⇒) Recíprocamente, si suponemos que los coeficientes a_{ij} vienen dados por (2.5), entonces para cada polinomio p de grado $\leq s-1$ se tiene

$$\begin{aligned} \int_0^{c_i} p(\tau) d\tau &= \int_0^{c_i} \sum_{j=1}^{s} p(c_j) l_j(\tau) d\tau \\ &= \sum_{j=1}^{s} p(c_j) \int_0^{c_i} l_j(\tau) d\tau = \sum_{j=1}^{s} a_{ij} p(c_j). \end{aligned}$$

Particularizando esta relación para los polinomios $p = 1, x, x^2, \ldots, x^{s-1}$, se obtiene (2.7). □

El siguiente resultado nos acerca a la noción de ortogonalidad presente en los métodos de Gauss a los que nos dedicamos en esta memoria.

Teorema 2.4 *Sea* $M(\tau) = \prod_{i=1}^{s}(\tau - c_i)$*, y supongamos que* M *es ortogonal a los polinomios de grado menor o igual que* $r-1$*; esto es,*

$$\int_0^1 M(\tau)\tau^{q-1}d\tau = 0, \qquad q = 1, \ldots, r. \tag{2.9}$$

Entonces, el método de colocación (2.4) asociado a los nodos c_i*,* $1 \leq i \leq s$*, tiene orden* $p = r + s$*.*

Demostración. Por el Teorema 2.3 tenemos asegurado $C(s)$. Veamos que, gracias a la condición de ortogonalidad, se satisface $B(r+s)$:

Consideramos $M(\tau)$, de grado s, y ortogonal a los polinomios de grado $\leq r-1$. Entonces, aplicando la cuadratura de Gauss, se sigue

$$\int_0^1 f(\tau)d\tau = \sum_{j=1}^{s} b_j f(c_j),$$

con $b_j = \int_0^1 l_j(\tau)d\tau$ y f cualquier polinomio de grado $\leq s + r - 1$. Aplicándolo a $f(x) = x^{k-1}$, $1 \leq k \leq s + r$, se tiene

$$\frac{1}{k} = \int_0^1 \tau^{k-1}d\tau = \sum_{j=1}^{s} b_j c_j^{k-1}, \qquad k = 1, \ldots, s + r,$$

que no es más que la condición $B(r+s)$.

Para poder aplicar el Teorema 2.1 y concluir que el método tiene orden $p = r + s$, falta ver que también se cumple $D(r)$ a partir de $B(r+s)$ y $C(s)$:

Consideramos la matriz de Vandermonde V utilizada en (2.8), y los vectores $\mathbf{u}^{(q)}$ y $\mathbf{v}^{(q)}$, con $q \leq r$, cuyas componentes son $u_j^{(q)} = \sum_{i=1}^{s} b_i c_i^{q-1} a_{ij}$ y $v_j^{(q)} = b_j(1 - c_j^q)/q$, $1 \leq j \leq s$, respectivamente. Queremos probar que $\mathbf{u}^{(q)} = \mathbf{v}^{(q)}$. Para ello, multiplicaremos ambos vectores por la matriz V, veremos que $V \cdot \mathbf{u}^{(q)} = V \cdot \mathbf{v}^{(q)}$ y, como V es invertible, ha de ser $\mathbf{u}^{(q)} = \mathbf{v}^{(q)}$.

La coordenada l de $V \cdot \mathbf{u}^{(q)}$ es

$$(V \cdot \mathbf{u}^{(q)})_l = \sum_{j=1}^{s} c_j^{l-1} \sum_{i=1}^{s} b_i c_i^{q-1} a_{ij}, \tag{2.10}$$

mientras que la misma coordenada de $V \cdot \mathbf{v}^{(q)}$ es

$$(V \cdot \mathbf{v}^{(q)})_l = \frac{1}{q} \sum_{j=1}^{s} (b_j c_j^{l-1} - b_j c_j^{l+q-1}). \tag{2.11}$$

Manipulamos (2.10) y (2.11), utilizando $C(s)$ y $B(r+s)$ para obtener

$$\begin{aligned}\sum_{i,j=1}^{s} b_i c_i^{q-1} a_{ij} c_j^{l-1} &= \sum_{i=1}^{s} b_i c_i^{q-1} \left[\sum_{j=1}^{s} a_{ij} c_j^{l-1}\right] = \frac{1}{l}\sum_{i=1}^{s} b_i c_i^{l+q-1} = \frac{1}{l(l+q)},\\ \frac{1}{q}\sum_{j=1}^{s}(b_j c_j^{l-1} - b_j c_j^{l+q-1}) &= \frac{1}{q}\left[\sum_{j=1}^{s} b_j c_j^{l-1} - \sum_{j=1}^{s} b_j c_j^{l+q-1}\right] = \frac{1}{q}(\frac{1}{l} - \frac{1}{l+q}) = \frac{1}{l(l+q)}.\end{aligned}$$

□

2.3. Elección óptima de los nodos

El Teorema 2.4 nos dice cómo escoger los nodos c_i, $1 \le i \le s$, pues nuestro objetivo es que se satisfaga (2.9) para r lo más alto posible. Esto se consigue eligiendo para cada s como c_i los ceros del *polinomio de Legendre trasladado* de grado s

$$\tilde{P}_s(x) = \frac{1}{s!}\frac{d^s}{dx^s}\left(x^s(x-1)^s\right),$$

obteniendo así métodos de colocación basados en la cuadratura gaussiana. Estos métodos se denominan *métodos de Gauss*.

El polinomio $\tilde{P}_s(x)$ es el resultado de aplicar una transformación afín al polinomio de Legendre

$$P_s(y) = \frac{1}{2^s s!}\frac{d^s}{dy^s}\left((y+1)^s(y-1)^s\right),$$

dada por el cambio de variable $y = 2x - 1$. De esta manera, conseguimos ortogonalidad en el intervalo $[0,1]$. Algunas de las propiedades más destacables de estos polinomios son las siguientes:

1. $\int_0^1 \tilde{P}_s(x)\tilde{P}_t(x)dx = 0, \quad s \neq t,$
2. $\int_0^1 \tilde{P}_s(x)^2 dx = \dfrac{1}{2s+1}, \quad s = 0,1,2,\ldots,$
3. $\tilde{P}_s(1) = 1, \quad s = 0,1,2,\ldots,$
4. $\tilde{P}_s(1-x) = (-1)^s\tilde{P}_s(x), \quad s = 0,1,2,\ldots,$
5. $\tilde{P}_s$ tiene s raíces reales distintas en el intervalo $(0,1)$, $s = 0,1,2,\ldots$

Las demostraciones de todas estas propiedades se encuentran en el Apéndice A.

En las Tablas 2.3, 2.4 y 2.5 se muestran los coeficientes de los métodos de Gauss de 1, 2, 3 y 4 etapas.

$\frac{1}{2}$	$\frac{1}{2}$
	1

$\frac{1}{2}-\frac{\sqrt{3}}{6}$	$\frac{1}{4}$	$\frac{1}{4}-\frac{\sqrt{3}}{6}$
$\frac{1}{2}+\frac{\sqrt{3}}{6}$	$\frac{1}{4}+\frac{\sqrt{3}}{6}$	$\frac{1}{4}$
	$\frac{1}{2}$	$\frac{1}{2}$

Tabla 2.3: Métodos de Gauss de 1 y 2 etapas

$\frac{1}{2}-\frac{\sqrt{15}}{10}$	$\frac{5}{36}$	$\frac{2}{9}-\frac{\sqrt{15}}{15}$	$\frac{5}{36}-\frac{\sqrt{15}}{30}$
$\frac{1}{2}$	$\frac{5}{36}+\frac{\sqrt{15}}{24}$	$\frac{2}{9}$	$\frac{5}{36}-\frac{\sqrt{15}}{24}$
$\frac{1}{2}+\frac{\sqrt{15}}{10}$	$\frac{5}{36}+\frac{\sqrt{15}}{30}$	$\frac{2}{9}+\frac{\sqrt{15}}{15}$	$\frac{5}{36}$
	$\frac{5}{18}$	$\frac{4}{9}$	$\frac{5}{18}$

Tabla 2.4: Método de Gauss de 3 etapas

$\frac{1}{2}-\omega_2$	ω_1	$\omega_1'-\omega_3+\omega_4'$	$\omega_1'-\omega_3-\omega_4'$	$\omega_1-\omega_5$
$\frac{1}{2}-\omega_2'$	$\omega_1-\omega_3'+\omega_4$	ω_1'	$\omega_1'-\omega_5'$	$\omega_1-\omega_3'-\omega_4$
$\frac{1}{2}+\omega_2'$	$\omega_1+\omega_3'+\omega_4$	$\omega_1'+\omega_5'$	ω_1'	$\omega_1+\omega_3'-\omega_4$
$\frac{1}{2}+\omega_2$	$\omega_1+\omega_5$	$\omega_1'+\omega_3+\omega_4'$	$\omega_1'+\omega_3-\omega_4'$	ω_1
	$2\omega_1$	$2\omega_1'$	$2\omega_1'$	$2\omega_1$

$$\omega_1=\frac{1}{8}-\frac{\sqrt{30}}{144}, \qquad \omega_1'=\frac{1}{8}+\frac{\sqrt{30}}{144},$$

$$\omega_2=\frac{1}{2}\sqrt{\frac{15+2\sqrt{30}}{35}}, \qquad \omega_2'=\frac{1}{2}\sqrt{\frac{15-2\sqrt{30}}{35}},$$

$$\omega_3=\omega_2\left(\frac{1}{6}+\frac{\sqrt{30}}{24}\right), \qquad \omega_3'=\omega_2'\left(\frac{1}{6}-\frac{\sqrt{30}}{24}\right),$$

$$\omega_4=\omega_2\left(\frac{1}{21}+\frac{5\sqrt{30}}{168}\right), \qquad \omega_4'=\omega_2'\left(\frac{1}{21}-\frac{5\sqrt{30}}{168}\right),$$

$$\omega_5=\omega_2-2\omega_3, \qquad \omega_5'=\omega_2'-2\omega_3'.$$

Tabla 2.5: Método de Gauss de 4 etapas

Escogiendo los coeficientes $c_1, \ldots, c_s$ basándonos en la cuadratura gaussiana, que es exacta para los polinomios hasta grado $2s$, aseguramos el cumplimiento de la reducción $B(2s)$:

$$\frac{1}{k} = \int_0^1 x^{k-1} dx = \sum_{i=1}^{s} b_i c_i^{k-1}, \qquad k = 1, \ldots, 2s.$$

Este hecho nos permite enunciar el siguiente resultado.

Teorema 2.5 *El método de Gauss de s etapas tiene orden $2s$.*

Demostración. Por ser un método de colocación basado en la cuadratura gaussiana, se cumplen $B(2s)$ y $C(s)$. Como se procedió en la demostración del Teorema 2.4, estas dos reducciones implican $D(s)$, luego basta aplicar el Teorema 2.1. □

Capítulo 3

Los métodos de Gauss: estabilidad

3.1. Introducción

Al igual que el orden, la estabilidad es uno de los factores que hemos de estudiar en referencia a los métodos Runge-Kutta. Veamos de dónde surge esta idea:

Ejemplo 4 *Consideramos el problema de valores iniciales*

$$\left\{ \begin{array}{rcll} x'(t) & = & \lambda \cdot x(t), & Re(\lambda) << 0, \\ x(0) & = & x_0, & \end{array} \right. \tag{3.1}$$

donde la notación $Re(\lambda) << 0$ indica que $Re(\lambda) < 0$ y $|Re(\lambda)|$ es grande. Claramente, la solución exacta viene dada por $x(t) = x_0 \cdot e^{\lambda t}$, que tiende a 0 rápidamente cuando $t \to \infty$. La pregunta natural es ¿la solución numérica conservará también esta propiedad?, es decir, ¿$\lim_{n\to\infty} x_n = 0$? Veamos qué ocurre con la regla implícita del punto medio:

$$\begin{array}{rcl} k_1 & = & f(t_n + \frac{h}{2}, x_n + \frac{h}{2}k_1) = \lambda(x_n + \frac{h}{2}k_1) \Longrightarrow k_1 = \dfrac{\lambda x_n}{1 - \frac{\lambda h}{2}}, \\ x_{n+1} & = & x_n + hk_1 = R(\lambda h)x_n, \end{array}$$

donde $R(z) = \dfrac{1 + \frac{z}{2}}{1 - \frac{z}{2}}$.
¿Qué tiene que pasar para que $\lim_{n\to\infty} x_n = 0$, con independencia del valor de x_0? Está claro que ha de ser $|R(\lambda h)| < 1$, pues

$$x_{n+1} = R(\lambda h)x_n = [R(\lambda h)]^2 x_{n-1} = \cdots = [R(\lambda h)]^{n+1} x_0.$$

En el caso de la regla implícita del punto medio, $|R(z)| < 1$ para todo z con $Re(z) < 0$ y, por tanto, para cualquier longitud de paso h utilizada, la solución numérica satisface $\lim_{n\to\infty} x_n = 0$, la misma propiedad asintótica que tiene la solución exacta de (3.1).

Definición 3.1 *La función $R(z)$ que satisface $x_{n+1} = R(h\lambda)x_n$, siendo x_n la solución numérica de (3.1) en $t_n = nh$, se llama función de estabilidad del método y puede interpretarse tomando $z = h\lambda$ como la solución numérica tras un paso de longitud h de la famosa ecuación de Dahlquist*

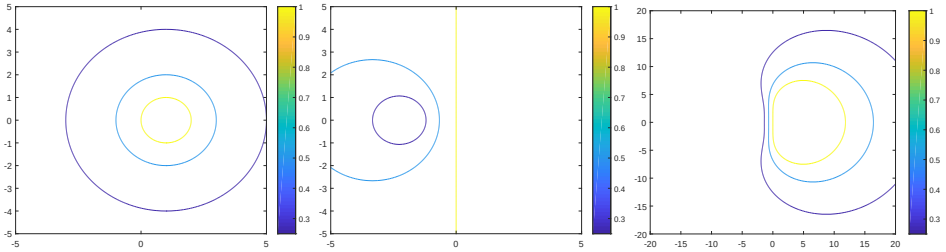
$$x' = \lambda x, \qquad x(0) = 1. \tag{3.2}$$

El conjunto

$$S = \{z \in \mathbb{C} : |R(z)| \leq 1\}$$

se llama región de estabilidad del método.

A continuación mostramos las regiones de estabilidad de algunos métodos Runge-Kutta.



(a) Euler implícito (b) Regla implícita del punto medio (c) Radau IA de orden 5

La función de estabilidad está ligada al método numérico que usemos, así que veamos qué forma tiene cuando usamos los métodos Runge-Kutta implícitos.

Proposición 3.1 *La función de estabilidad de un método Runge-Kutta con coeficientes (1.2) tiene la siguiente forma:*

$$R(z) = \frac{det(I - z\mathcal{A} + z\mathbf{1}\mathbf{b}^T)}{det(I - z\mathcal{A})}, \tag{3.3}$$

con $\mathbf{1} = (1, \ldots, 1)^T \in \mathbb{R}^s$.
Una forma alternativa pero equivalente de definir dicha función es

$$R(z) = 1 + z\mathbf{b}^T(I - z\mathcal{A})^{-1}\mathbf{1}, \tag{3.4}$$

Demostración. Para probar la primera fórmula, solo debemos aplicar el método (1.5)-(1.6) a la ecuación de Dahlquist (3.2), lo que nos lleva al sistema lineal $(s+1) \times (s+1)$

$$\begin{pmatrix} I - z\mathcal{A} & 0 \\ -z\mathbf{b}^T & 1 \end{pmatrix} \begin{pmatrix} \mathbf{X} \\ x_{n+1} \end{pmatrix} = x_n \begin{pmatrix} \mathbf{1} \\ 1 \end{pmatrix},$$

donde $\mathbf{X} = [X_1, \ldots, X_s]^T$. Aplicando la regla de Cramer para despejar x_{n+1}, se obtiene lo siguiente

$$x_{n+1} = \frac{det\begin{pmatrix} I - z\mathcal{A} & \mathbf{1} \\ -z\mathbf{b}^T & 1 \end{pmatrix}}{det(I - z\mathcal{A})} x_n = \frac{det\begin{pmatrix} I - z\mathcal{A} + z\mathbf{1}\mathbf{b}^T & 0 \\ -z\mathbf{b}^T & 1 \end{pmatrix}}{det(I - z\mathcal{A})} x_n = \frac{det(I - z\mathcal{A} + z\mathbf{1}\mathbf{b}^T)}{det(I - z\mathcal{A})} x_n,$$

con lo que $R(z) = \dfrac{det(I - z\mathcal{A} + z\mathbf{1}\mathbf{b}^T)}{det(I - z\mathcal{A})}$.

Para probar la segunda fórmula, aplicamos el método (1.5) a la ecuación de Dahlquist, y nos fijamos en la relación que define las etapas intermedias, para obtener $\mathbf{X} = (I - z\mathcal{A})^{-1}\mathbf{1}x_n$. Sustituyendo este valor de $\mathbf{X}$ en (1.6), obtenemos

$$x_{n+1} = x_n + z\mathbf{b}^T(I - z\mathcal{A})^{-1}\mathbf{1} \cdot x_n = \left[1 + z\mathbf{b}^T(I - z\mathcal{A})^{-1}\mathbf{1}\right] x_n,$$

con lo que $R(z) = 1 + z\mathbf{b}^T(I - z\mathcal{A})^{-1}\mathbf{1}$. □

De (3.3) y (3.4) observamos que, para métodos Runge-Kutta explícitos, $R(z)$ es un polinomio de grado menor o igual que s, que coincide con el desarrollo de Taylor de la función exponencial hasta los términos en z^p incluidos, donde p es el orden del método. Para métodos implícitos, $R(z)$ es una función racional, cuyo numerador y denominador son polinomios en z ambos de grado menor o igual que s, y que deben aproximar a la función e^z hasta los términos en z^p incluidos.

La Tabla 3.1 presenta las funciones de estabilidad de algunos métodos Runge-Kutta.

	Método	$R(z)$
a)	Euler implícito	$\dfrac{1}{1-z}$
b)	Regla implícita del punto medio	$\dfrac{1+z/2}{1-z/2}$
c)	Gauss de 2 etapas	$\dfrac{1+z/2+z^2/12}{1-z/2+z^2/12}$

Tabla 3.1: Funciones de estabilidad de algunos métodos Runge-Kutta.

3.2. A-estabilidad

En esta sección queremos estudiar si los métodos de Gauss son A-estables. Para ello, introducimos brevemente el concepto de aproximante de Padé.

Definición 3.2 *Un método cuya región de estabilidad satisfaga*

$$S \supset \mathbb{C}^- = \{z \in \mathbb{C} : Re(z) \leq 0\},$$

se denomina A-estable.

Los *aproximantes de Padé* son funciones racionales que, fijados los grados del numerador y el denominador, tienen el mejor orden de aproximación posible. Su origen yace en la teoría de fracciones continuas, y jugaron un papel fundamental en la prueba de la trascendencia del número e dada por Hermite en 1873. Estas aproximaciones óptimas pueden obtenerse también para la función exponencial, e^z, como muestra el siguiente Teorema.

Teorema 3.1 *(Capítulo IV.3 de [1]) El aproximante de Padé (k, j) a la exponencial viene dado por*

$$R_{kj}(z) = \frac{P_{kj}(z)}{Q_{kj}(z)},$$

donde

$$\begin{aligned} P_{kj}(z) &= 1 + \frac{k}{j+k}z + \frac{k(k-1)}{(j+k)(j+k-1)} \cdot \frac{z^2}{2!} + \cdots + \frac{k(k-1)\cdots 1}{(j+k)\cdots(j+1)} \cdot \frac{z^k}{k!}, \\ Q_{kj}(z) &= 1 - \frac{j}{k+j}z + \frac{j(j-1)}{(k+j)(k+j-1)} \cdot \frac{z^2}{2!} + \cdots + \frac{j(j-1)\cdots 1}{(k+j)\cdots(k+1)} \cdot \frac{z^j}{j!} \\ &= P_{jk}(-z). \end{aligned}$$

Es la única aproximación racional a e^z de orden $k+j$ con numerador y denominador de grados k y j respectivamente.

En la Tabla 3.2 se muestran los primeros aproximantes de Padé a la exponencial.

$j\backslash k$	$k=0$	$k=1$	$k=2$
$j=0$	1	$1+z$	$1+z+z^2/2$
$j=1$	$\dfrac{1}{1-z}$	$\dfrac{1+z/2}{1-z/2}$	$\dfrac{1+2z/3+z^2/6}{1-z/3}$
$j=2$	$\dfrac{1}{1-z+z^2/2}$	$\dfrac{1+z/3}{1-2z/3+z^2/6}$	$\dfrac{1+z/2+z^2/12}{1-z/2+z^2/12}$

Tabla 3.2: Aproximantes de Padé R_{kj} a e^z para $0 \leq j, k \leq 2$.

Se aprecia que algunas de las funciones de estabilidad de métodos Runge-Kutta incluidos en la Tabla 3.1 son aproximantes de Padé. En particular, la regla implícita del punto medio corresponde a $R_{11}(z)$, y el método de Gauss de dos etapas, a $R_{22}(z)$.

En general, si consideramos el método de Gauss de s etapas y analizamos su función de estabilidad dada por (3.3), observamos que tanto $det(I - z\mathcal{A} + z\mathbf{1}\mathbf{b}^T)$ como $det(I - z\mathcal{A})$ son polinomios en z de grado menor o igual que s. Veamos que de hecho $R(z)$ es el cociente de dos polinomios de grado exactamente s. Para ello, probamos el siguiente teorema.

Teorema 3.2 *La función de estabilidad $R(z)$ de un método de colocación basado en los nodos $c_1, \ldots, c_s$ viene dada por*

$$R(z) = \frac{P(z)}{Q(z)},$$

donde $P(z)$ y $Q(z)$ tienen la siguiente forma

$$\begin{aligned} P(z) &= M^{(s)}(1) + M^{(s-1)}(1)z + \cdots + M(1)z^s, \\ Q(z) &= M^{(s)}(0) + M^{(s-1)}(0)z + \cdots + M(0)z^s, \end{aligned}$$

con $M(x) = \dfrac{1}{s!}\displaystyle\prod_{i=1}^{s}(x - c_i)$, y donde $M^{(k)}$ denota la derivada k-ésima de M, $1 \leq k \leq s$.

Demostración. Sean $z = \lambda h$ y $u(x)$ el polinomio de colocación. Como $u'(x) - zu(x)$ es un polinomio de grado s que se anula en los nodos de colocación c_i, existe una constante K_0 tal que

$$u'(x) - zu(x) = K_0 \cdot M(x).$$

Denotando por D el operador de derivación, reescribimos esta expresión:

$$\begin{aligned} (D - z)u(x) &= K_0 \cdot M(x) \\ (1 - \frac{D}{z})u(x) &= -\frac{1}{z} K_0 \cdot M(x) \end{aligned}$$

Despejando $u(x)$ de esta última fórmula se obtiene

$$u(x) = -\left(1 - \frac{D}{z}\right)^{-1} \frac{K_0}{z} \cdot M(x).$$

Desarrollando el término $(1 - D/z)^{-1}$ en su forma de serie geométrica, obtenemos

$$u(x) = -\frac{K_0}{z}\left(1 + \frac{D}{z} + \cdots + \frac{D^s}{z^s}\right) M(x),$$

ya que $D^j M(x) = 0$ para $j > s$. A partir de esta expresión, y dada la relación $u(1) = R(z)u(0)$, obtenemos la expresión buscada de $R(z)$

$$\begin{aligned} R(z) &= \frac{u(1)}{u(0)} = \frac{\left(1 + \frac{D}{z} + \cdots + \frac{D^s}{z^s}\right) M(1)}{\left(1 + \frac{D}{z} + \cdots + \frac{D^s}{z^s}\right) M(0)} \\ &= \frac{M(1)z^s + M'(1)z^{s-1} + \cdots + M^{(s)}(1)}{M(0)z^s + M'(0)z^{s-1} + \cdots + M^{(s)}(0)}. \end{aligned}$$

□

En el caso de los métodos de Gauss, $M(x)$ es el polinomio de Legendre de grado s trasladado al intervalo $[0, 1]$, salvo un factor de escala que afecta por igual al numerador y el denominador. Sabemos, por lo visto en la Sección 2.2, que

$$M(0) = (-1)^s M(1) \;\; y \;\; M(1) = 1,$$

por lo que los polinomios $P(z)$ y $Q(z)$ del Teorema 3.2 son de grado exactamente s.

Su cociente, $R(z)$, proporciona una aproximación racional a e^z de orden $2s$, luego se trata del aproximante de Padé (s, s) de la exponencial. La A-estabilidad de los métodos de Gauss de cualquier número de etapas se deduce entonces del Teorema 4.12 de [4].

3.3. B-estabilidad y estabilidad algebraica

También nos interesa el estudio de la estabilidad y convergencia para sistemas no lineales; es decir, queremos generalizar la A-estabilidad.

Consideramos la ecuación diferencial no lineal

$$\mathbf{x}'(t) = \mathbf{f}(t, \mathbf{x}(t)), \tag{3.5}$$

y suponemos que cumple la *condición de Lipschitz unilateral*

$$\langle \mathbf{f}(t, \mathbf{x}) - \mathbf{f}(t, \mathbf{z}), \mathbf{x} - \mathbf{z} \rangle \leq \nu ||\mathbf{x} - \mathbf{z}||^2, \tag{3.6}$$

para la norma euclídea. La constante ν se denomina *constante de Lipschitz unilateral* de $\mathbf{f}$. Cuando $\nu \leq 0$ en (3.6), la distancia entre dos soluciones de (3.5) con distinta condición inicial es una función no creciente de t. Lógicamente, esta propiedad es deseable también para las soluciones numéricas.

Definición 3.3 *Un método Runge-Kutta se dice B-estable si la condición contractiva*

$$\langle \mathbf{f}(t, \mathbf{x}) - \mathbf{f}(t, \mathbf{z}), \mathbf{x} - \mathbf{z} \rangle \leq 0 \tag{3.7}$$

implica, para todo $h \geq 0$*,*

$$||\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1}|| \leq ||\mathbf{x}_n - \hat{\mathbf{x}}_n||, \tag{3.8}$$

donde $\mathbf{x}_{n+1}$ *y* $\hat{\mathbf{x}}_{n+1}$ *son las aproximaciones numéricas tras un paso a las soluciones de (3.5) con valores iniciales* $\mathbf{x}_n$ *y* $\hat{\mathbf{x}}_n$*, respectivamente.*

Es claro que la B-estabilidad implica la A-estabilidad. Para ello veamos que, tomando $z \in \mathbb{C}^-$, se cumple $|R(z)| \leq 1$. Podemos suponer que $z = \lambda h$, con $\lambda = |\lambda| e^{i\theta}$, y $Re(\lambda) \leq 0$, $h \geq 0$. Primero, comprobemos que se cumple la condición (3.7):

$$\begin{aligned} \langle \mathbf{f}(t, \mathbf{x}) - \mathbf{f}(t, \mathbf{z}), \mathbf{x} - \mathbf{z} \rangle &= \langle \lambda (\mathbf{x} - \mathbf{z}), \mathbf{x} - \mathbf{z} \rangle = |\lambda| \langle e^{i\theta}(\mathbf{x} - \mathbf{z}), \mathbf{x} - \mathbf{z} \rangle \\ &= |\lambda| \cdot ||\mathbf{x} - \mathbf{z}||^2 \cdot cos(\theta). \end{aligned}$$

Si λ es real (no positivo) está claro que la condición se cumple. Si λ es complejo con parte real negativa o nula, también, pues $Re(\lambda) = |\lambda| cos(\theta) \leq 0$.

Como suponemos B-estabilidad y se da la condición contractiva (3.7), entonces se tiene (3.8). Además, se cumplen las igualdades

$$\begin{aligned} \mathbf{x}_{n+1} &= R(\lambda h)\mathbf{x}_n, \\ \hat{\mathbf{x}}_{n+1} &= R(\lambda h)\hat{\mathbf{x}}_n. \end{aligned}$$

Por lo tanto, si las restamos y tomamos normas, obtenemos

$$||\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1}|| = |R(\lambda h)| \cdot ||\mathbf{x}_n - \hat{\mathbf{x}}_n|| \leq ||\mathbf{x}_n - \hat{\mathbf{x}}_n||.$$

La única forma en que la última desigualdad se cumpla es que $|R(\lambda h)| \leq 1$, por lo que concluimos el resultado deseado.

Probemos ahora que los métodos de Gauss son B-estables.

Teorema 3.3 *Los métodos de colocación basados en la cuadratura gaussiana son B-estables.*

Demostración. Denotamos por $\mathbf{u}(t)$ y $\hat{\mathbf{u}}(t)$ a los polinomios de colocación con valores iniciales $\mathbf{x}_0$ y $\hat{\mathbf{x}}_0$ respectivamente. Derivamos la función $m(t) = ||\mathbf{u}(t) - \hat{\mathbf{u}}(t)||^2$, y evaluando en los nodos de colocación $\xi_i = t_n + c_i h$ obtenemos

$$m'(\xi_i) = 2\langle \mathbf{f}(\xi_i, \hat{\mathbf{u}}(\xi_i)) - \mathbf{f}(\xi_i, \hat{\mathbf{u}}(\xi_i)), \hat{\mathbf{u}}(\xi_i) - \hat{\mathbf{u}}(\xi_i)\rangle \leq 0.$$

El resultado, entonces, se deriva de que la cuadratura gaussiana es exacta para $m'(t)$ (que es de grado $2s-1$), y de que los pesos b_i son todos positivos:

$$\begin{aligned} ||\mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1}|| &= m(t_n + h) = m(t_n) + \int_{t_n}^{t_n+h} m'(t)dt \\ &= m(t_n) + h\sum_{i=1}^{s} b_i m'(\xi_i) \leq m(t_n) = ||\mathbf{x}_n - \hat{\mathbf{x}}_n||^2. \end{aligned}$$

□

Veamos ahora un criterio algebraico para la B-estabilidad.

Teorema 3.4 *Si los coeficientes del método Runge-Kutta (1.5)-(1.6) satisfacen las dos condiciones siguientes*

(i) $b_i \geq 0$ *para* $1 \leq i \leq s$,

(ii) $M = (m_{ij})_{i,j=1}^{s} = (b_i a_{ij} + b_j a_{ji} - b_i b_j)_{i,j=1}^{s}$ *es semidefinida positiva,*

entonces el método es B-estable.

A los métodos Runge-Kutta que satisfacen las propiedades (i) y (ii) se les llama algebraicamente estables.

Demostración. Tomamos las diferencias

$$\begin{aligned} \Delta\mathbf{x}_n &= \mathbf{x}_n - \hat{\mathbf{x}}_n, \quad \Delta\mathbf{x}_{n+1} = \mathbf{x}_{n+1} - \hat{\mathbf{x}}_{n+1}, \quad \Delta\mathbf{X}_i = \mathbf{X}_i - \hat{\mathbf{X}}_i \\ \Delta\mathbf{f}_i &= h\left(\mathbf{f}(t_n + c_i h, \mathbf{X}_i) - \mathbf{f}(t_n + c_i h, \hat{\mathbf{X}}_i)\right), \end{aligned}$$

y restamos las fórmulas del método (1.5)-(1.6) para $\mathbf{x}$ y $\hat{\mathbf{x}}$

$$\Delta\mathbf{x}_{n+1} = \Delta\mathbf{x}_n + \sum_{i=1}^{s} b_i \Delta\mathbf{f}_i, \tag{3.9}$$

$$\Delta\mathbf{X}_i = \Delta\mathbf{x}_n + \sum_{j=1}^{s} a_{ij} \Delta\mathbf{f}_j. \tag{3.10}$$

A continuación, calculamos el cuadrado de la norma euclídea de $\Delta\mathbf{x}_{n+1}$, utilizando (3.9):

$$||\Delta\mathbf{x}_{n+1}||^2 = ||\Delta\mathbf{x}_n||^2 + 2\sum_{i=1}^{s} b_i \langle \Delta\mathbf{f}_i, \Delta\mathbf{x}_n\rangle + \sum_{i=1}^{s}\sum_{j=1}^{s} b_i b_j \langle \Delta\mathbf{f}_i, \Delta\mathbf{f}_j\rangle.$$

Sustituyendo $\Delta\mathbf{x}_n$ por su valor al despejarlo en (3.10), nos lleva a

$$\begin{aligned}||\Delta\mathbf{x}_{n+1}||^2 &= ||\Delta\mathbf{x}_n||^2 + 2\sum_{i=1}^{s} b_i\langle\Delta\mathbf{f}_i, \Delta\mathbf{X}_i\rangle - 2\sum_{i,j=1}^{s} b_i a_{ij}\langle\Delta\mathbf{f}_i, \Delta\mathbf{f}_j\rangle + \sum_{i,j=1}^{s} b_i b_j\langle\Delta\mathbf{f}_i\Delta\mathbf{f}_j\rangle \\ &= ||\Delta\mathbf{x}_n||^2 + 2\sum_{i=1}^{s} b_i\langle\Delta\mathbf{f}_i, \Delta\mathbf{X}_i\rangle - \sum_{i,j=1}^{s} m_{ij}\langle\Delta\mathbf{f}_i, \Delta\mathbf{f}_j\rangle.\end{aligned}$$

El segundo sumando de (3.11) es ≤ 0 por (3.7). Veamos que el término que aparece restando en (3.11) es no negativo. Para ello, probaremos que dada una matriz simétrica M de tamaño $s \times s$, M es definida positiva si y solo si $\sum_{i,j=1}^{s} m_{ij}\langle\mathbf{u}_i, \mathbf{u}_j\rangle \geq 0$ para todo $\{\mathbf{u}_1, \ldots, \mathbf{u}_s\} \subset \mathbb{R}^n$.

Como M es simétrica, diagonaliza ortogonalmente $M = Q^T DQ$, con $Q = (q_{ij})_{i,j=1}^{s}$ ortogonal y $D = (d_{ii})_{i=1}^{s}$ diagonal. Entonces, se tiene $m_{ij} = \sum_{k=1}^{s} q_{ki} d_{kk} q_{kj}$, y por lo tanto

$$\begin{aligned}\sum_{i,j=1}^{s} m_{ij}\langle\mathbf{u}_i, \mathbf{u}_j\rangle &= \sum_{i,j,k=1}^{s} d_{kk} q_{ki} q_{kj}\langle\mathbf{u}_i, \mathbf{u}_j\rangle = \sum_{i,j,k=1}^{s} d_{kk}\langle q_{ki}\mathbf{u}_i, q_{kj}\mathbf{u}_j\rangle \\ &= \sum_{k=1}^{s} d_{kk}\langle\sum_{i=1}^{s} q_{ki}\mathbf{u}_i, \sum_{j=1}^{s} q_{kj}\mathbf{u}_j\rangle = \sum_{k=1}^{s} d_{kk}||\sum_{i=1}^{s} q_{ki}\mathbf{u}_i||^2 \\ &= \sum_{k=1}^{s} d_{kk}||\mathbf{v}_k||^2,\end{aligned}$$

donde $\mathbf{v}_k = \sum_{i=1}^{s} q_{ki}\mathbf{u}_i$ o, equivalentemente, $[\mathbf{v}_1, \ldots, \mathbf{v}_s] = [\mathbf{u}_1, \ldots, \mathbf{u}_s] \cdot Q^T$.

$\Rightarrow$) Si M es semidefinida positiva, entonces $d_{kk} \geq 0$, $1 \leq k \leq s$, y, en consecuencia,

$$\sum_{i,j=1}^{s} m_{ij}\langle\mathbf{u}_i, \mathbf{u}_j\rangle = \sum_{k=1}^{s} d_{kk}||\mathbf{v}_k||^2 \geq 0.$$

$\Leftarrow$) Si $\sum_{i,j=1}^{s} m_{ij}\langle\mathbf{u}_i, \mathbf{u}_j\rangle \geq 0$ para todo $\{\mathbf{u}_1, \ldots, \mathbf{u}_s\} \subset \mathbb{R}^n$, entonces también $\sum_{k=1}^{s} d_{kk}||\mathbf{v}_k||^2 \geq 0$ para todo $\{\mathbf{v}_1, \ldots, \mathbf{v}_s\} \subset \mathbb{R}^n$. Razonamos ahora por reducción al absurdo: si hubiera algún $d_{jj} < 0$, tomando $\mathbf{v}_j = \mathbf{e}_j \in \mathbb{R}^n$ (el vector coordenado j-ésimo), $\mathbf{v}_l = \mathbf{0}$ si $l \neq j$, obtendríamos vectores $\mathbf{u}_1, \ldots, \mathbf{u}_s$, definidos por $[\mathbf{u}_1, \ldots, \mathbf{u}_s] = [\mathbf{v}_1, \ldots, \mathbf{v}_s] \cdot Q$, para los que

$$\sum_{i,j=1}^{s} m_{ij}\langle\mathbf{u}_i, \mathbf{u}_j\rangle = \sum_{k=1}^{s} d_{kk}||\mathbf{v}_k||^2 = d_{jj} < 0,$$

lo cual es absurdo por hipótesis.

Estamos ahora en condiciones de volver a (3.11) y concluir que

$$||\Delta \mathbf{x}_{n+1}||^2 \leq ||\Delta \mathbf{x}_n||^2,$$

lo que implica la B-estabilidad de todo método algebraicamente estable. □

Para finalizar este capítulo, vamos a probar que los métodos de Gauss son algebraicamente estables.

Teorema 3.5 *Los métodos de Gauss son algebraicamente estables. Es más, la matriz M cuyo elemento (i,j) viene dado por*

$$m_{ij} = b_i a_{ij} + b_j a_{ji} - b_i b_j, \qquad 1 \leq i,j \leq s,$$

es la idénticamente nula.

Demostración Vamos a probar que se cumplen las condiciones (i) y (ii) del Teorema 3.4:

(i) Sean $c_1, \ldots, c_s$ las raíces del polinomio de Legendre trasladado al $[0,1]$, y consideramos $f_i(x) = \prod_{j \neq i}^{s} (x - c_j)^2 \geq 0$. El grado de este polinomio es $2s-2$, luego la fórmula de cuadratura gaussiana lo integra exactamente.

$$0 \leq \int_0^1 f_i(x)dx = \sum_{j=1}^{s} b_j f_i(c_j) = b_i f_i(c_i) \geq 0, \qquad 1 \leq i \leq s,$$

donde la última igualdad se deduce de la forma del polinomio f_i. Por lo tanto, como $f_i(c_i) > 0$, se deduce que $b_i \geq 0, \;\; 1 \leq i \leq s$.

(ii) Sea V la matriz de Vandermonde de (2.8), y sea $\tilde{M} = VMV^T$. Vamos a demostrar que la matriz $\tilde{M}$ es la idénticamente nula: el elemento (i,j) de la matriz VM es

$$(VM)_{ij} = (c_1^{i-1}, \ldots, c_s^{i-1}) \cdot \begin{pmatrix} b_1 a_{1j} + b_j a_{j1} - b_1 b_j \\ \vdots \\ b_s a_{sj} + b_j a_{js} - b_s b_j \end{pmatrix} = \sum_{k=1}^{s} c_k^{i-1} (b_k a_{kj} + b_j a_{jk} - b_k b_j).$$

Por lo tanto, el elemento (i,j) de la matriz $\tilde{M}$ es

$$(\tilde{M})_{ij} = [(VM)V^T]_{ij} = \sum_{k,l=1}^{s} c_l^{j-1} c_k^{i-1} (b_k a_{kl} + b_l a_{lk} - b_l b_k).$$

Operando con esta expresión, y haciendo uso de que los métodos de Gauss satisfacen

$B(2s)$, $C(s)$ y $D(s)$, llegamos a

$$\begin{aligned}
(\tilde{M})_{ij} &= \sum_{l=1}^{s} c_l^{j-1} \left(\sum_{k=1}^{s} b_k c_k^{i-1} a_{kl} \right) + \sum_{k=1}^{s} c_k^{i-1} \left(\sum_{l=1}^{s} b_l c_l^{j-1} a_{lk} \right) - \sum_{k,l=1}^{s} b_k c_k^{i-1} b_l c_l^{j-1} \\
&= \sum_{l=1}^{s} c_l^{j-1} \frac{b_l}{i} (1 - c_l^i) + \sum_{k=1}^{s} c_k^{i-1} \frac{b_k}{j} (1 - c_k^j) - \frac{1}{i} \frac{1}{j} \\
&= \frac{1}{i} \sum_{l=1}^{s} \left(b_l c_l^{j-1} - b_l c_l^{i+j-1} \right) + \frac{1}{j} \sum_{k=1}^{s} \left(b_k c_k^{i-1} - b_k c_k^{i+j-1} \right) - \frac{1}{ij} \\
&= \frac{1}{i} \left(\frac{1}{j} - \frac{1}{i+j} \right) + \frac{1}{j} \left(\frac{1}{i} - \frac{1}{i+j} \right) - \frac{1}{ij} = \frac{i+j-(i+j)}{ij(i+j)} = 0,
\end{aligned}$$

para todo $1 \leq i, j \leq s$, luego $\tilde{M} \equiv 0$. Por consiguiente, $M = V^{-1} \tilde{M} (V^T)^{-1} \equiv 0$, como queríamos. □

Capítulo 4

Implementación y resultados numéricos

Ahora que ya hemos definido los métodos de Gauss y hemos hablado de sus propiedades, vamos a analizar su comportamiento al integrar un problema concreto. Pero necesitamos estudiar antes algunos aspectos relacionados con la implementación eficiente de los métodos Runge-Kutta implícitos.

4.1. Reformulación de las ecuaciones

Para resolver (1.5)-(1.6) en la práctica, es aconsejable considerar los incrementos

$$\mathbf{Z}_i = \mathbf{X}_i - \mathbf{x}_n, \qquad 1 \leq i \leq s,$$

para así reducir la influencia de los errores de redondeo. Entonces (1.5) se convierte en:

$$\mathbf{Z}_i = h \sum_{j=1}^{s} a_{ij} \mathbf{f}(t_n + c_j h, \mathbf{x}_n + \mathbf{Z}_j), \qquad 1 \leq i \leq s. \tag{4.1}$$

Cuando ya se conoce una solución $(\mathbf{Z}_1^T, \ldots, \mathbf{Z}_s^T)^T$ del sistema de ecuaciones (4.1), se puede hallar $\mathbf{x}_{n+1}$ mediante aplicación directa de (1.6), pero requiere s evaluaciones adicionales de $\mathbf{f}$. Dichas evaluaciones adicionales pueden evitarse si la matriz de coeficientes del método, $\mathcal{A}$, es invertible, como sucede en el caso de los métodos de Gauss. De hecho, (4.1) puede reescribirse como

$$\begin{pmatrix} \mathbf{Z}_1 \\ \vdots \\ \mathbf{Z}_s \end{pmatrix} = h \cdot (\mathcal{A} \otimes I) \begin{pmatrix} \mathbf{f}(t_n + c_1 h, \mathbf{x}_n + \mathbf{Z}_1) \\ \vdots \\ \mathbf{f}(t_n + c_s h, \mathbf{x}_n + \mathbf{Z}_s) \end{pmatrix},$$

donde I denota la matriz identidad $D \times D$, y $\mathcal{A} \otimes I$ denota la matriz de tamaño $sD \times sD$

$$\mathcal{A} \otimes I = \begin{pmatrix} a_{11} I & \cdots & a_{1s} I \\ \cdots & \cdots & \cdots \\ a_{s1} I & \cdots & a_{ss} I \end{pmatrix}.$$

Así, $\mathbf{x}_{n+1}$ se puede obtener mediante

$$\mathbf{x}_{n+1} = \mathbf{x}_n + \sum_{i=1}^{s} d_i \mathbf{Z}_i,$$

con

$$(d_1, \ldots, d_s) = (b_1, \ldots, b_s)\mathcal{A}^{-1}.$$

4.2. Métodos iterativos para las ecuaciones no lineales

Para resolver el sistema de ecuaciones no lineales (4.1) y obtener una aproximación a su solución $(\mathbf{Z}_1^T, \ldots, \mathbf{Z}_s^T)^T$, debemos usar un método iterativo. Para nuestros experimentos, usaremos la iteración de punto fijo y el método iterativo de Newton.

La iteración de punto fijo es el método más directo para resolver el sistema que se nos presenta. Siendo ν el índice de la iteración tendríamos, para $\nu = 0, 1, 2, \ldots,$

$$\mathbf{Z}_i^{[\nu+1]} = h \sum_{j=1}^{s} a_{ij} \mathbf{f}(t_n + c_j h, \mathbf{x}_n + \mathbf{Z}_j^{[\nu]}), \qquad 1 \leq i \leq s,$$

partiendo del iterante inicial $\mathbf{Z}_i^{[0]} = \mathbf{0}$, $1 \leq i \leq s$, que resulta adecuado puesto que a la vista de (4.1) es claro que $\mathbf{Z}_i = \mathcal{O}(h)$. Cada iteración requiere s evaluaciones de $\mathbf{f}$, y la iteración se detiene si $||\mathbf{Z}^{[\nu+1]} - \mathbf{Z}^{[\nu]}||_\infty < TOL$, siendo $\mathbf{Z}^{[\nu]} = (\mathbf{Z}_1^{[\nu]T}, \ldots, \mathbf{Z}_s^{[\nu]T})^T$ la ν-ésima aproximación a la solución de (4.1), y TOL una tolerancia prefijada.

Sin embargo, esta iteración convierte el algoritmo en un método explícito y estropea las buenas propiedades de estabilidad que tienen los métodos de Gauss por su carácter implícito.

El método iterativo de Newton aporta grandes ventajas en este sentido. Aplicando este método, tendremos que resolver en cada iteración el sistema lineal con matriz

$$\begin{pmatrix} I - ha_{11}\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n + c_1 h, \mathbf{x}_n + \mathbf{Z}_1^{[\nu]}) & \cdots & -ha_{1s}\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n + c_s h, \mathbf{x}_n + \mathbf{Z}_s^{[\nu]}) \\ \vdots & \ddots & \vdots \\ -ha_{s1}\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n + c_1 h, \mathbf{x}_n + \mathbf{Z}_1^{[\nu]}) & \cdots & I - ha_{ss}\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n + c_s h, \mathbf{x}_n + \mathbf{Z}_s^{[\nu]}) \end{pmatrix}.$$

Para evitar tantas evaluaciones de la función $\mathbf{f}$ y de sus derivadas respecto de $\mathbf{x}$, sustituiremos todas las matrices jacobianas $\dfrac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n + c_i h, \mathbf{x}_n + \mathbf{Z}_i)$ por una aproximación

$$J \approx \frac{\partial \mathbf{f}}{\partial \mathbf{x}}(t_n, \mathbf{x}_n),$$

que se calculará al principio de cada paso y que reduce notablemente el coste computacional del método iterativo. Así, el método de Newton modificado aplicado a (4.1) se convierte en

$$\begin{aligned} (I - h(\mathcal{A} \otimes J))\Delta \mathbf{Z}^{[\nu]} &= -\mathbf{Z}^{[\nu]} + h(\mathcal{A} \otimes I) \cdot \mathbf{F}(\mathbf{Z}^{[\nu]}), \\ \mathbf{Z}^{[\nu+1]} &= \mathbf{Z}^{[\nu]} + \Delta \mathbf{Z}^{[\nu]}, \end{aligned}$$

donde $\mathbf{F}(\mathbf{Z}^{[\nu]})$ es el vector

$$\mathbf{F}(\mathbf{Z}^{[\nu]}) = (\mathbf{f}(t_n + c_1 h, \mathbf{x}_n + \mathbf{Z}_1^{[\nu]})^T, \ldots, \mathbf{f}(t_n + c_s h, \mathbf{x}_n + \mathbf{Z}_s^{[\nu]})^T)^T.$$

Cada iteración requiere s evaluaciones de $\mathbf{f}$ y la solución de un sistema lineal de tamaño sD. Como la matriz $(I - h(\mathcal{A} \otimes J))$ es la misma para cada iteración, su evaluación y su factorización LU se hacen una vez por paso, al principio, para reducir el coste operativo.

Una elección natural para los iterantes iniciales $\mathbf{Z}_i^{[0]}$ es de nuevo

$$\mathbf{Z}_i^{[0]} = \mathbf{0}, \quad 1 \leq i \leq s,$$

ya que la solución exacta de (4.1) satisface $\mathbf{Z}_i = \mathcal{O}(h)$. Para detener la iteración pediremos que $||\Delta\mathbf{Z}^{[\nu]}||_\infty$ sea menor que una tolerancia prescrita por el usuario, dado que $\Delta\mathbf{Z}^{[\nu]} = \mathbf{Z}^{[\nu+1]} - \mathbf{Z}^{[\nu]}$.

4.3. Elección de la tolerancia

Para estudiar el efecto que tiene la elección de la tolerancia al detener en cada paso la iteración elegida para resolver el sistema no lineal, vamos a realizar algunos experimentos numéricos. Para ello se considera el problema de Kepler, en su formulación como sistema de primer orden

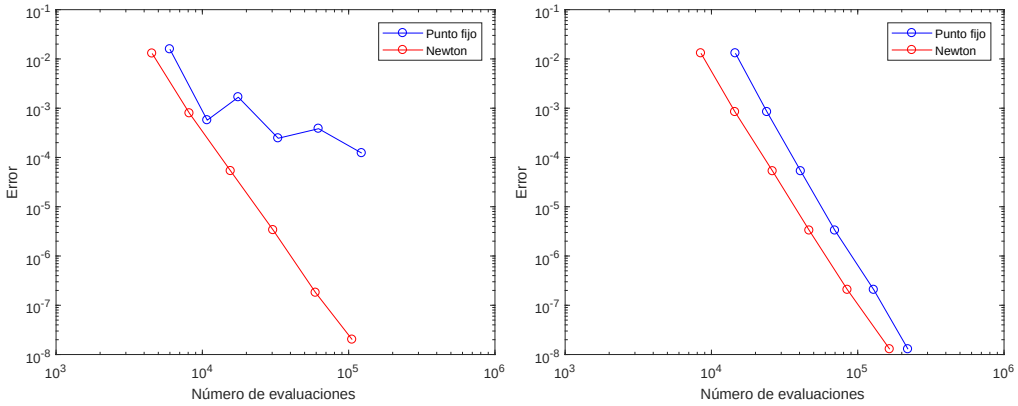
$$\mathbf{x}'(t) = \left[x_3(t), x_4(t), \frac{-x_1(t)}{(\sqrt{x_1(t)^2 + x_2(t)^2})^3}, \frac{-x_2(t)}{(\sqrt{x_1(t)^2 + x_2(t)^2})^3} \right]^T, \tag{4.2}$$

con condiciones iniciales

$$\mathbf{x}(0) = \left[1 - e, 0, 0, \sqrt{\frac{1+e}{1-e}} \right]^T. \tag{4.3}$$

Su solución corresponde a una órbita 2π-periódica de excentricidad e en el plano (x_1, x_2). Se quiere aproximar numéricamente la solución de (4.2)-(4.3) en tiempos de la forma $T = N \times 2\pi$, para distintos valores del número de periodos, N, y de la excentricidad e. Para obtener los resultados y posteriores conclusiones, debemos fijar el valor de tres parámetros: la excentricidad e, la tolerancia para detener la iteración TOL, y el número de periodos durante los que se integra N.

El primer experimento que llevaremos a cabo consiste en integrar nuestro problema con tolerancias 10^{-6} y 10^{-15}, para ver si podemos determinar cuál es mejor. Realizaremos diversas gráficas a escala doblemente logarítmica correspondientes a los resultados obtenidos con dichas tolerancias, excentricidad $e = 0.5$ y longitudes de paso $h = \frac{2\pi}{64}, \frac{2\pi}{128}, \ldots, \frac{2\pi}{2048}$, reflejando el error global cometido frente al número de evaluaciones de la función $\mathbf{f}$. Integraremos el problema durante 10 periodos, pues para otros valores de N se observa un comportamiento análogo, con la salvedad de que cuanto mayor es el número de periodos durante los que integramos el problema, mayor será el error total.



(a) $TOL = 10^{-6}$ (b) $TOL = 10^{-15}$

Figura 4.1: Error frente al número de evaluaciones de $\mathbf{f}$ utilizando el método de Gauss de orden 4 implementado con iteración de punto fijo y de Newton.

Con la tolerancia 10^{-6} (la gráfica izquierda de la Figura 4.1), a partir de $h = \dfrac{2\pi}{256}$, $h^4 < TOL$, por lo que el error procedente del integrador temporal se ve contaminado por el error procedente de la iteración utilizada para resolver el sistema no lineal. Esto hace que no se aprecie el verdadero comportamiento del integrador numérico. Además, podemos observar que, con dicha tolerancia, los dos métodos iterativos se comportan de manera muy diferente. Implementado con iteración de punto fijo, el error acaba estancándose en valores de tamaño 10^{-4}, mientras que con el método iterativo de Newton los errores descienden hasta 10^{-8}. Este comportamiento es debido a que cuando el error ronda el valor 10^{-5}, todavía por encima de la tolerancia requerida, se realiza una iteración más, y entonces la convergencia cuadrática del método de Newton hace que el error tras esa nueva iteración sea de tamaño 10^{-10}. Tras 10 periodos, dicho error se acumula hasta el valor que se observa en la gráfica correspondiente.

Con $TOL = 10^{-15}$ (la gráfica derecha de la Figura 4.1) no se observa ninguna contaminación debida a los errores ocasionados por los métodos iterativos, ya que el error global del método de Gauss está por encima de esta tolerancia (permanece alrededor de 10^{-8} con la longitud de paso más pequeña). Ésta es la razón de que la gráfica de la derecha muestre que, cuando $h \to 0$, el error del integrador temporal se comporta como $\mathcal{O}(h^4)$, con independencia del método iterativo utilizado. Sin embargo, conlleva un mayor número de evaluaciones, al ser la tolerancia tan exigente, como se puede ver al comparar las correspondientes líneas en ambas gráficas. De hecho, para ambas iteraciones se observa casi un factor de 2 entre el número de evaluaciones necesarias con $TOL = 10^{-15}$ y el requerido con $TOL = 10^{-6}$.

A la vista de estos resultados podemos concluir que la tolerancia 10^{-15} es preferible, pues a costa de un mayor número de evaluaciones, permite observar toda la precisión que proporciona el método Runge-Kutta. No obstante, no debemos conformarnos con esto. En este experimento solo hemos considerado dos tolerancias fijas, una más grosera y otra más exigente. Sin embargo, también cabe la posibilidad de utilizar una tolerancia

variable. El error esperado en la integración temporal es $\mathcal{O}(h^p)$, donde p es el orden del método utilizado, lo que nos sugiere escoger una tolerancia que se comporte como $\mathcal{O}(h^p)$ pero que sea ligeramente menor, por ejemplo, $h^p \times 10^{-2}$.

Hemos realizado de nuevo la integración de nuestro problema, esta vez con tolerancia $TOL = h^p \times 10^{-2}$, y hemos comparado los nuevos resultados obtenidos con los que obtuvimos con $TOL = 10^{-15}$, considerando el número de iteraciones promedio y el error al final de la integración, cuando dividimos repetidamente las longitudes del paso de integración a la mitad, partiendo de $2\pi/64$. Estos resultados pueden observarse en las Tablas 4.1 y 4.2.

h	$TOL = 10^{-15}$		$TOL = h^4 \times 10^{-2}$	
	Iter. promedio	Error	Iter. promedio	Error
$2\pi/64$	11.4	1.304×10^{-2}	4.7	1.573×10^{-2}
$2\pi/128$	9.4	8.374×10^{-4}	4.7	8.571×10^{-4}
$2\pi/256$	8.0	5.268×10^{-5}	4.6	5.258×10^{-5}
$2\pi/512$	6.9	3.298×10^{-6}	4.6	3.281×10^{-6}
$2\pi/1024$	6.3	2.063×10^{-7}	4.6	2.052×10^{-7}
$2\pi/2048$	5.4	1.282×10^{-8}	4.5	1.277×10^{-8}

Tabla 4.1: Error tras 10 periodos y promedio de iteraciones realizadas utilizando el método de Gauss de orden 4 con iteración de punto fijo y distintas longitudes de paso h.

h	$TOL = 10^{-15}$		$TOL = h^4 \times 10^{-2}$	
	Iter. promedio	Error	Iter. promedio	Error
$2\pi/64$	6.6	1.304×10^{-2}	3.5	1.291×10^{-2}
$2\pi/128$	5.7	8.374×10^{-4}	3.5	8.206×10^{-4}
$2\pi/256$	5.1	5.268×10^{-5}	3.5	5.228×10^{-5}
$2\pi/512$	4.6	3.298×10^{-6}	3.5	3.295×10^{-6}
$2\pi/1024$	4.2	2.064×10^{-7}	3.5	2.062×10^{-7}
$2\pi/2048$	4.0	1.282×10^{-8}	3.5	1.282×10^{-8}

Tabla 4.2: Error tras 10 periodos y promedio de iteraciones realizadas utilizando el método de Gauss de orden 4 con iteración de Newton y distintas longitudes de paso h.

Centrándonos en los datos recogidos en la Tabla 4.1, que corresponden a la utilización de iteración de punto fijo, los resultados son claros. Con la tolerancia variable se necesita un número de iteraciones promedio menor y casi constante e independiente de h, mientras que los errores (con ambas tolerancias) son casi idénticos y exhiben el orden esperado.

De la Tabla 4.2, correspondiente al método de Newton, extraemos las mismas conclusiones. Comparando los datos de ambas tablas, es inmediato concluir que, al menos en el ejemplo que estamos considerando, y atendiendo únicamente al número de iteraciones promedio (y, por tanto, al número de evaluaciones de función), el método de Gauss de

orden 4 es mejor con el método de Newton que con la iteración de punto fijo, pues necesita menos iteraciones promedio y los errores son comparables. Sin embargo, como se indicará más adelante, el costo operativo del método de Newton no se limita al número de evaluaciones de función, puesto que requiere además la evaluación del jacobiano y la resolución de sistemas lineales.

El único motivo para elegir una tolerancia fija más exigente sería reducir el error obtenido, pero la comparativa realizada desecha dicha posibilidad. Por lo tanto, la tolerancia que emplearemos en nuestros futuros experimentos, será la adaptativa, $h^p \times 10^{-2}$.

Sin embargo, cuando vayamos reduciendo h cada vez más, llegará un momento en el que $h^p \times 10^{-2}$ será menor que 10^{-15}. Una tolerancia tan pequeña no nos interesa, ya que no será apreciable trabajando en doble precisión, y puede incluso impedir que el método converja por la contribución de los errores de redondeo. Por consiguiente, la tolerancia óptima será $\max(h^p \times 10^{-2}, 10^{-15})$.

Todo este razonamiento ha sido realizado a través de los datos obtenidos usando el método de Gauss de orden 4. Los resultados obtenidos utilizando el método de Gauss de orden 8 son similares, y las conclusiones que se pueden extraer de ellos, las mismas. Por lo tanto, obviaremos dichos resultados en esta memoria.

4.4. Resultados numéricos

Ahora que ya hemos elegido la tolerancia que usaremos, vamos a comparar entre sí los métodos iterativos utilizados, así como ambos integradores numéricos (el método de Gauss de orden 4 y el de orden 8).

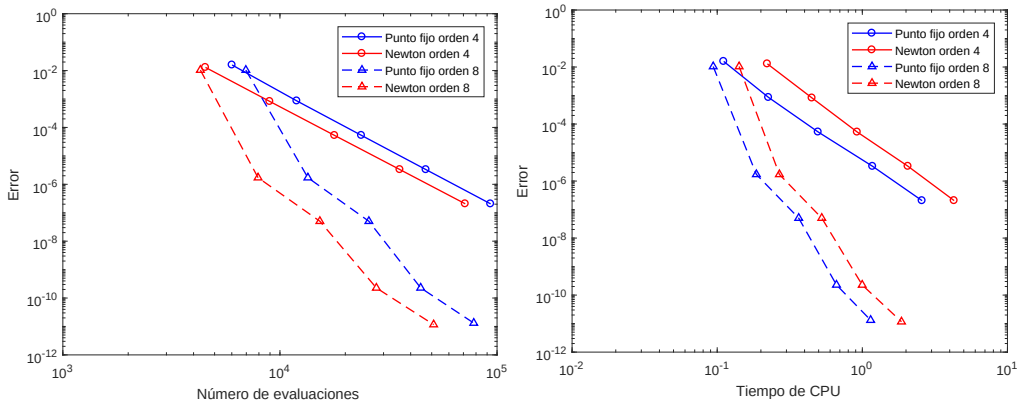


Figura 4.2: Error frente al número de evaluaciones de $\mathbf{f}$ y el tiempo de CPU utilizando los métodos de Gauss de orden 4 y 8, implementados con iteración de punto fijo y de Newton, con excentricidad $e = 0.5$.

En la Figura 4.2 se muestran dos gráficas que hemos realizado con tolerancia $\max(h^p \times 10^{-2}, 10^{-15})$, $N = 10$ y $e = 0.5$ para comparar los métodos de Gauss de 2 y 4 etapas combinados con los dos métodos iterativos con los que trabajamos en esta memoria. En

la gráfica de la izquierda se ha representado error frente al número de evaluaciones de función, mientras que en la gráfica de la derecha se muestra error frente a tiempo de CPU. Se ha utilizado el color azul para indicar la iteración de punto fijo, y el rojo para la iteración de Newton. Asimismo, las líneas continuas corresponden al método de Gauss de orden 4, mientras que las discontinuas representan datos generados con el método de orden 8.

En la gráfica del lado izquierdo en la Figura 4.2 podemos apreciar, al igual que en las Tablas 4.1 y 4.2, que el método iterativo de Newton conlleva un menor número de evaluaciones de la función $\mathbf{f}$ puesto que las líneas de color rojo permanecen siempre a la izquierda de las correspondientes líneas de color azul. Sin embargo, en la gráfica de la derecha, observamos que la iteración de punto fijo requiere un menor tiempo de CPU. También se aprecia el orden de cada método en la pendiente de las líneas representadas en ambas gráficas: las dos líneas que representan datos generados con el método de orden 4 (con ambos métodos iterativos) muestran pendiente -4, mientras que las líneas discontinuas muestran pendiente aproximadamente -8, como corresponde a un método de orden 8. Veamos si dicha pendiente se distingue más fácilmente si disminuimos la excentricidad hasta $e = 0.3$.

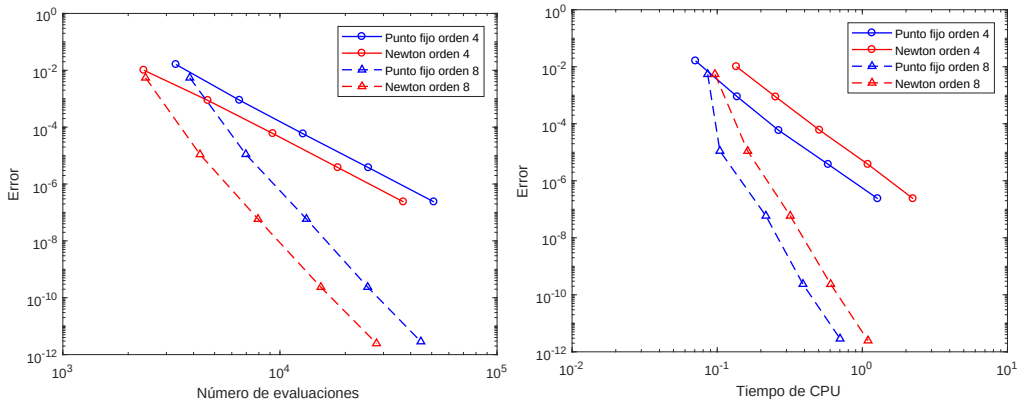


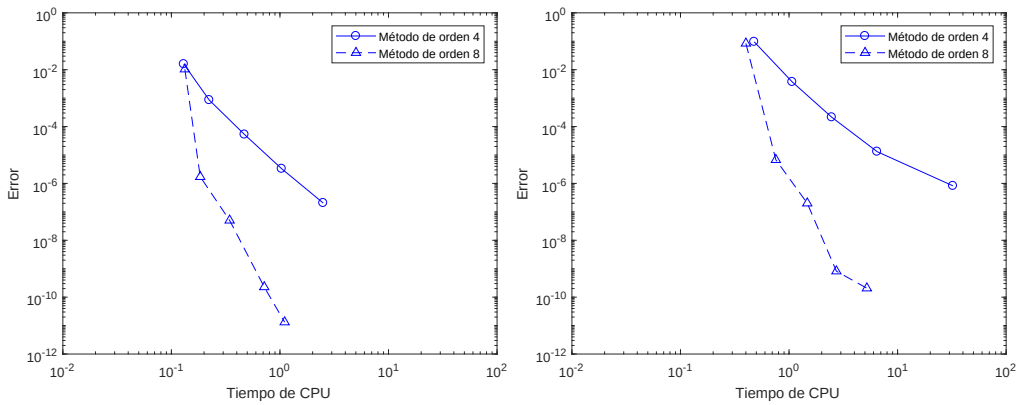
Figura 4.3: Error frente al número de evaluaciones de $\mathbf{f}$ y el tiempo de CPU utilizando los métodos de Gauss de orden 4 y 8, implementados con iteración de punto fijo y de Newton, con excentricidad $e = 0.3$.

El motivo de relajar la excentricidad es facilitar el trabajo del integrador numérico, que al estar implementado con paso fijo, permite longitudes mayores de paso si la excentricidad de la órbita se reduce. Los resultados obtenidos se han representado en la Figura 4.3. Las conclusiones al analizar estos nuevos datos son las mismas, pero los resultados referentes al método de Gauss de orden 8 han cambiado ligeramente dada la menor excentricidad: los puntos representados se disponen de manera más alineada y la pendiente, de valor -8, se aprecia con mayor facilidad.

Atendiendo únicamente a estos resultados, no sabríamos decidir qué método iterativo es mejor para resolver nuestro problema, ya que las conclusiones que se extraen de las gráficas de la izquierda son, aparentemente, opuestas a las que se derivan de las

gráficas del lado derecho. No obstante, debemos tener en cuenta qué es lo que estamos midiendo exactamente en cada una de ellas. Respecto al número de evaluaciones, estamos contando estrictamente el número de evaluaciones de la función **f**, sin embargo, con el método iterativo de Newton es necesario resolver, en cada iteración, un sistema de ecuaciones de tamaño $D \times s$, además de que requiere calcular la matriz jacobiana de la función **f**. Aunque cuando se comparan métodos Runge-Kutta explícitos se suele medir el costo computacional atendiendo al número de evaluaciones de función, cuando se trata de métodos implícitos es preciso comparar los tiempos de CPU. Por tanto, en nuestro ejemplo, podemos concluir de las Figuras 4.2 y 4.3 que la iteración de punto fijo es la más eficiente tanto para el método de orden 4 como para el de orden 8.

A continuación, vamos a mostrar la influencia que tienen en el error y en el costo computacional el número de periodos durante los que integramos, así como la excentricidad elegida. Realizaremos nuevas gráficas, integrando nuestro problema con la tolerancia adaptativa que venimos considerando y la iteración de punto fijo (la implementación que ha resultado más eficiente), haciendo variar las excentricidades y el número de periodos.

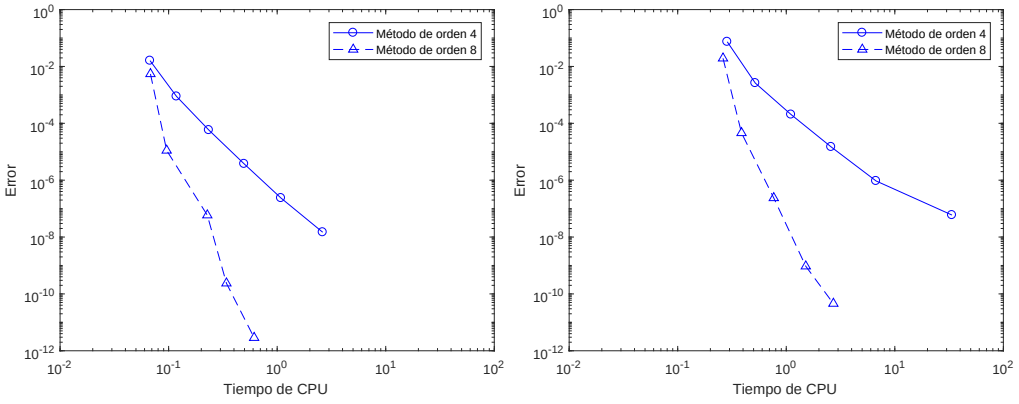


(a) $N = 10$, $e = 0.5$ (b) $N = 40$, $e = 0.5$

Figura 4.4: Error frente al tiempo de CPU utilizando los métodos de Gauss de 2 y 4 etapas implementados con iteración de punto fijo y excentricidad $e = 0.5$.

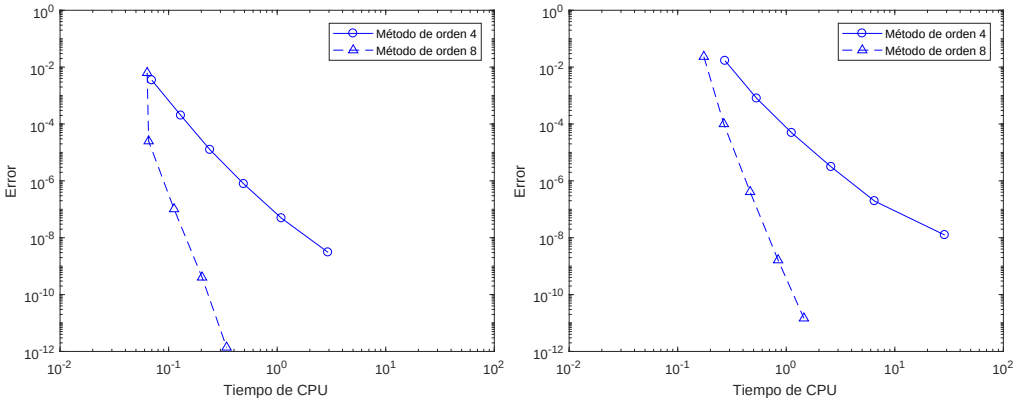
En las Figuras 4.4, 4.5 y 4.6 se han incluido 6 gráficas que muestran error al final de la integración frente al tiempo de CPU requerido. Las gráficas de la izquierda corresponden a la integración durante $N = 10$ periodos mientras que las de la derecha se han realizado con $N = 40$ periodos. Como en las gráficas anteriores se ha utilizado línea continua para el método de orden 4 y discontinua para el de orden 8. La Figura 4.4 muestra los resultados obtenidos para excentricidad $e = 0.5$, la Figura 4.5 corresponde a excentricidad $e = 0.3$, y la Figura 4.6 se ha generado con $e = 0.1$.

Está claro que el método de orden 8 produce resultados considerablemente mejores en todos los casos: errores más bajos y tiempos de CPU notablemente menores, llegando a dividirse por un factor de 10 cuando se compara con el método de orden 4 para obtener errores de tamaño 10^{-6}. Además, disminuir la excentricidad nos permite visualizar mejor



(a) $N = 10$, $e = 0.3$ (b) $N = 40$, $e = 0.3$

Figura 4.5: Error frente al tiempo de CPU utilizando los métodos de Gauss de 2 y 4 etapas implementados con iteración de punto fijo y excentricidad $e = 0.3$.



(a) $N = 10$, $e = 0.1$ (b) $N = 40$, $e = 0.1$

Figura 4.6: Error frente al tiempo de CPU utilizando los métodos de Gauss de 2 y 4 etapas implementados con iteración de punto fijo y excentricidad $e = 0.1$.

el orden de los métodos y los tiempos de CPU necesarios disminuyen (las correspondientes líneas se desplazan hacia la izquierda a medida que disminuye e).

Como los errores producidos por el método de Gauss de orden 8 decrecen a mayor velocidad al reducir h que los correspondientes al método de orden 4, las longitudes de paso con las que hemos integrado nuestro problema no son las mismas con ambos métodos. De hecho existe un factor de al menos 4 entre unas y otras. Por ejemplo, en la gráfica izquierda de la Figura 4.4, los errores mayores (de tamaño 10^{-2}) se han generado utilizando $h = \frac{2\pi}{64}$ con el método de orden 4 y $h = \frac{2\pi}{16}$ con el método de orden 8. Para excentricidad $e = 0.1$, errores del mismo tamaño se consiguen con $h = \frac{2\pi}{32}$ con el método de Gauss de 2 etapas y $h = \frac{2\pi}{4}$ con el de 4 etapas.

Por último, si para cada una de las excentricidades consideradas comparamos los errores que se obtienen para $N = 10$ y $N = 40$, observamos que para ambos métodos los errores se multiplican aproximadamente por 16 (4^2), mientras que el tiempo de CPU sólo se multiplica por 4 (en cada figura las líneas de la gráfica derecha están desplazadas hacia la derecha y ligeramente hacia arriba respecto de la posición que tienen en la gráfica izquierda). Sigue siendo, por tanto, más eficiente el método de Gauss de 4 etapas.

Bibliografía

[1] J. C. Butcher, Numerical Methods for Ordinary Differential Equations, Wiley, 2003.

[2] F. Galindo, J. Sanz & L. A. Tristán, Guía Práctica de Cálculo Infinitesimal en Varias Variables, Paraninfo, 2005.

[3] E. Hairer, S.P. Nørsett & G. Wanner, Solving Ordinary Differential Equations I, Springer, 1993.

[4] E. Hairer & G. Wanner, Solving Ordinary Differential Equations II, Springer, 1996.

Apéndice A

Propiedades de los polinomios de Legendre

En este apéndice nos dedicaremos a demostrar algunas propiedades de los polinomios de Legendre en $[0,1]$ utilizados en el Capítulo 2. Para demostrar algunas de dichas propiedades haremos uso de la fórmula recursiva de Bonnet

$$\begin{cases} \tilde{P}_s(x) &= \dfrac{2s-1}{s}(2x-1)\tilde{P}_{s-1}(x) - \dfrac{s-1}{s}\tilde{P}_{s-2}(x), \qquad s \geq 2, \\ \tilde{P}_0(x) &= 1, \quad \tilde{P}_1(x) = 2x-1. \end{cases}$$

1. $\displaystyle\int_0^1 \tilde{P}_s(x)\tilde{P}_t(x)dx = 0$ si $s \neq t$:

   Para probarlo, usaremos que $\tilde{P}_s(x)$ satisface la ecuación diferencial siguiente

   $$8[(x-x^2)\tilde{P}_s'(x)]' + s(s+1)\tilde{P}_s = 0.$$

   Si $\tilde{P}_s(x)$ y $\tilde{P}_t(x)$ son soluciones de la ecuación anterior, multiplicando la ecuación correspondiente a $\tilde{P}_s(x)$ por $\tilde{P}_t(x)$ y viceversa, y restándolas, obtenemos

   $$8\int_0^1 \left(\tilde{P}_t[(x-x^2)\tilde{P}_s']' - \tilde{P}_s[(x-x^2)\tilde{P}_t']'\right)dx + [s(s+1)-t(t+1)]\int_0^1 \tilde{P}_s\tilde{P}_t dx = 0.$$

   La primera integral vale 0, puesto que integrando por partes el primer término de esta relación, se convierte en:

   $$[\tilde{P}_t(x-x^2)\tilde{P}_s']_0^1 - [\tilde{P}_s(x-x^2)\tilde{P}_t']_0^1 - \int_0^1 \left(\tilde{P}_t'(x-x^2)\tilde{P}_s' - \tilde{P}_s'(x-x^2)\tilde{P}_t'\right)dx,$$

   que claramente vale 0 puesto que, por un lado, el factor $(x-x^2)$ se anula en $x=0$ y $x=1$ y, por otro, el integrando es idénticamente nulo. Por lo tanto,

   $$[s(s+1)-t(t+1)]\int_0^1 \tilde{P}_s(x)\tilde{P}_t(x)dx = 0,$$

   y, si $s \neq t$, se concluye que la integral es 0.

2. $\int_0^1 \tilde{P}_s(x)^2 dx = \frac{1}{2s+1}$:

Haremos uso de la fórmula recursiva que hemos introducido al principio de este apéndice:

$$\begin{aligned}\int_0^1 \tilde{P}_s(x)^2 dx &= \int_0^1 \tilde{P}_s(x)\left[\frac{2s-1}{s}(2x-1)\tilde{P}_{s-1}(x) - \frac{s-1}{s}\tilde{P}_{s-2}(x)\right]dx \\ &= \frac{2s-1}{s}\int_0^1 (2x-1)\tilde{P}_s(x)\tilde{P}_{s-1}(x)dx - \frac{s-1}{s}\int_0^1 \tilde{P}_s(x)\tilde{P}_{s-2}(x)dx.\end{aligned}$$

El segundo término es nulo por la ortogonalidad de los polinomios de Legendre. Ahora, para deshacernos del término $(2x-1)\tilde{P}_s(x)\tilde{P}_{s-1}(x)$ de la primera integral, utilizamos de nuevo la fórmula recursiva, pero esta vez para los índices $s+1, s$ y $s-1$, y despejaremos el término $(2x-1)\tilde{P}_s(x)$ para sustituirlo en la igualdad de arriba.

$$\begin{aligned}\int_0^1 \tilde{P}_s(x)^2 dx &= \frac{2s-1}{s}\int_0^1 (2x-1)\tilde{P}_s(x)\tilde{P}_{s-1}(x)dx - \frac{s-1}{s}\int_0^1 \tilde{P}_s(x)\tilde{P}_{s-2}(x)dx \\ &= \frac{2s-1}{s}\left[\frac{s+1}{2s+1}\int_0^1 \tilde{P}_{s+1}(x)\tilde{P}_{s-1}(x)dx + \frac{s}{2s+1}\int_0^1 \tilde{P}_{s-1}(x)^2 dx\right] \\ &= \frac{2s-1}{2s+1}\int_0^1 \tilde{P}_{s-1}(x)^2 dx.\end{aligned}$$

Llamando $g(s) = \int_0^1 \tilde{P}_s(x)^2 dx$, hemos llegado a la siguiente recurrencia

$$g(s) = \frac{2s-1}{2s+1}g(s-1).$$

Vamos a probar por inducción que $g(s) = \frac{1}{2s+1}$, partiendo de que $g(0) = 1$, $g(1) = 3$. Aplicamos ahora la hipótesis de inducción: suponemos que $g(s-1) = \frac{1}{2s-1}$, y tenemos que

$$g(s) = \frac{2s-1}{2s+1}g(s-1) = \frac{2s-1}{2s+1}\cdot\frac{1}{2s-1} = \frac{1}{2s+1}.$$

3. $\tilde{P}_s(1) = 1$:

$$\begin{aligned}&\tilde{P}_0(1) = 1, \qquad \tilde{P}_1(1) = 1, \\ &\tilde{P}_s(1) = \frac{2s-1}{s}\tilde{P}_{s-1}(1) - \frac{s-1}{s}\tilde{P}_{s-2}(1) = \frac{2s-1}{s} - \frac{s-1}{s} = \frac{s}{s} = 1.\end{aligned}$$

4. $\tilde{P}_s(1-x) = (-1)^s\tilde{P}_s(x)$:

$$\tilde{P}_0(1-x) = 1 = \tilde{P}_0(x), \qquad \tilde{P}_1(1-x) = 2(1-x) - 1 = 1 - 2x = -\tilde{P}_1(x),$$

$$\begin{aligned} P_s(1-x) &= \frac{2s-1}{s}(1-2x)\tilde{P}_{s-1}(1-x) - \frac{s-1}{s}\tilde{P}_{s-2}(1-x) \\ &= \frac{2s-1}{s}(1-2x)(-1)^{s-1}\tilde{P}_{s-1}(x) - \frac{s-1}{s}(-1)^{s-2}\tilde{P}_{s-2}(x) \\ &= \frac{2s-1}{s}(2x-1)(-1)^{s}\tilde{P}_{s-1}(x) - \frac{s-1}{s}(-1)^{s}\tilde{P}_{s-2}(x) \\ &= (-1)^s\tilde{P}_s(x). \end{aligned}$$

5. $\tilde{P}_s(x)$ tiene s raíces reales distintas en el intervalo $(0,1)$:

   Razonamos por reducción al absurdo, suponiendo que $\tilde{P}_s(x) = Q(x)R(x)$, siendo Q y R de grados m y $s-m$, con $m < s$, respectivamente. Supongamos también que R no tiene raíces en $(0,1)$. Entonces, por la ortogonalidad de los polinomios de Legendre, $0 = \int_0^1 \tilde{P}_s(x)Q(x)dx = \int_0^1 Q(x)^2R(x)dx$, aunque el integrando es no nulo y tiene signo constante, lo cual es absurdo.

Apéndice B

Funciones en Matlab

Este segundo apéndice contiene las funciones que hemos programado en Matlab y con las que se han generado las gráficas del Capítulo 4, implementando los métodos de Gauss de orden 4 y 8 tanto con iteración de punto fijo como con iteración de Newton.

B.1. El problema de Kepler

Para empezar, hemos implementado todo lo necesario para integrar el problema de Kepler que utilizamos como problema test en esta memoria. Incluimos en esta sección las funciones `kepler`, `kepler_jac` e `inicial`, que proporcionan el lado derecho del sistema de Kepler (4.2), el jacobiano de dicha función, y la condición inicial del problema, respectivamente.

```
function f=kepler(t,x)
    r=sqrt(x(1)^2+x(2)^2)^3;
    f=[x(3);x(4);-x(1)/r;-x(2)/r];
end
```

```
function J=kepler_jac(t,x)
    r=x(1)^2+x(2)^2;

    J(1,:)=[0, 0, 1, 0];
    J(2,:)=[0, 0, 0, 1];
    J(3,:)=[(3*x(1)^2 - r)/r^(5/2),3*x(2)^2/r^(5/2), 0, 0];
    J(4,:)=[3*x(1)^2/r^(5/2), (3*x(2)^2 - r)/r^(5/2), 0, 0];
end
```

```
function x0=inicial(e)
    %Esta funcion generara la condicion inicial para cada excentricidad
    x0=[1-e;0;0;sqrt((1+e)/(1-e))];
end
```

B.2. Los métodos de Gauss

Con la función **gauss4pf** calculamos una aproximación a la solución de (4.2)-(4.3) mediante el método de Gauss de orden 4 implementado con iteración de punto fijo.

```
function [tt,xx,neval,niter,npasos,error,cpu] = gauss4pf(t0,tF,x0,h,TOL,f)

    %Inicializamos
     D=length(x0);
     tt=t0; xx=x0';                 %Mas adelante los iremos ampliando.
     neval=0; niter=0; npasos=0;
     xn=x0; tn=t0;

    %El metodo
     a=[1/4,1/4-sqrt(3)/6;1/4+sqrt(3)/6,1/4];
     b=[1/2,1/2];
     c=[1/2 - sqrt(3)/6; 1/2 + sqrt(3)/6];
     d=b/a;   %( b/a <=> b*inv(a) )
     s=length(b);

    %Empezamos
     F=zeros(D,s);      %La columna j sera el vector f(tn+c_j*h,xn+Z_j).
     tic
     while tn<tF
         if tn+h>tF
             h=tF-tn;
         end
         Z0=ones(D,s);      %La columna j sera el vector Z_j.
         Z1=zeros(D,s);
         while (max(max(abs(Z1-Z0)))>TOL && niter<15)
             Z0=Z1;
             for j=1:s
                 F(:,j)=f(tn+c(j)*h,xn+Z0(:,j));
             end
             neval=neval+s;
             for j=1:s
                 Z1(:,j)=h*(F*a(j,:)');
             end
             niter=niter+1;
         end
         xn=xn+Z1*d';
         tn=tn+h;
         npasos=npasos+1;
         xx=[xx;xn'];
         tt=[tt;tn];
     end
     cpu=toc;
     error=norm(xx(end,:)-x0');
end
```

La **gauss4nw** hace lo mismo, salvo que en este caso la solución del sistema no lineal se calcula con iteración de Newton.

```
function [tt,xx,neval,niter,npasos,error,cpu] = gauss4nw(t0,tF,x0,h,TOL,f,jac)

    %Inicializamos
     D=length(x0);
```

```
    tt=t0; xx=x0';
    neval=0; niter=0; npasos=0;

   %El metodo
    a=[1/4,1/4-sqrt(3)/6;1/4+sqrt(3)/6,1/4];
    b=[1/2,1/2];
    c=[1/2 - sqrt(3)/6; 1/2 + sqrt(3)/6];
    d=b/a;   %( b/a <=> b*inv(a) )
    s=length(b);

    F=zeros(D*s,1);
    FF=zeros(D*s,1);
    xn=x0; tn=t0;

   %Empezamos
    tic
    while tn<tF
        if tn+h>tF
            h=tF-tn;
        end
        Z1=zeros(D*s,1); Delta=TOL+1;
        J=eye(D*s) - h*kron(a,jac(tn,xn));
        [L,U,P]=lu(J);
        while (max(abs(Delta))>TOL && niter<10)
            for i=1:s
                F(D*(i-1)+1:D*i) = f(tn+c(i)*h,xn+Z1(D*(i-1)+1:D*i));
            end
            neval=neval+s;
            for i=1:s
               FF(D*(i-1)+1:D*i) = Z1(D*(i-1)+1:D*i) - h*(a(i,1)*F(1:D)...
               + a(i,2)*F(D+1:2*D));
            end
            y=L\P*FF; Delta=U\y;
            Z1=Z1-Delta;
            niter=niter+1;
        end
        Z1=reshape(Z1,[D,s]);
        xn=xn+Z1*d';
        tn=tn+h;
        npasos=npasos+1;
        xx=[xx;xn'];
        tt=[tt;tn];
    end
    cpu=toc;
    error=norm(xx(end,:)-x0');
end
```

Los dos siguientes programas, **gauss8pf** y **gauss8nw** son los análogos a los dos anteriores, pero esta vez para el método de Gauss de orden 8.

```
function [tt,xx,neval,niter,npasos,error,cpu] = gauss8pf(t0,tF,x0,h,TOL,f)

   %Inicializamos
    D=length(x0);
    tt=t0; xx=x0';
    neval=0; niter=0; npasos=0;
    xn=x0; tn=t0;

   %El metodo
```

```
    w1=1/8 - sqrt(30)/144;           W1=1/8 + sqrt(30)/144;
    w2=sqrt((15+2*sqrt(30))/35)/2;   W2=sqrt((15-2*sqrt(30))/35)/2;
    w3=w2*(1/6 + sqrt(30)/24);       W3=W2*(1/6 - sqrt(30)/24);
    w4=w2*(1/21 + 5*sqrt(30)/168);   W4=W2*(1/21 - 5*sqrt(30)/168);
    w5=w2-2*w3;                      W5=W2-2*W3;

    a=[   w1   , W1-w3+W4, W1-w3-W4,  w1-w5;
       w1-W3+w4,    W1   ,  W1-W5  , w1-W3-w4;
       w1+W3+w4,  W1+W5  ,    W1   , w1+W3-w4;
        w1+w5  , W1+w3+W4, W1+w3-W4,    w1];
    b=[2*w1, 2*W1, 2*W1, 2*w1];
    c=[1/2-w2; 1/2-W2; 1/2 + W2; 1/2+w2];
    d=b/a;   %( b/a <=> b*inv(a) )
    s=length(b);

   %Empezamos
    F=zeros(D,s);      %La columna j sera el vector f(tn+c_j*h,xn+Z_j).
    tic
    while tn<tF
        if tn+h>tF
            h=tF-tn;
        end
        Z0=ones(D,s);      %La columna j sera el vector Z_j.
        Z1=zeros(D,s);
        while (max(max(abs(Z1-Z0)))>TOL && niter<15)
            Z0=Z1;
            for j=1:s
                F(:,j)=f(tn+c(j)*h,xn+Z0(:,j));
            end
            neval=neval+s;
            for j=1:s
                Z1(:,j)=h*(F*a(j,:)');
            end
            niter=niter+1;
        end
        xn=xn+Z1*d';
        tn=tn+h;
        npasos=npasos+1;
        xx=[xx;xn'];
        tt=[tt;tn];
    end
    cpu=toc;
    error=norm(xx(end,:)-x0');
end


function [tt,xx,neval,niter,npasos,error,cpu] = gauss8nw(t0,tF,x0,h,TOL,f,jac)

   %Inicializamos
    D=length(x0);
    tt=t0; xx=x0';
    neval=0; niter=0; npasos=0;

   %El metodo
    w1=1/8 - sqrt(30)/144;           W1=1/8 + sqrt(30)/144;
    w2=sqrt((15+2*sqrt(30))/35)/2;   W2=sqrt((15-2*sqrt(30))/35)/2;
    w3=w2*(1/6 + sqrt(30)/24);       W3=W2*(1/6 - sqrt(30)/24);
    w4=w2*(1/21 + 5*sqrt(30)/168);   W4=W2*(1/21 - 5*sqrt(30)/168);
    w5=w2-2*w3;                      W5=W2-2*W3;
```

```
    a=[   w1   , W1-w3+W4, W1-w3-W4,  w1-w5;
       w1-W3+w4,    W1   ,  W1-W5  , w1-W3-w4;
       w1+W3+w4,  W1+W5  ,    W1   , w1+W3-w4;
        w1+w5  , W1+w3+W4, W1+w3-W4,    w1];
    b=[2*w1, 2*W1, 2*W1, 2*w1];
    c=[1/2-w2; 1/2-W2; 1/2 + W2; 1/2+w2];
    d=b/a;  %( b/a <=> b*inv(a) )
    s=length(b);

    F=zeros(D*s,1);
    FF=zeros(D*s,1);
    xn=x0; tn=t0;

   %Empezamos
    tic
    while tn<tF
        if tn+h>tF
            h=tF-tn;
        end
        Z1=zeros(D*s,1); Delta=TOL+1;
        J=eye(D*s) - h*kron(a,jac(tn,xn));
        [L,U,P]=lu(J);
        while (max(abs(Delta))>TOL && niter<10)
            for i=1:s
                F(D*(i-1)+1:D*i) = f(tn+c(i)*h,xn+Z1(D*(i-1)+1:D*i));
            end
            neval=neval+s;
            for i=1:s
               FF(D*(i-1)+1:D*i) = Z1(D*(i-1)+1:D*i) - h*(a(i,1)*F(1:D)...
               + a(i,2)*F(D+1:2*D) + a(i,3)*F(2*D+1:3*D)...
               + a(i,4)*F(3*D+1:4*D));
            end
            y=L\P*FF; Delta=U\y;
            Z1=Z1-Delta;
            niter=niter+1;
        end
        Z1=reshape(Z1,[D,s]);
        xn=xn+Z1*d';
        tn=tn+h;
        npasos=npasos+1;
        xx=[xx;xn'];
        tt=[tt;tn];
    end
    cpu=toc;
    error=norm(xx(end,:)-x0');
end
```

B.3. Visualización de resultados

Todos los resultados que hemos mostrado a lo largo de la memoria en forma de tablas y gráficas también han sido obtenidos mediante algunos funciones programadas en Matlab. Mostramos a continuación solo dos de ellas, utilizadas para generar las Tablas 4.1 y 4.2, y las Figuras 4.2 y 4.3. Estas funciones son `tabla_tol` y `grafica_total`, y aunque por sí solas no podrían reproducir toda la información que hemos aportado, son una muestra representativa.

```
function []=tabla_tol(N,e,metodo,f,jac)
    H=zeros(7,1); Niter1=zeros(7,1); Npasos1=zeros(7,1); T=N*2*pi;
    Niter2=zeros(7,1); Npasos2=zeros(7,1);
    Error1=zeros(7,1); Error2=zeros(7,1); h=2*pi/16;

    for i=1:7
        h=h/2; H(i)=h;
        switch metodo
            case 1 %1==punto fijo
                [~,~,~,Niter1(i),Npasos1(i),Error1(i),~] =...
                    gauss4pf(0,T,inicial(e),H(i),1e-15,f);
                [~,~,~,Niter2(i),Npasos2(i),Error2(i),~] =...
                    gauss4pf(0,T,inicial(e),H(i),h^4/100,f);
            case 2 %2==Newton
                [~,~,~,Niter1(i),Npasos1(i),Error1(i),~] =...
                    gauss4nw(0,T,inicial(e),H(i),1e-15,f,jac);
                [~,~,~,Niter2(i),Npasos2(i),Error2(i),~] =...
                    gauss4nw(0,T,inicial(e),H(i),h^4/100,f,jac);
        end
    end

    promedio1=Niter1(2:end)./Npasos1(2:end);
    promedio2=Niter2(2:end)./Npasos2(2:end);

    format short e
    disp('       h       Iteraciones     Error     Iteraciones     Error')
    disp('                 promedio                  promedio')
    disp('                      TOL=1e-15                TOL=h^p/100')
    [H(2:end),promedio1,Error1(2:end),promedio2,Error2(2:end)]
end



function []=grafica_total(N,e,f,jac)
    %N numero de vueltas
    %e excentricidad
    %p orden del metodo

    h=2*pi/16; T=N*2*pi;
    Neval_4pf=zeros(7,1); CPU_4pf=zeros(7,1); Error_4pf=zeros(7,1);
    Neval_4nw=zeros(7,1); CPU_4nw=zeros(7,1); Error_4nw=zeros(7,1);
    Neval_8pf=zeros(7,1); CPU_8pf=zeros(7,1); Error_8pf=zeros(7,1);
    Neval_8nw=zeros(7,1); CPU_8nw=zeros(7,1); Error_8nw=zeros(7,1);


    for i=1:7
        h=h/2; tol4=max(h^4/100,1e-15); tol8=max(h^8/100,1e-15);
        [~,~,Neval_4pf(i),~,~,Error_4pf(i),CPU_4pf(i)] =...
            gauss4pf(0,T,inicial(e),h,tol4,f);
        [~,~,Neval_4nw(i),~,~,Error_4nw(i),CPU_4nw(i)] =...
            gauss4nw(0,T,inicial(e),h,tol4,f,jac);
        [~,~,Neval_8pf(i),~,~,Error_8pf(i),CPU_8pf(i)] =...
            gauss8pf(0,T,inicial(e),4*h,tol8,f);
        [~,~,Neval_8nw(i),~,~,Error_8nw(i),CPU_8nw(i)] =...
            gauss8nw(0,T,inicial(e),4*h,tol8,f,jac);
    end

    figure(1)
        loglog(Neval_4pf(1:end-2),Error_4pf(1:end-2),'bo-',...
            'MarkerSize',5,'LineWidth',1);
```

```
        hold on
        loglog(Neval_4nw(1:end-2),Error_4nw(1:end-2),'ro-',...
            'MarkerSize',5,'LineWidth',1);
        hold on
        loglog(Neval_8pf(1:end-2),Error_8pf(1:end-2),'b^--',...
            'MarkerSize',5,'LineWidth',1);
        hold on
        loglog(Neval_8nw(1:end-2),Error_8nw(1:end-2),'r^--',...
            'MarkerSize',5,'LineWidth',1);
        xlabel('Numero de evaluaciones');
        ylabel('Error');
        legend('Punto fijo orden 4','Newton orden 4',...
            'Punto fijo orden 8','Newton orden 8');
    figure(2)
        loglog(CPU_4pf(1:end-2),Error_4pf(1:end-2),'bo-',...
            'MarkerSize',5,'LineWidth',1);
        hold on
        loglog(CPU_4nw(1:end-2),Error_4nw(1:end-2),'ro-',...
            'MarkerSize',5,'LineWidth',1);
        hold on
        loglog(CPU_8pf(1:end-2),Error_8pf(1:end-2),'b^--',...
            'MarkerSize',5,'LineWidth',1);
        hold on
        loglog(CPU_8nw(1:end-2),Error_8nw(1:end-2),'r^--',...
            'MarkerSize',5,'LineWidth',1);
        xlabel('Tiempo de CPU');
        ylabel('Error');
        legend('Punto fijo orden 4','Newton orden 4',...
            'Punto fijo orden 8','Newton orden 8');
end
```