Introducción a los Métodos de Integración Numérica de Ecuaciones Diferenciales Ordinarias

Ernesto Kofman

Departamento de Control, Facultad de Ciencias Exactas, Ingeniería y Agrimensura
Universidad Nacional de Rosario.
CIFASIS – CONICET

Organización de la Presentación

Sistemas Continuos y Ecuaciones Diferenciales Ordinarias

- En la mayoría de los modelos que se utilizan en Ingeniería las variables evolucionan continuamente en el tiempo.
- Por esto, dichos modelos se suelen denominar Sistemas Continuos.
- Entre estos encontramos principalmente los modelos provenientes distintos dominios de la Física: sistemas mecánicos, termodinámicos, electromagnéticos, hidráulicos, etc.

Cuando estos sistemas tienen parámetros concentrados, las relaciones matemáticas que vinculan la evolución de las distintas variables conducen a Sistemas de Ecuaciones Diferenciales Ordinarias.

Sistemas Continuos y Ecuaciones Diferenciales Ordinarias

Por ejemplo, un sistema-masa–resorte–amortiguamiento muy simple como el de la Figura puede representarse por el sistema de ecuaciones de la derecha:

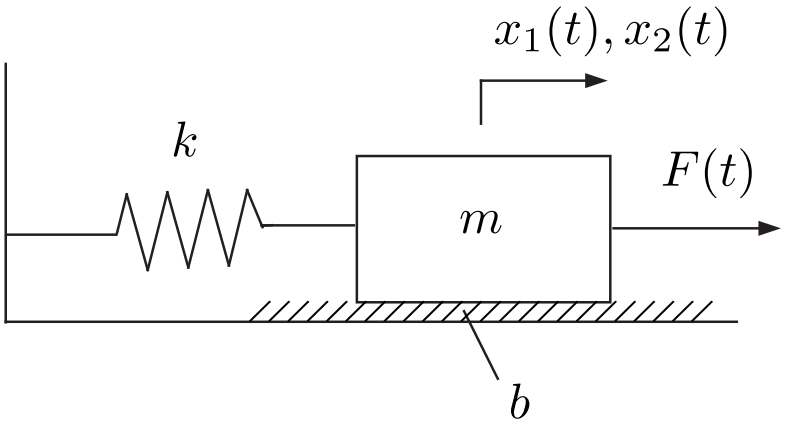

Figura 1.1: Sistema Masa Resorte

$$\dot{x}_1(t) = x_2(t)$$
$$\dot{x}_2(t) = \frac{1}{m}[-k\,x_1(t) - b\,x_2(t) + F(t)] \quad (1.1)$$

Sistemas Continuos y Ecuaciones Diferenciales Ordinarias

- Para determinar la evolución las variables de este sistema en el tiempo, deberemos resolver la Ecuación (1.1).
- Tomando parámetros $m = b = k = 1$, suponiendo que la entrada es constante $F(t) = 1$, y asumiendo que $x_1(0) = x_2(0) = 0$, obtenemos la siguiente solución analítica:

$$\begin{aligned} x_1(t) &= 1 - \frac{\sqrt{3}}{3} e^{-t/2} \sin \frac{\sqrt{3}}{2} t - e^{-t/2} \cos \frac{\sqrt{3}}{2} t \\ x_2(t) &= \frac{\sqrt{12}}{3} e^{-t/2} \sin \frac{\sqrt{3}}{2} t \end{aligned} \tag{1.2}$$

Sistemas Continuos y Ecuaciones Diferenciales Ordinarias

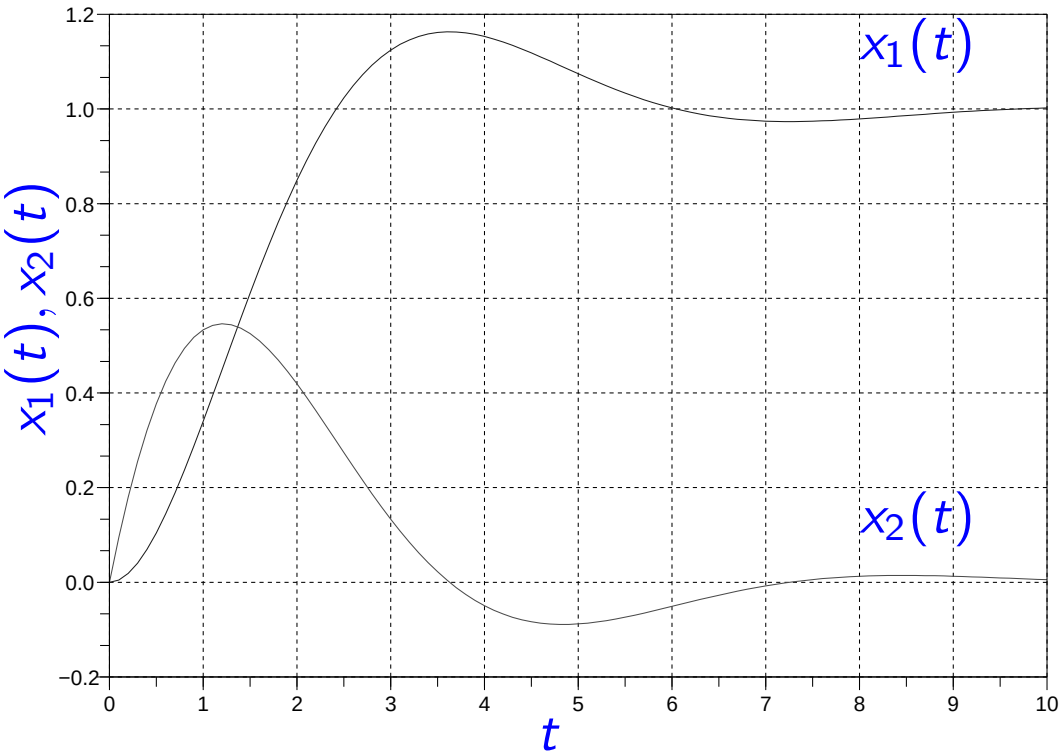

Figura 1.2: Solución de la Ecuación (1.1) del Sistema Masa Resorte

Sistemas Continuos y Ecuaciones Diferenciales Ordinarias

- Si bien en este caso fue posible resolver de manera analítica la EDO, en general esto no será posible.
- En presencia de no linealidades, excepto casos muy simples y triviales, no es posible obtener una expresión analítica de la solución.
- En los casos lineales, aún siendo posible resolver las ecuaciones, es muchas veces engorroso y poco práctico debido a la complejidad de los cálculos y de las expresiones resultantes.
- Por esto, generalmente se recurre a ditintos métodos que permiten obtener soluciones aproximadas, normalmente mediante el uso de computadoras.

Estos algoritmos, denominados métodos de integración numérica para ecuaciones diferenciales ordinarias constituyen la herramienta básica fundamental para la Simulación de Sistemas Continuos.

Organización de la Presentación

Principios Básicos de la Aproximación Numérica

En general, nos interesará resolver un sistema de la forma

$$\begin{aligned}
\dot{x}_1 &= f_1(x_1, x_2, \ldots, x_n, u_1, \cdots, u_m, t) \\
\dot{x}_2 &= f_2(x_1, x_2, \ldots, x_n, u_1, \cdots, u_m, t) \\
&\vdots \\
\dot{x}_n &= f_n(x_1, x_2, \ldots, x_n, u_1, \cdots, u_m, t)
\end{aligned}$$

donde las x_i son las variables de estado y $u_i(t)$ son las entradas. Este sistema puede reescribirse en forma vectorial como sigue:

$$\dot{\mathbf{x}}(t) = \hat{\mathbf{f}}(\mathbf{x}(t), \mathbf{u}(t), t)$$

Principios Básicos de la Aproximación Numérica

Para calcular la solución numérica será necesario conocer $\mathbf{u}(t)$ y no tiene sentido expresarlo como una variable. Por este motivo, en la literatura sobre métodos de integración numérica se define

$$\mathbf{f}(\mathbf{x}(t), t) \triangleq \hat{\mathbf{f}}(\mathbf{x}(t), \mathbf{u}(t), t)$$

En virtud de esto, el objetivo de los métodos de integración numérica será encontrar una solución aproximada del sistema de ecuaciones diferenciales:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), t) \tag{1.3}$$

a partir de la condición inicial conocida

$$\mathbf{x}(t_0) = \mathbf{x}_0 \tag{1.4}$$

Organización de la Presentación

Método de Forward Euler

El método más sencillo para resolver la Ec.(1.3) fue propuesto por Leonhard Euler en 1768. Consiste básicamente en aproximar:

$$\dot{\mathbf{x}}(t) \approx \frac{\mathbf{x}(t+h) - \mathbf{x}(t)}{h} \approx \mathbf{f}(\mathbf{x}(t), t)$$

de donde despejamos

$$\mathbf{x}(t+h) \approx \mathbf{x}(t) + h\, \mathbf{f}(\mathbf{x}(t), t)$$

Llamando entonces $t_k = t_0 + k\ h$ para $k = 0, 1, \cdots$ y definiendo $\mathbf{x}_k \triangleq \mathbf{x}(t_k)$, resulta la fórmula de Forward Euler:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\, \mathbf{f}(\mathbf{x}_k, t_k) \tag{1.5}$$

El parámetro h se denomina paso de integración

Método de Forward Euler

El algoritmo de Forward Euler es entonces trivial:

- Dado $\mathbf{x}_0$ y elegido un valor de h calculamos $\mathbf{x}_1 = \mathbf{x}_0 + h\ \mathbf{f}(\mathbf{x}_0, t_0)$.
- Luego, a partir de $\mathbf{x}_1$ calculamos $\mathbf{x}_2 = \mathbf{x}_1 + h\ \mathbf{f}(\mathbf{x}_1, t_1)$.
- A partir de $\mathbf{x}_1$ calculamos $\mathbf{x}_2$ y así sucesivamente hasta llegar a $t_k = t_f$ (el tiempo final de simulación).

Método de Forward Euler

Para el sistema masa resorte de la Ec.(1.1), el método de Forward Euler con paso $h = 0.1$ comenzaría con los siguientes cálculos:

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{f}(\mathbf{x}_0, t_0) = \begin{bmatrix} x_2 = 0 \\ 1 - x_1 - x_2 = 1 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \mathbf{x}_1 = \mathbf{x}_0 + h\, \mathbf{f}(\mathbf{x}_0, t_0) = \begin{bmatrix} 0 + 0.1 \cdot 0 \\ 0 + 0.1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} \Rightarrow \mathbf{f}(\mathbf{x}_1, t_1) = \begin{bmatrix} x_2 = 0.1 \\ 1 - x_1 - x_2 = 1 - 0 - 0.1 = 0.9 \end{bmatrix} \Rightarrow$$

$$\Rightarrow \mathbf{x}_2 = \mathbf{x}_1 + h\, \mathbf{f}(\mathbf{x}_1, t_1) = \begin{bmatrix} 0 + 0.1 \cdot 0.1 \\ 0.1 + 0.1 \cdot 0.9 \end{bmatrix} = \begin{bmatrix} 0.01 \\ 0.19 \end{bmatrix}$$

Esto es, de acuerdo al método de Forward Euler, $\mathbf{x}(t_1 = 0.1) \approx [0\ 0.1]^T$ y $\mathbf{x}(t_2 = 0.2) \approx [0.01\ 0.19]^T$.

Método de Forward Euler

La Figura muestra el resultado de continuar calculando esta solución numérica y la compara con la analítica.

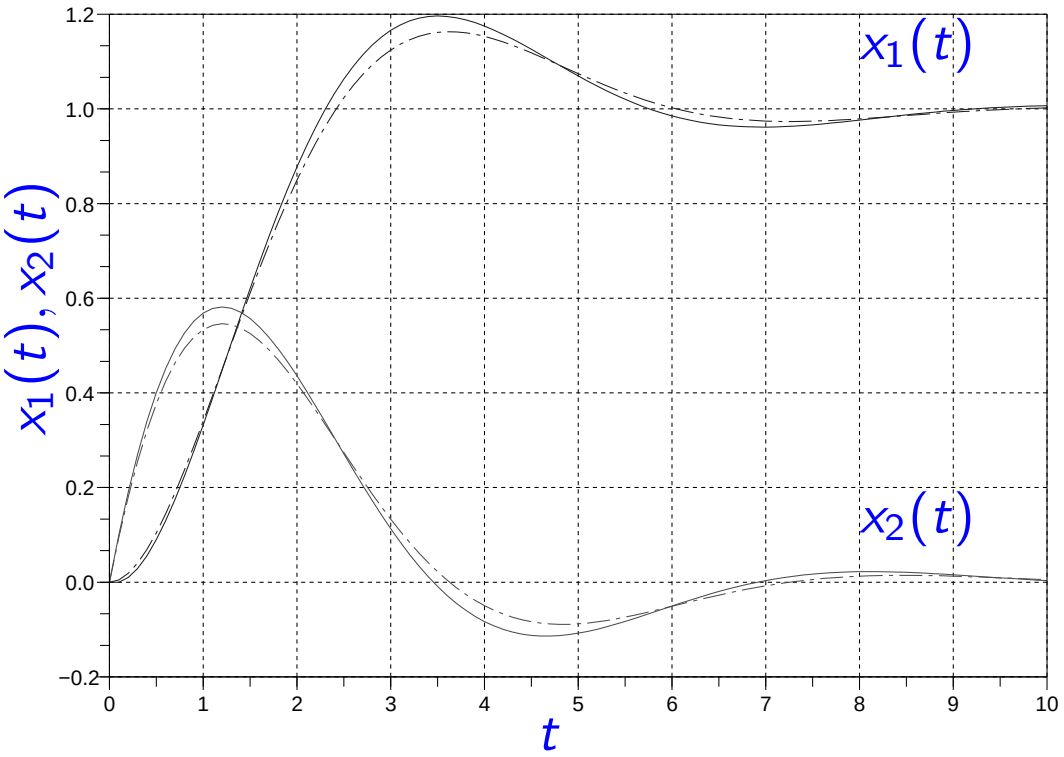

Figura 1.3: Solución de la Ecuación (1.1) del Sistema Masa Resorte con Forward Euler (sólida) con $h = 0.1$ y Solución Analítica (punteada)

Método de Backward Euler

El Método de Backward Euler, reemplaza la fórmula de la Ec.(1.5) por:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,\mathbf{f}(\mathbf{x}_{k+1}, t_{k+1}) \tag{1.6}$$

- Notar que para calcular $\mathbf{x}_{k+1}$ necesitamos conocer $\mathbf{x}_{k+1}$.
- Conociendo $\mathbf{x}_k$ se puede resolver la Ec.(1.6) y obtener de allí el valor de $\mathbf{x}_{k+1}$. Por eso se dice que Backward Euler es un Método Implícito.
- Cuando la función $\mathbf{f}$ es lineal, la resolución de la ecuación implícita es muy sencilla (requiere invertir una matriz).
- En el caso no lineal se usan algoritmos iterativos, generalmente la iteración de Newton.

La implementación práctica de Backward Euler es mucho más compleja y computacionalmente más costosa que la de Forward Euler.

Método de Backward Euler

La Figura muestra el resultado numérico de usar Backward Euler en el sistema masa resorte y lo compara con la solución analítica.

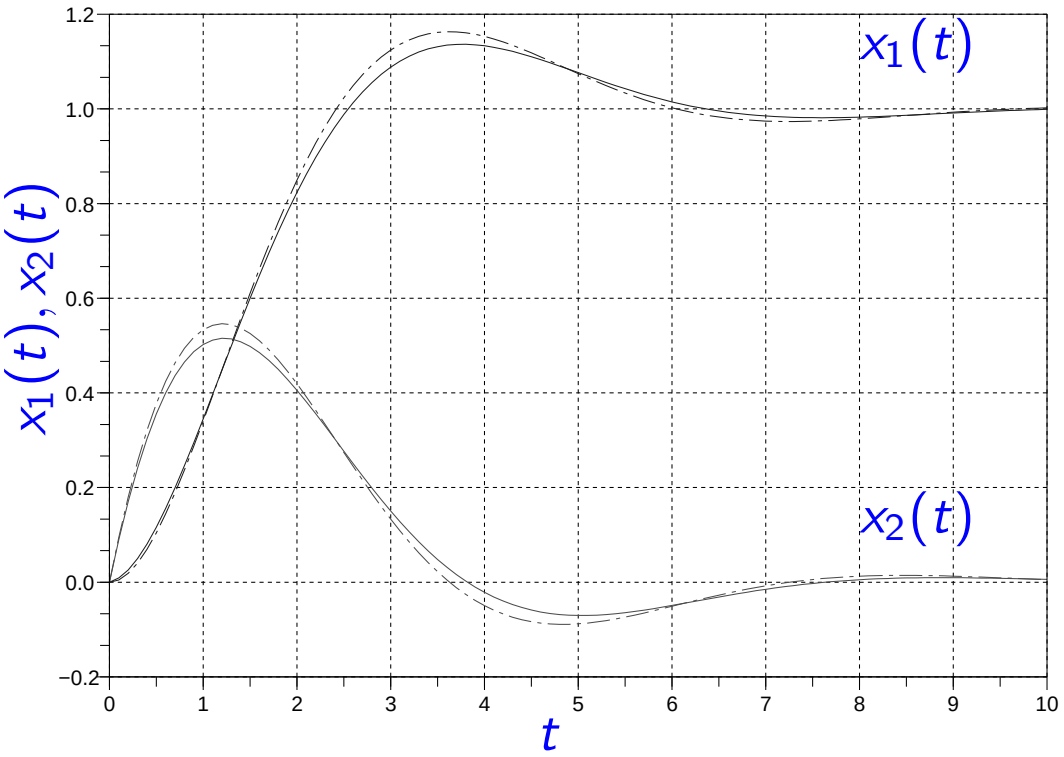

Figura 1.4: Solución de la Ecuación (1.1) del Sistema Masa Resorte con Backward Euler (sólida) con $h = 0.1$ y Solución Analítica (punteada)

Organización de la Presentación

Precisión de las Aproximaciones

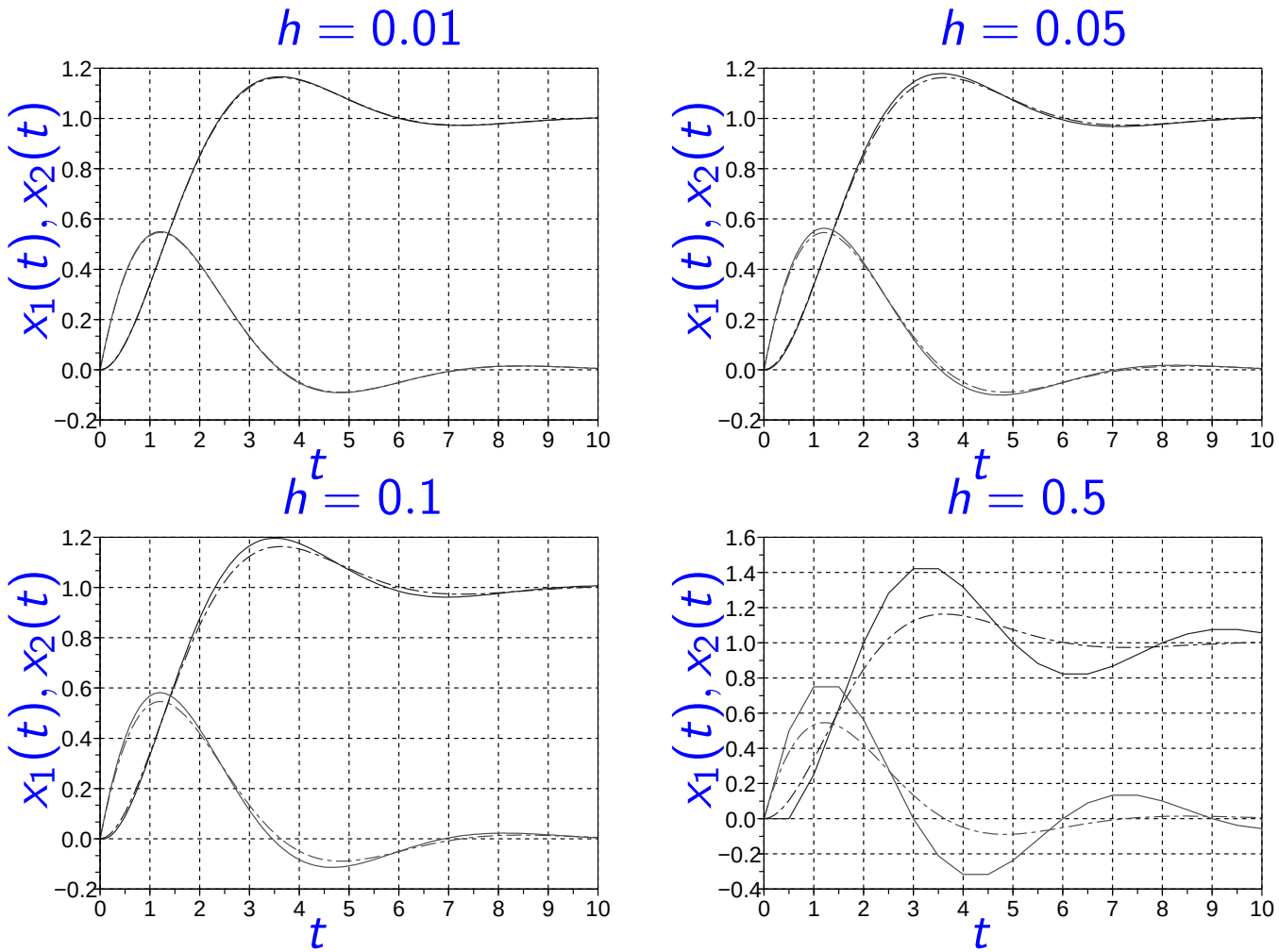

Figura 1.5: Solución de la Ecuación (1.1) con Forward Euler (sólida) para distintos pasos de integración y Solución Analítica (punteada)

Precisión de las Aproximaciones

Dada la solución analítica en un punto $\mathbf{x}(t_k)$, podemos calcula la solución exacta tras un paso de integración de valor h usando la serie de Taylor:

$$\begin{aligned}\mathbf{x}(t_{k+1}) &= \mathbf{x}(t_k) + h\,\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}(t_k) + \frac{h^2}{2!}\,\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}(t_k) + \frac{h^3}{3!}\,\frac{\mathrm{d}^3\mathbf{x}}{\mathrm{d}t^3}(t_k) + \cdots \\ &= \mathbf{x}(t_k) + h\,\mathbf{f}(\mathbf{x}(t_k), t_k) + \frac{h^2}{2!}\,\frac{\mathrm{d}^2\mathbf{x}}{\mathrm{d}t^2}(t_k) + \frac{h^3}{3!}\,\frac{\mathrm{d}^3\mathbf{x}}{\mathrm{d}t^3}(t_k) + \cdots \\ &= \mathbf{x}(t_k) + h\,\mathbf{f}(\mathbf{x}(t_k), t_k) + o(h^2)\end{aligned}$$

El método de Forward Euler utiliza la expansión de Taylor sólo hasta el término de h, ignorando los términos de h^2, h^3, etc.

Dado que el método iguala la serie hasta el término de h^1, se dice que es un algoritmo de primer orden.

Error Local y Error Global

- El error que comete el método de Euler tras el primer paso es del orden de h^2.
- Este error se denomina Error Local por Truncamiento y en general aumenta con el paso h.
- El Error Local por Truncamiento es proporcional a h^{N+1} donde N es el orden del método.
- Salvo en el primer paso, no conoceremos el valor de la solución analítica en t_k. Por lo tanto, en t_{k+1} habrá un error adicional ya que en cada caso estaremos partiendo de un valor inexacto.
- La diferencia que observamos entre la solución analítica y la solución numérica se denomina Error Global.
- Puede demostrarse que el máximo valor que alcanza el error global en un método de orden N es proporcional a h^N.

Precisión y Orden del Método

- En los métodos de Forward y Backward Euler, al ser ambos de primer orden, el máximo error global resulta proporcional al paso h.
- Para realizar una simulación 10000 veces más precisa necesitamos usar un paso 10000 veces más chico, lo que implica realizar 10000 veces más cálculos.
- En un método de cuarto orden, en cambio, disminuir 10 veces el paso de integración implica disminuir 10000 veces el error.
- para realizar una simulación 10000 veces más precisa con un método de cuarto orden necesitamos usar un paso 10 veces más chico, lo que sólo implica realizar 10 veces más cálculos.

Cuando queremos simular con mucha precisión debemos necesariamente utilizar métodos de orden alto.

Errores de Redondeo

- hay aplicaciones en las que no se pueden ignorar los Errores de Redondeo debidos a la representación numérica con un número finito de bits.
- Para evitar estos errores siempre se intenta trabajar con representaciones de punto flotante de 64 bits (doble precisión), lo que permite en general despreciar los efectos del redondeo.
- Una característica interesante de los errores de redondeo es que, a diferencia de los errores por truncamiento, tienden a aumentar a medida que el paso de integración disminuye.

Organización de la Presentación

Estabilidad Numérica

- Una pregunta que surge naturalmente tras analizar el error global de un método es si el mismo se sigue acumulando indefinidamente o no a medida que avanza el tiempo de la simulación.
- Observando nuevamente la Figura 1.5, vemos que a medida que la solución analítica se acerca a su valor final, la solución numérica tiende al mismo valor.
- Es decir, en dicha figura la solución numérica preserva la estabilidad y el punto de equilibrio de la solución analítica.

¿Ocurrirá esto para cualquier paso de integración?.

Estabilidad Numérica

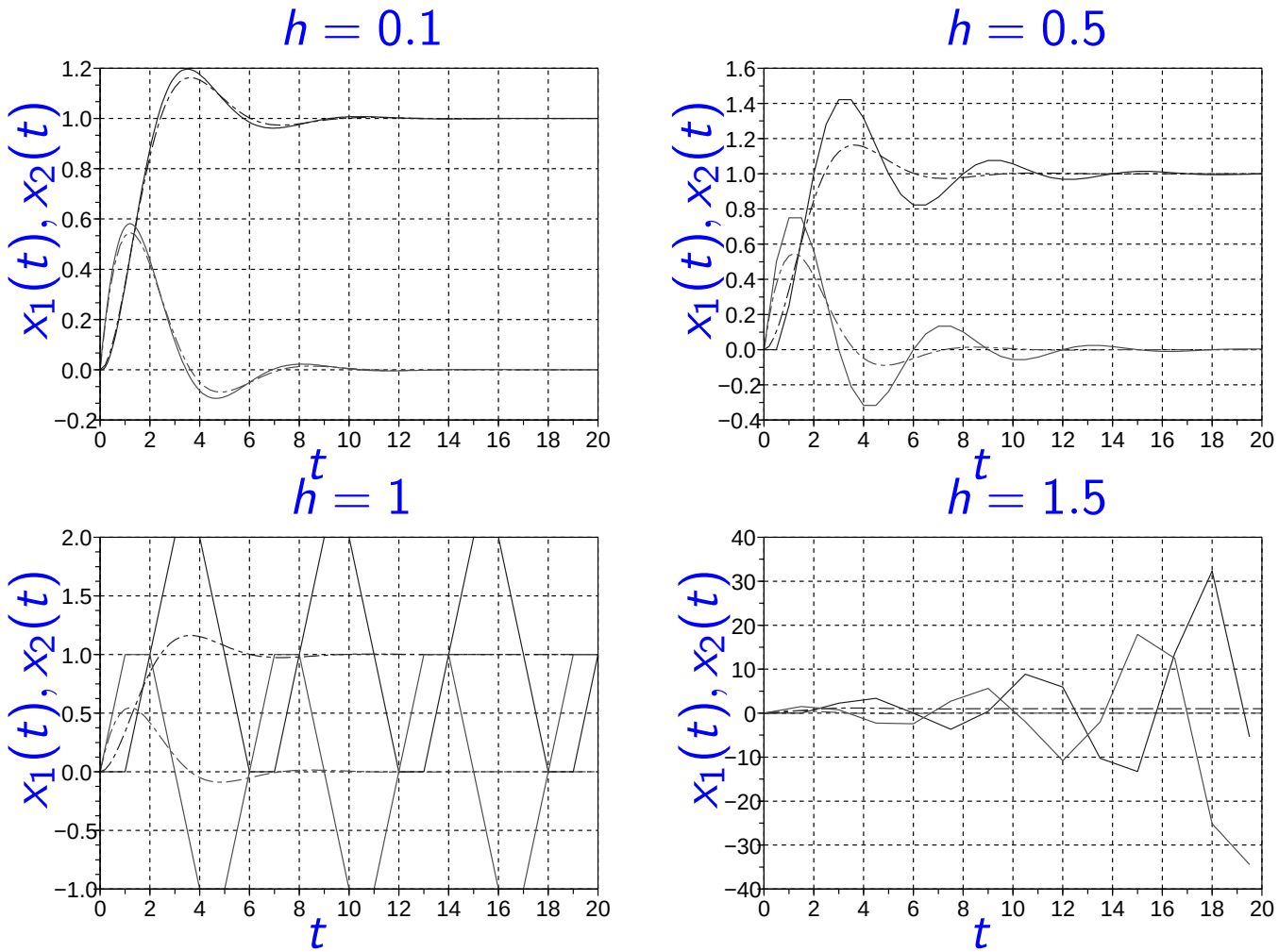

Figura 1.6: Solución de la Ecuación (1.1) con Forward Euler (sólida) para distintos pasos de integración y Solución Analítica (punteada)

Estabilidad Numérica

- Evidentemente no siempre se preserva la estabilidad.
- Para $h = 1$ perdemos la *estabilidad asintótica* y para $h = 1.5$ la solución numérica es claramente inestable.

¿Pasará lo mismo con todos los métodos numéricos?

La Figura 1.7 muestra los resultados con Backward Euler.

Estabilidad Numérica

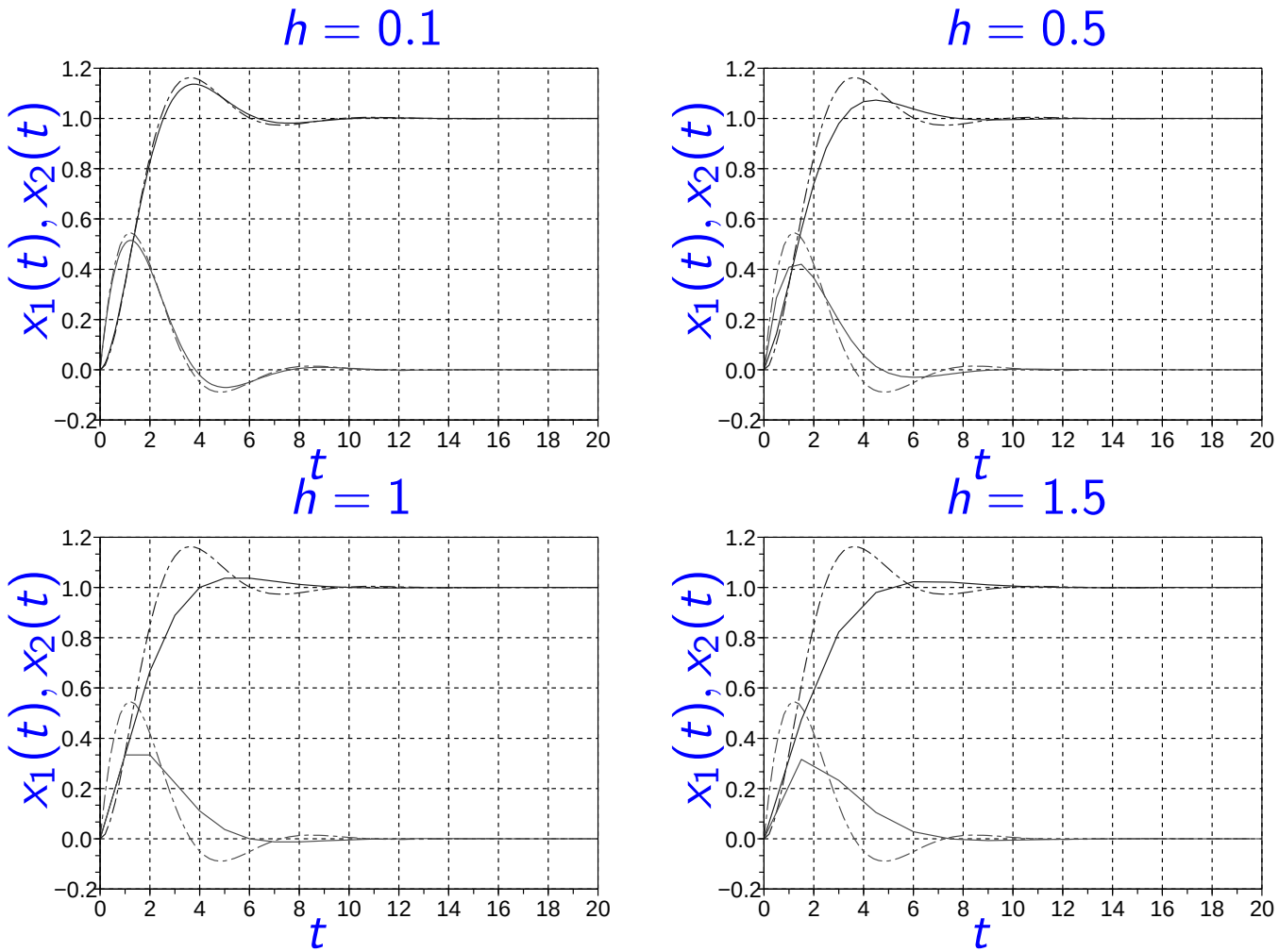

Figura 1.7: Solución de la Ecuación (1.1) con Backward Euler (sólida) para distintos pasos de integración y Solución Analítica (punteada)

Estabilidad Numérica - Análisis Teórico

Si aplicamos el método de Forward Euler a un sistema lineal y estacionario,

$$\dot{\mathbf{x}}(t) = A\,\mathbf{x}(t) + B\,\mathbf{u}(t) \tag{1.7}$$

resulta,

$$\mathbf{x}_{k+1} = \mathbf{x}_k + h\,(A\,\mathbf{x}_k + B\,\mathbf{u}_k) = (I + h\,A)\,\mathbf{x}_k + h\,B\,\mathbf{u}_k$$

Es decir, la ecuación en diferencias resultante tiene matriz de evolución

$$A_d = (I + h\,A)$$

Los autovalores λ_d de A_d serán las soluciones de

$$\det(\lambda_d\,I - A_d) = \det(\lambda_d\,I - I - h\,A) = 0$$

Estabilidad Numérica - Análisis Teórico

Luego,

$$\det(h^{-1}\ (\lambda_d - 1)\ I - A) = 0$$

de donde resulta que los autovalores de A_d y los de A se relacionan según:

$$h^{-1}\ (\lambda_d - 1) = \lambda$$

de donde,

$$\lambda_d = \lambda\ h + 1 \tag{1.8}$$

Es decir, los autovalores de la matriz del sistema discreto aproximado pueden calcularse directamente en función de los autovalores del sistema continuo original. Para que la aproximación resulte estable, deberá cumplirse para todos los autovalores de A que

$$|\lambda\ h + 1| < 1 \tag{1.9}$$

lo que sólo ocurrirá para valores pequeños de h.

Estabilidad Numérica – Análisis Teórico

Si aplicamos en cambio Backward Euler al sistema lineal y estacionario de la Ec.(1.7), resulta

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + h\,(A\,\mathbf{x}_{k+1} + B\,\mathbf{u}_{k+1}) \Rightarrow \\ (I - h\,A)\,\mathbf{x}_{k+1} &= \mathbf{x}_k + h\,B\,\mathbf{u}_{k+1} \\ \mathbf{x}_{k+1} &= (I - h\,A)^{-1}\,[\mathbf{x}_k + h\,B\,\mathbf{u}_{k+1}] \end{aligned}$$

de donde la ecuación en diferencias resultante tiene matriz de evolución

$$A_d = (I - h\,A)^{-1}$$

Los autovalores λ_d de A_d serán las soluciones de

$$\det(\lambda_d\,I - A_d) = \det(\lambda_d\,I - (I - h\,A)^{-1}) = 0$$

Estabilidad Numérica – Análisis Teórico

Luego,

$$\det(\lambda_d \ \frac{(I - h\ A)}{h\ \lambda_d} - \frac{I}{h\ \lambda_d}) = \det(I\ (\frac{1}{h} - \frac{1}{h\ \lambda_d}) - A) = 0$$

de donde resulta que los autovalores de A_d y los de A se relacionan según:

$$\frac{1}{h} - \frac{1}{h\ \lambda_d} = \lambda$$

o bien

$$\lambda_d = \frac{1}{1 - \lambda\ h}$$

Por lo tanto, la condición para que el resultado numérico sea estable es que

$$|\frac{1}{1 - \lambda\ h}| < 1 \qquad (1.10)$$

Notar que cuando $\mathbb{R}e(\lambda) < 0$, la condición se cumplirá siempre.

Estabilidad Numérica – Análisis Teórico

- Al simular un sistema estable con el método de Backward Euler, la solución numérica siempre resultará estable.
- Esta propiedad de preservar la estabilidad numérica es la principal ventaja de Backward Euler sobre Forward Euler.
- Una desventaja de Backward Euler es que para autovalores inestables $\mathbb{R}\mathrm{e}(\lambda) > 0$ puede resultar (si h es muy grande) $|\lambda_d| < 1$ (estable).

Por ejemplo, el sistema inestable

$$\dot{x}(t) = x(t)$$

tiene un autovalor $\lambda = 1$. Uusando cualquier paso $h > 2$, vemos que la condición de la Ec.(1.10) se cumple lo que indica que la solución numérica será estable.

Organización de la Presentación

Métodos Monopaso

- Los métodos de Forward y Backward Euler realizan sólo aproximaciones de primer orden. Luego, si se quiere obtener una buena precisión, se debe reducir excesivamente el paso de integración lo que implica una cantidad de cálculos en general inaceptable.
- Para obtener aproximaciones de orden mayor, será necesario utilizar más de una evaluación de la función $\mathbf{f}(\mathbf{x}, t)$ en cada paso.
- Cuando dichas evaluaciones se realizan de manera tal que para calcular $\mathbf{x}_{k+1}$ sólo se utiliza el valor de $\mathbf{x}_k$, se dice que el algoritmo es monopaso.
- Cuando se utilizan además valores anteriores de la solución ($\mathbf{x}_{k-1}$, $\mathbf{x}_{k-2}$, etc.), se dice que el algoritmo es multipaso.

Los métodos monopaso se suelen denominar también Métodos de Runge–Kutta, ya que el primero de estos métodos de orden alto fue formulado por Runge y Kutta a finales del siglo XIX.

Organización de la Presentación

Método de Heun

- Los métodos explícitos de Runge–Kutta realizan varias evaluaciones de la función $\mathbf{f}(\mathbf{x}, t)$ en cercanías del punto $(\mathbf{x}_k, t_k)$ y luego calculan $\mathbf{x}_{k+1}$ realizando una suma pesada de dichas evaluaciones.
- Uno de los algoritmos más simple es el denominado *Método de Heun*, inventado por Heun en el año 1900:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left(\mathbf{k}_1 + \mathbf{k}_2\right)$$

donde

$$\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_k, t_k)$$
$$\mathbf{k}_2 = \mathbf{f}(\mathbf{x}_k + h\,\mathbf{k}_1, t_k + h)$$

Comparado con Euler, este método realiza una evaluación adicional de la función $\mathbf{f}$ para calcular $\mathbf{k}_2$. El beneficio que se obtiene es una aproximación de segundo orden mucho más precisa que la de Euler.

Método de Heun

Para el sistema masa resorte de la Ec.(1.1), el método de Heun con paso $h = 0.1$ comenzaría con los siguientes cálculos:

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow$$

$$\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_0, t_0) = \begin{bmatrix} x_2 \\ 1 - x_1 - x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\mathbf{x}_0 + h\,\mathbf{k}_1 = \begin{bmatrix} 0 + 0.1 \cdot 0 \\ 0 + 0.1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0.1 \end{bmatrix} \Rightarrow$$

$$\mathbf{k}_2 = \mathbf{f}(\mathbf{x}_0 + h\,\mathbf{k}_1, t_0 + h) = \begin{bmatrix} 0.1 \\ 1 - 0 - 0.1 \end{bmatrix} = \begin{bmatrix} 0.1 \\ 0.9 \end{bmatrix}$$

$$\mathbf{x}_1 = \mathbf{x}_0 + \frac{h}{2}\,(\mathbf{k}_1 + \mathbf{k}_2) = \begin{bmatrix} 0 + 0.05 \cdot (0 + 0.1) \\ 0 + 0.05 \cdot (1 + 0.9) \end{bmatrix} = \begin{bmatrix} 0.005 \\ 0.095 \end{bmatrix}$$

Método de Runge–Kutta 4

Otro método, uno de los más utilizados, es el de Runge–Kutta de orden 4 (RK4), desarrollado por Runge y Kutta en 1895:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{6}\left(\mathbf{k}_1 + 2\,\mathbf{k}_2 + 2\,\mathbf{k}_3 + \mathbf{k}_4\right)$$

donde

$$\begin{aligned}
\mathbf{k}_1 &= \mathbf{f}(\mathbf{x}_k, t_k) \\
\mathbf{k}_2 &= \mathbf{f}(\mathbf{x}_k + h\,\frac{\mathbf{k}_1}{2}, t_k + \frac{h}{2}) \\
\mathbf{k}_3 &= \mathbf{f}(\mathbf{x}_k + h\,\frac{\mathbf{k}_2}{2}, t_k + \frac{h}{2}) \\
\mathbf{k}_4 &= \mathbf{f}(\mathbf{x}_k + h\,\mathbf{k}_3, t_k + h)
\end{aligned}$$

Este algoritmo, utiliza cuatro evaluaciones de la función $\mathbf{f}$ en cada paso. El beneficio es que resulta una aproximación de orden 4.

Métodos Explícitos de Runge–Kutta

La Tabla 1 resume los resultados de simular el sistema Masa Resorte de la Ec.(1.1) con los métodos de Euler, Heun y Runge–Kutta de orden 4. En cada caso, se reporta el máximo error obtenido en toda la simulación (esto es, el máximo error global).

Paso	Error Euler	Error Heun	Error RK4
$h = 0.5$	0.298	0.0406	4.8×10^{-4}
$h = 0.1$	0.042	1.47×10^{-3}	6.72×10^{-7}
$h = 0.05$	0.0203	3.6×10^{-4}	4.14×10^{-8}
$h = 0.01$	3.94×10^{-3}	1.42×10^{-5}	6.54×10^{-11}

Cuadro 1: Máximo Error Cometido por los Métodos de Euler, Heun y RK4 para Distintos Pasos de Integración con el Sistema Masa Resorte de le Ec.(1.1)

Métodos Explícitos de Runge–Kutta

- Hay infinidad de métodos de Runge Kutta explícitos para diversos órdenes (incluyendo métodos de órdenes mayores a 10).
- Los de orden 1 a 5 son los más utilizados en la práctica.
- En lo referente a estabilidad, los métodos de RK tienen características similares a las de Forward Euler. Esto es, preservan la estabilidad para valores no muy grandes del paso de integración h.
- Por lo tanto, aunque la precisión sea buena, no podremos utilizar pasos muy grandes debido a los límites que impone la estabilidad.

Métodos Explícitos de Runge–Kutta – Estabilidad

Por ejemplo, para el método de Heun, podemos analizar la estabilidad sobre un sistema escalar de primer orden:

$$\dot{x} = \lambda\ x(t)$$

cuya solución analítica será estable si $\lambda < 0$. Aplicando Heun, tenemos,

$$\begin{aligned} x_{k+1} &= x_k + \frac{h}{2}(k_1 + k_2) \\ &= x_k + \frac{h}{2}(\lambda x_k + \lambda\ (x_k + h\ \lambda x_k)) \\ &= x_k + h\ \lambda x_k + \frac{h^2\lambda^2}{2}\ x_k = (1 + h\ \lambda + \frac{h^2\lambda^2}{2})\ x_k \\ &= \lambda_d\ x_k \end{aligned}$$

es decir, el autovalor discreto resulta $\lambda_d = 1 + h\ \lambda + \frac{h^2\lambda^2}{2}$.

Métodos Explícitos de Runge–Kutta – Estabilidad

La condición de estabilidad numérica es entonces

$$|1 + h\,\lambda + \frac{h^2\lambda^2}{2}| < 1$$

que se traduce en dos desigualdades:

$$\begin{cases} 1 + h\,\lambda + \frac{h^2\lambda^2}{2} & < 1 \\ 1 + h\,\lambda + \frac{h^2\lambda^2}{2} & > -1 \end{cases}$$

La primera desigualdad conduce a la condición

$$h < \frac{2}{|\lambda|} \tag{1.11}$$

y la segunda se cumple siempre que $\lambda < 0$.

Para un autovalor real negativo, el método de Heun dará un resultado estable cuando se cumpla la Ec.(1.11). Este resultado es idéntico al de Forward Euler para autovalores reales negativos.

Organización de la Presentación

Métodos Monopaso Implícitos

- Backward Euler es un algoritmo implícito de primer orden cuya principal ventaja es preservar la estabilidad de la solución numérica para cualquier paso de integración.
- Hay diversos métodos implícitos monopaso de orden mayor que, al igual que Backward Euler, preservan la estabilidad. Uno de los más usados es la Regla Trapezoidal

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left[\mathbf{f}(\mathbf{x}_k, t_k) + \mathbf{f}(\mathbf{x}_{k+1}, t_{k+1})\right] \qquad (1.12)$$

Este método implícito realiza una aproximación de segundo orden y tiene la propiedad (al menos en sistemas lineales y estacionarios) de que la solución numérica es estable si y sólo si la solución analítica es estable.

Organización de la Presentación

Algoritmos de Control de Paso

- En muchos casos es posible implementar algoritmos que cambien el paso de integración h de forma automática a medida que avanza la simulación.
- Los algoritmos de control de paso tienen por propósito mantener el error de simulación acotado ajustando el paso en función del error estimado.

La idea es muy simple, en cada paso se hace lo siguiente:

1. Se da un paso con el método de integración elegido calculando $\mathbf{x}_{k+1}$ y cierto paso h.
2. Se estima el error cometido.
3. Si el error es mayor que la tolerancia, se disminuye el paso de integración h y se recalcula $\mathbf{x}_{k+1}$ volviendo al punto 1.
4. Si el error es menor que la tolerancia, se acepta el valor de $\mathbf{x}_{k+1}$ calculado, se incrementa el paso h y se vuelve al punto 1 para calcular $\mathbf{x}_{k+2}$.

Algoritmos de Control de Paso

Si bien la idea es muy sencilla, de la descripción anterior surgen varias cuestiones:

- ¿Cómo estimamos el error?
- ¿Qué ley utilizamos para recalcular el paso de integración h en cada caso?
- ¿Qué valor de paso de integración usamos inicialmente?

Algoritmos de Control de Paso – Estima del error

La estima del error de integración se hace comparando la solución con dos métodos de orden diferente. Para ahorrar cálculos, se suelen buscar métodos que compartan evaluaciones comunes en las funciones.

Por ejemplo, las siguientes fórmulas calculan el método de Heun (2do orden) y un método de RK de tercer orden:

$$\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_k, t_k)$$
$$\mathbf{k}_2 = \mathbf{f}(\mathbf{x}_k + h\,\mathbf{k}_1, t_k + h)$$
$$\mathbf{k}_3 = \mathbf{f}(\mathbf{x}_k + \frac{h}{4}\,\mathbf{k}_1 + \frac{h}{4}\,\mathbf{k}_2, t_k + \frac{h}{2})$$

y luego

$$\mathbf{x}_{k+1} = x_k + \frac{h}{2}(\mathbf{k}_1 + \mathbf{k}_2) \tag{1.13}$$

$$\tilde{\mathbf{x}}_{k+1} = x_k + \frac{h}{6}(\mathbf{k}_1 + \mathbf{k}_2 + 4\mathbf{k}_3) \tag{1.14}$$

Algoritmos de Control de Paso – Estima del error

de donde puede estimarse el error como

$$\mathbf{e}_{k+1} \approx \mathbf{x}_{k+1} - \tilde{\mathbf{x}}_{k+1} = \frac{\mathbf{k}_1 + \mathbf{k}_2 - 2\mathbf{k}_3}{3}$$

- Este método se denominará RK23, ya que se trata de un algoritmo de Runge–Kutta de orden 2 con estima de error de orden 3.
- Como puede observarse la Ec.(1.13) (Heun) y la Ec.(1.14) (RK3) comparten las etapas de cálculo de $\mathbf{k}_1$ y $\mathbf{k}_2$, lo que ahorra bastante costo computacional.
- Uno de los métodos más utilizados en la práctica es un RK45 denominado *Runge–Kutta–Fehlberg*, que utiliza 6 evaluaciones de la función $\mathbf{f}$ para calcular dos soluciones, una de orden 4 y otra de orden 5.

Algoritmos de Control de Paso – Ajuste del Paso

El error local que comete un método de orden N es

$$err \approx c\ h^{N+1} \tag{1.15}$$

donde c es una constante desconocida. Si usamos cierto paso h y obtenemos un error err, la pregunta es, ¿que paso h_0 deberíamos usar para obtener un error tol, donde tol es la tolerancia de error que admitimos?

Podemos escribir

$$tol \approx c\ h_0^{N+1}$$

Dividiendo miembro a miembro con la Ec.(1.15) resulta,

$$\frac{tol}{err} \approx \frac{h_0^{N+1}}{h^{N+1}}$$

de donde,

$$h_0 = h\ (tol/err)^{\frac{1}{N+1}}$$

que nos da el nuevo paso a utilizar.

Algoritmos de Control de Paso – Ajuste del Paso

Esta última ecuación generalmente se modifica de manera tal que

$$h_0 = 0.8\ h\ (tol/err)^{\frac{1}{N+1}} \tag{1.16}$$

para asegurar que el paso utilizado no resulte demasiado cercano al límite teórico de precisión y evitar tener que recalcular pasos.

Algoritmos de Control de Paso – Ejemplo

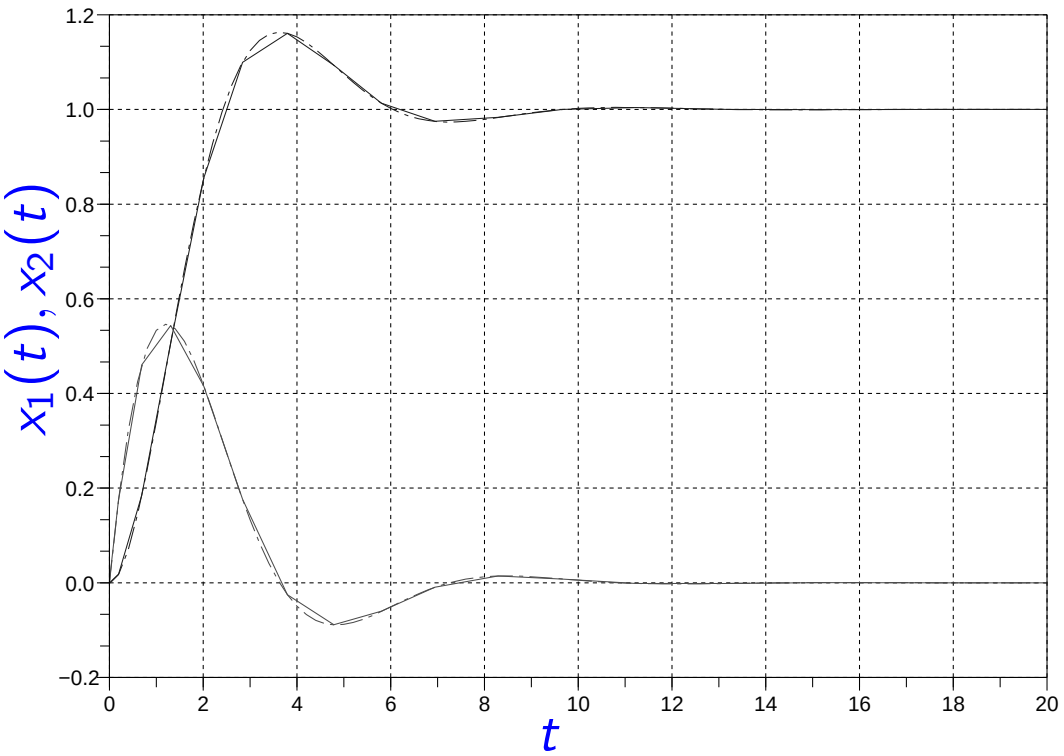

Figura 1.8: Solución del Sistema Masa Resorte con Runge–Kutta–Fehlberg (sólida) con $tol = 0.001$ y Solución Analítica (punteada)

Algoritmos de Control de Paso – Ejemplo

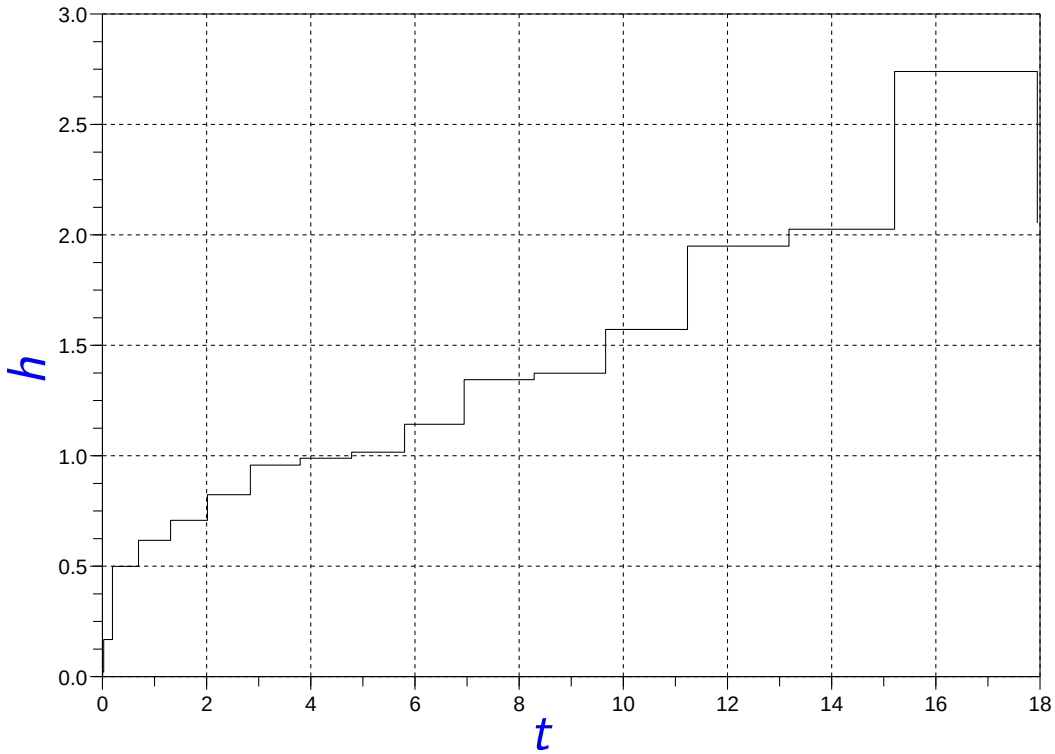

Figura 1.9: Variación del Paso de Integración en la Solución de la Fig.1.8

Algoritmos de Control de Paso – Parámetros

Parámetros que utilizan los algoritmos de control de paso:

- Tolerancia relativa: Es el error que se admite, expresado como fracción de los valores de las variables.
- Tolerancia absoluta: Es el mínimo valor de tolerancia a utilizar, en caso que las variables tomen valores cercanos a 0.
- Máximo paso de integración: Es un límite para evitar pasos muy grandes.
- *Mínimo paso de integración:* Se utiliza para evitar la utilización de pasos exageradamente pequeños.
- Intervalo de Interpolación: Al utilizar control de paso, la solución se calcula a intervalos irregulares de tiempo. Si se quiere contar con la solución calculada en otros instantes de tiempo se pueden utilizar algoritmos de interpolación que calculan la solución en los instantes deseados.

Organización de la Presentación

Métodos Multipaso

- Los métodos monopaso obtienen aproximaciones de orden alto utilizando para esto varias evaluaciones de la función **f** en cada paso.
- Para evitar este costo computacional adicional, se han formulado diversos algoritmos que, en lugar de evaluar repetidamente la función **f** en cada paso, utilizan los valores evaluados en pasos anteriores.

Estos algoritmos se denominan Métodos Multipaso

Organización de la Presentación

Métodos Multipaso Explícitos – Adams–Bashforth

Dentro de los métodos multipaso explícitos encontramos, entre los más utilizados, a los métodos de Adams–Bashforth (AB).

- Llamando $\mathbf{f}_k \triangleq \mathbf{f}(\mathbf{x}_k, t_k)$, el método de AB de orden 2 es:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{2}\left(3\mathbf{f}_k - \mathbf{f}_{k-1}\right) \tag{1.17}$$

- El método de AB de orden 4 es:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \frac{h}{24}\left(55\mathbf{f}_k - 59\mathbf{f}_{k-1} + 37\mathbf{f}_{k-2} - 9\mathbf{f}_{k-3}\right) \tag{1.18}$$

Los métodos de AB requieren una única evaluación de la función $\mathbf{f}$ en cada paso.

Métodos Multipaso Explícitos – Arranque

Un problema de los métodos multipaso es el arranque.

- En AB2, la expresión de la Ec.(1.17) sólo puede utilizarse para calcular la solución a partir de $k = 2$. Para el primer pasos no está definido el valor de $\mathbf{x}_{-1}$ requerido por la fórmula.
- Similarmente, el algoritmo de AB4 de la Ec.(1.18) sólo puede usarse a partir de $k = 4$.

Los primeros pasos de los métodos multipaso deben realizarse con algún método monopaso de, al menos, el mismo orden de precisión. Por ejemplo, para el método de AB4 de la Ec.(1.18), deberían darse los tres primeros pasos con Runge–Kutta de orden 4.

Métodos Multipaso Explícitos – Ejemplo

Veamos por ejemplo, el resultado de utilizar AB2 para simular el sistema masa resorte del ejemplo. Usaremos como valores iniciales $\mathbf{x}_0 = [0\ 0]$ y $\mathbf{x}_1 = [0.005\ 0095]$, este último calculado previamente con el método de Heun. Luego,

$$\mathbf{x}_0 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \Rightarrow \mathbf{f}(\mathbf{x}_0, t_0) = \begin{bmatrix} x_2 \\ 1 - x_1 - x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

$$\mathbf{x}_1 = \begin{bmatrix} 0.005 \\ 0.095 \end{bmatrix} \Rightarrow \mathbf{f}(\mathbf{x}_1, t_1) = \begin{bmatrix} x_2 \\ 1 - x_1 - x_2 \end{bmatrix} = \begin{bmatrix} 0.095 \\ 0.9 \end{bmatrix}$$

$$\mathbf{x}_2 = \mathbf{x}_1 + \frac{h}{2}(3\ \mathbf{f}_1 - \mathbf{f}_0) = \begin{bmatrix} 0.005 \\ 0.095 \end{bmatrix} + 0.05 \left(3 \begin{bmatrix} 0.095 \\ 0.9 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix} \right) = \begin{bmatrix} 0.01925 \\ 0.18 \end{bmatrix}$$

Métodos Multipaso Explícitos – ABM

- Otro problema de los métodos de AB es que resultan estables sólo para valores muy pequeños del paso de integración.
- Esto se mejora utilizando los métodos de Adams–Bashforth–Moulton (ABM), donde la versión de tercer orden, por ejemplo, está dada por la siguiente fórmula

$$\mathbf{x}_{k+1}^P = \mathbf{x_k} + \frac{h}{12}(23\mathbf{f}_k - 16\mathbf{f}_{k-1} + 5\mathbf{f}_{k-2})$$

$$\mathbf{x}_{k+1} = \mathbf{x_k} + \frac{h}{12}(5\mathbf{f}_{k+1}^P + 8\mathbf{f}_k - \mathbf{f}_{k-1})$$

con $\mathbf{f}_{k+1}^P = \mathbf{f}(x_{k+1}^P, t_{k+1})$.

- Este método, denominado *predictor–corrector*, tiene mejor estabilidad que AB3 a costa de utilizar una evaluación extra de la función $\mathbf{f}$.

Organización de la Presentación

Métodos Multipaso Implícitos

- Para obtener soluciones numéricas que preserven la estabilidad independientemente del paso de integración hay que recurrir a los métodos implícitos.
- Los métodos implícitos más utilizados en la práctica son los denominados Backward Difference Formulae (BDF), que son de tipo multipaso. Por ejemplo, el siguiente es el método de BDF de orden 3:

$$\mathbf{x}_{k+1} = \frac{18}{11}\mathbf{x}_k - \frac{9}{11}\mathbf{x}_{k-1} + \frac{2}{11}\mathbf{x}_{k-2} + \frac{6}{11} \cdot h \cdot \mathbf{f}_{k+1} \qquad (1.19)$$

- Este método tiene prácticamente el mismo costo computacional que Backward Euler, ya que la ecuación a resolver es muy similar. Sin embargo, BDF3 es de tercer orden.
- Existen métodos de BDF de hasta orden 8, si bien los más utilizados son los de orden 1 (Backward Euler) hasta 5.

Organización de la Presentación

Control de Paso

- En los métodos multipaso se puede también controlar el paso de integración de manera similar a la de los métodos monopaso.
- Sin embargo, las fórmulas de los métodos multipaso son sólo válidas asumiendo paso constante.
- Por lo tanto, para cambiar el paso en un método multipaso hay que interpolar los últimos valores de la solución a intervalos regulares correspondientes al nuevo paso de integración.

Dado que cambiar el paso tiene un costo adicional, los algoritmos de control de paso en métodos multipaso sólo modifican el paso cuando el cambio es suficientemente grande.

Organización de la Presentación

Algunas Dificultades

Muchos sistemas dinámicos en la práctica tienen modelos en sistemas de ecuaciones diferenciales que resultan problemáticos para los distintos métodos de integración numérica. Entre estos encontramos:

- Sistemas Stiff o Rígidos,
- Sistemas Marginalmente Estables,
- Sistemas con Discontinuidades,

Organización de la Presentación

Sistemas Stiff

Consideremos nuevamente el sistema masa resorte de la Ec.(1.1), pero ahora supongamos que el coeficiente de rozamiento es $b = 100$. Con este cambio, el sistema tiene matriz de evolución

$$A = \begin{bmatrix} 0 & 1 \\ -1 & -100 \end{bmatrix}$$

cuyos autovalores son $\lambda_1 \approx -0.01$ y $\lambda_2 \approx -100$.

El sistema tiene ahora dinámica rápida y dinámica lenta de manera simultánea. Este es un caso típico de Sistema Stiff.

Sistemas Stiff

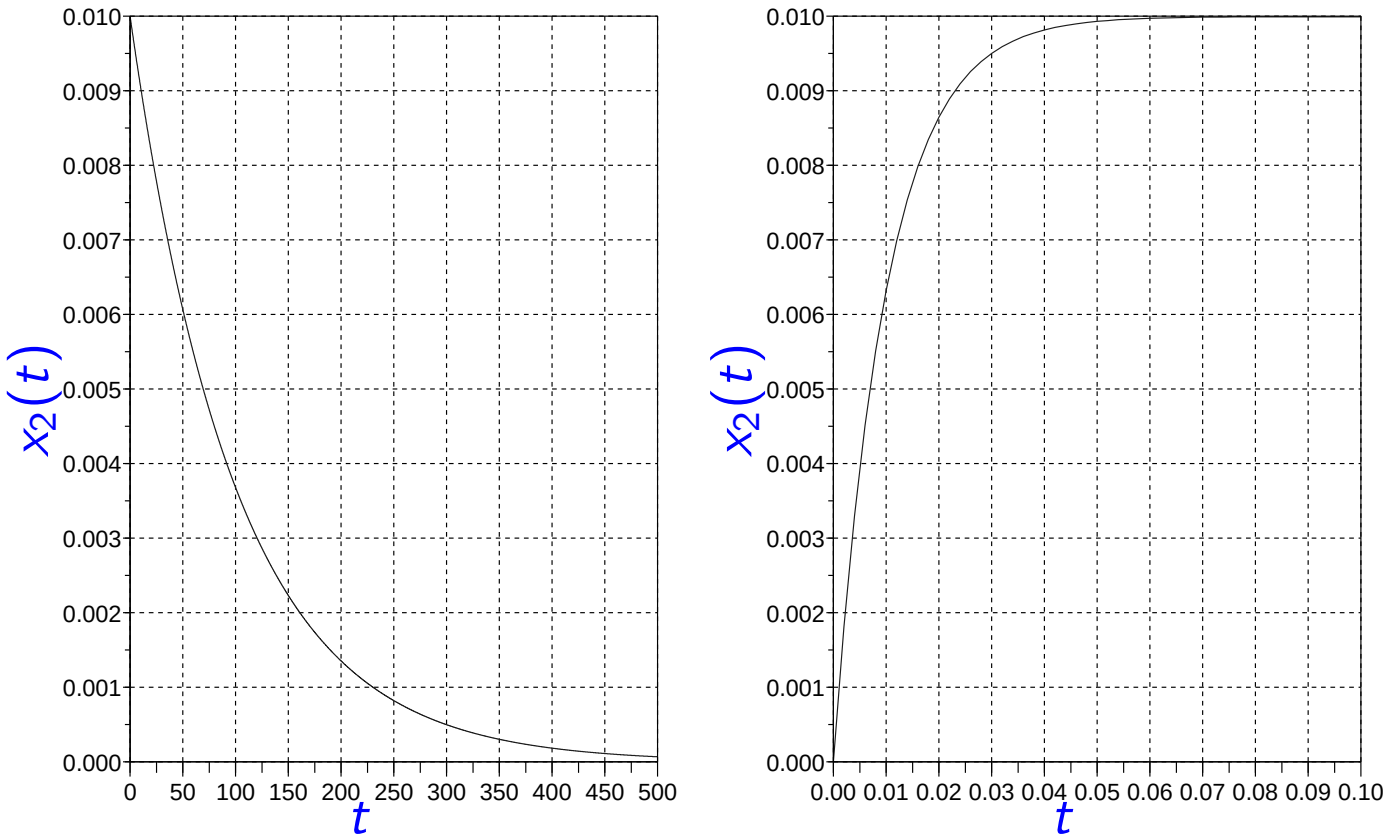

Figura 1.10: Solución Analítica del Sistema Masa Resorte con $b = 100$.

Sistemas Stiff

- Es claro que si queremos simular este sistema usando algún método de integración, debemos hacerlo hasta un tiempo final de al menos $t_f = 500$.
- Además, si utilizamos el método de Euler, por ejemplo, deberemos usar un paso de integración $h < 0.02$ ya que de otra manera tendremos un resultado inestable según la Ec.(1.9).
- Esto implicará que haremos al menos $500/0.2 = 25000$ pasos para completar la simulación.

La primer idea es entonces usar algún método de paso variable, que debería usar un paso pequeño al comienzo y luego agrandarlo.

Sistemas Stiff – Simulación con RK45

Utilizando el algoritmo de RK45 ocurre lo siguiente:

- El método necesita 13604 pasos para completar la simulación.
- La Figura 1.11 muestra la evolución del paso de integración:
- Aunque en la solución analítica de la Fig.1.10 la dinámica rápida desaparece a partir de $t = 0.1$, el método de RK45 nunca puede aumentar el paso de integración mucho más allá de $h \approx 0.04$.
- La explicación es sencilla: debido al autovalor rápido $\lambda_2 \approx -100$ con pasos de integración mayores el método de RK4 se torna inestable, lo que obliga a disminuir nuevamente el paso.

Sistemas Stiff – Simulación con RK45

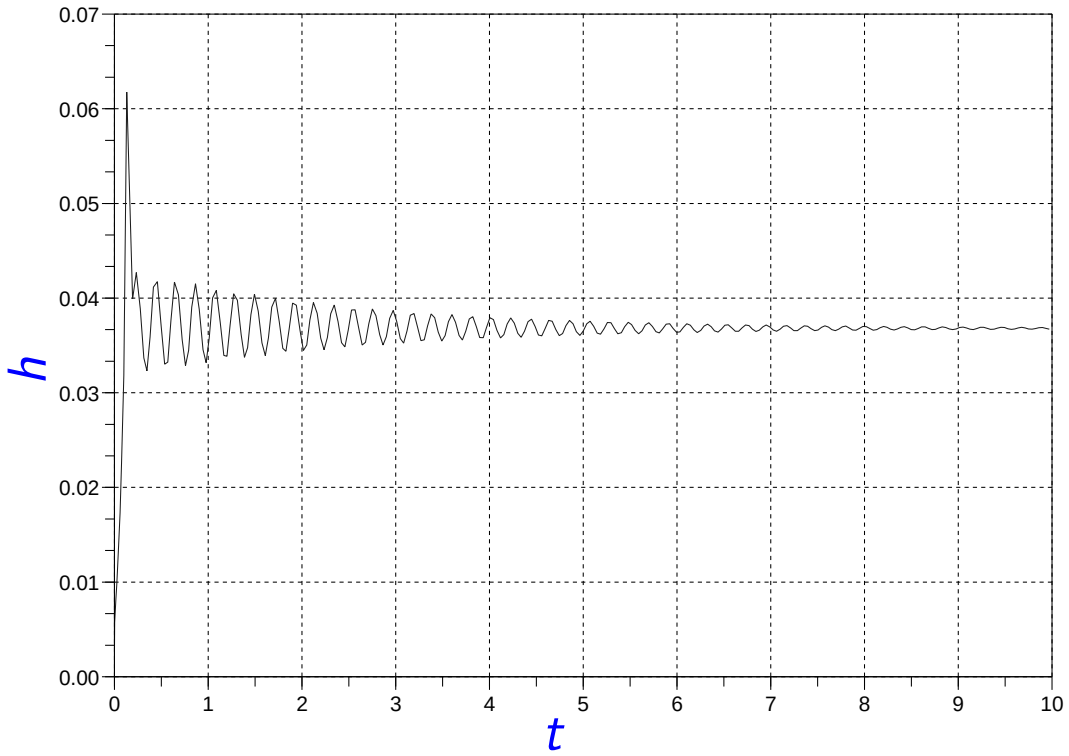

Figura 1.11: Paso de Integración h en la Simulación de un Sistema Stiff con RK45.

Sistemas Stiff – Métodos implícitos

El problema de RK45 ocurre con cualquier método explícito. Esto sólo puede evitarse usando algoritmos que no se inestabilicen (necesariamente implícitos).

- Utilizando por ejemplo, un método implícito de RK de cuarto orden con control de error de orden 5, con la misma tolerancia que en RK45 (0.001), la simulación se completa en sólo 33 pasos y se obtiene una precisión idéntica a la anterior.
- La Fig.1.12 muestra la evolución del paso de integración en este caso. El mismo está sólo limitado por la precisión y nunca por la estabilidad.

Sistemas Stiff – Métodos implícitos

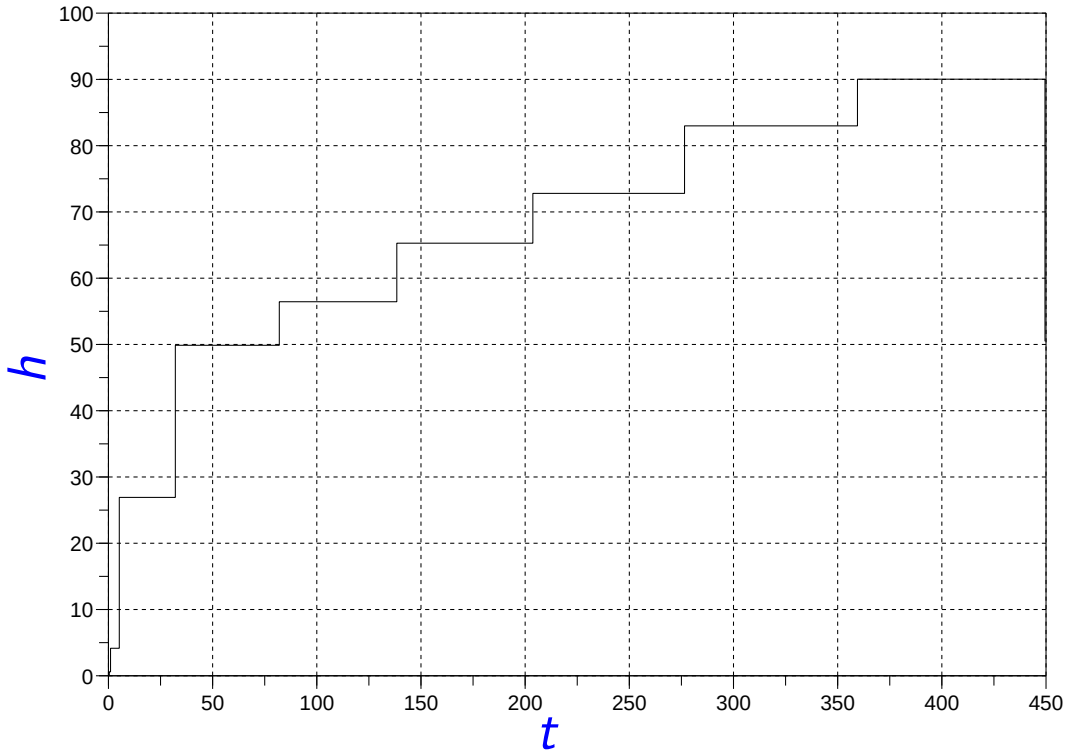

Figura 1.12: Paso de Integración h en la Simulación de un Sistema Stiff con un método implícito.

Organización de la Presentación

Sistemas Marginalmente Estables

- Muchos sistemas tienen nulo o escaso amortiguamiento.
- En estos casos, las soluciones muestran oscilaciones sostenidas o bien que se amortiguan muy lentamente.
- Esto puede observarse en el sistema Masa Resorte con $b = 0$ o tomando b muy peqeño.

Los sistemas con estas propiedades se denominan marginalmente estables y también revisten algunos problemas para la simulación numérica.

Sistemas Marginalmente Estables

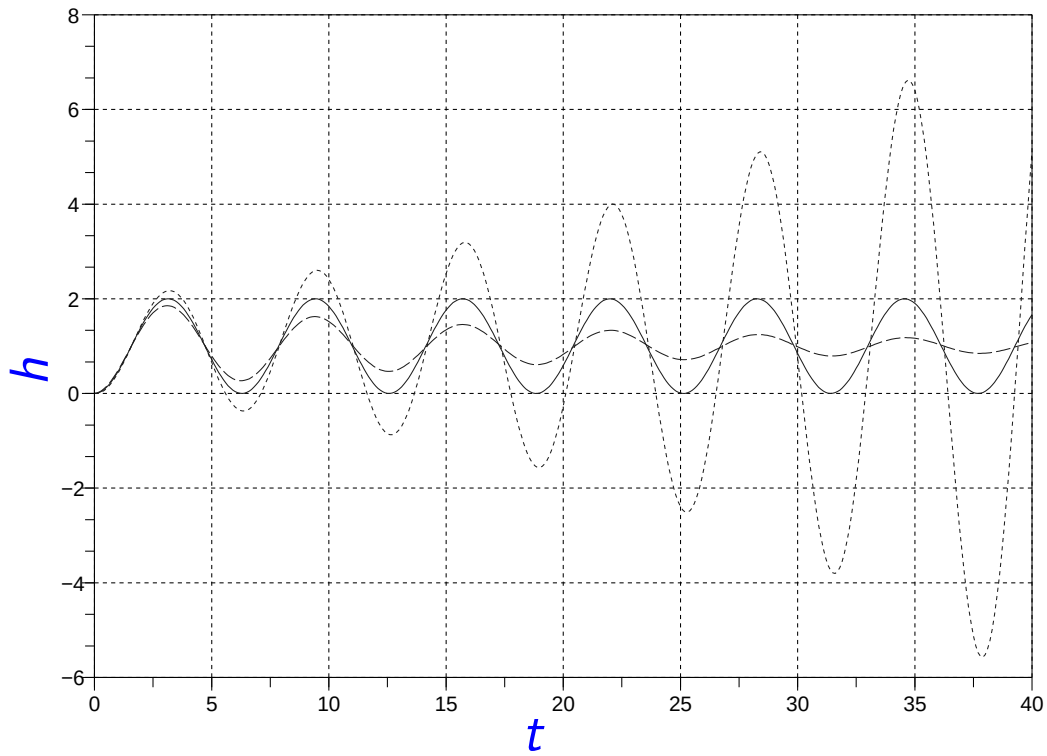

Figura 1.13: Sistema Masa Resorte con $b = 0$: Solución analítica (sólida), Forward Euler (punteada) y Backward Euler (línea discontinua).

Sistemas Marginalmente Estables

- Ninguna de las dos soluciones respeta la característica marginalmente estable original por lo que los resultados son cualitativamente incorrectos.
- Para obtener buenos resultados hay que utilizar métodos denominados F–Estables que preservan también la estabilidad marginal.
- Uno de estos métodos es la Regla Trapezoidal que presentamos en la Ec.(1.12).

Organización de la Presentación

Sistemas con Discontinuidades

La siguiente ecuación es un modelo muy simple de una pelotita que rebota contra el piso:

$$\dot{x}(t) = v(t)$$
$$\dot{v}(t) = -g - s_w(t) \cdot \frac{1}{m}(k \cdot x(t) + b \cdot v(t))$$

donde

$$s_w = \begin{cases} 0 & \text{si } x(t) > 0 \\ 1 & \text{en otro caso} \end{cases}$$

- Cuando $x(t) > 0$ la pelotita está en el aire y responde a una ecuación de caída libre ($s_w = 0$).
- Cuando la pelotita entra en contacto con el piso, en cambio, sigue un modelo masa–resorte–amortiguador, lo que produce el rebote.

Sistemas con Discontinuidades

- En este caso, tomamos parámetros $m = 1$, $b = 30$, $k = 1 \times 10^6$ y $g = 9.81$.
- Simulamos el sistema utilizando el método de RK4, durante 5 segundos, a partir de la condición inicial $x(0) = 1$, $v(0) = 0$.
- Comenzamos a tener resultados decentes con un paso de integración $h = 0.002$.
- En la Fig 1.14 se muestran los resultados de la simulación con pasos $h = 0.002$, $h = 0.001$ y $h = 0.0005$.

Sistemas con Discontinuidades

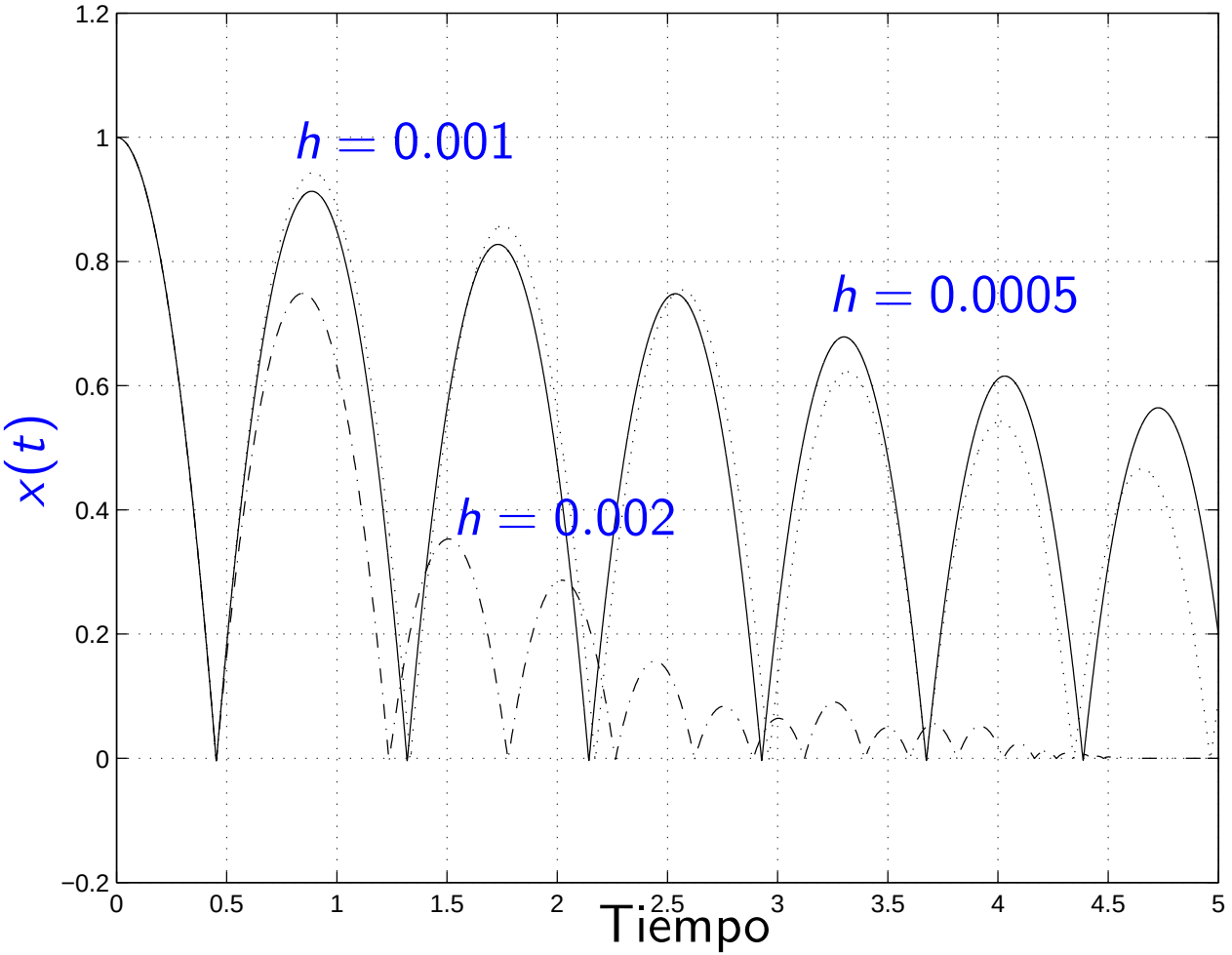

Figura 1.14: Simulación con RK4 de la pelotita rebotando.

Sistemas con Discontinuidades

- Evidentemente, no podemos confiar en los resultados de esta simulación.
- Mirando el comienzo de la simulación, no hay error apreciable hasta el primer pique. Evidentemente, el problema tiene que ver con la discontinuidad.
- Veamos que ocurre en tanto si utilizamos un método de paso variable.
- En la Fig 1.15 se pueden apreciar los resultados con RK23 utilizando las tolerancias relativas 10^{-3}, 10^{-4} y 10^{-5}.

Sistemas con Discontinuidades

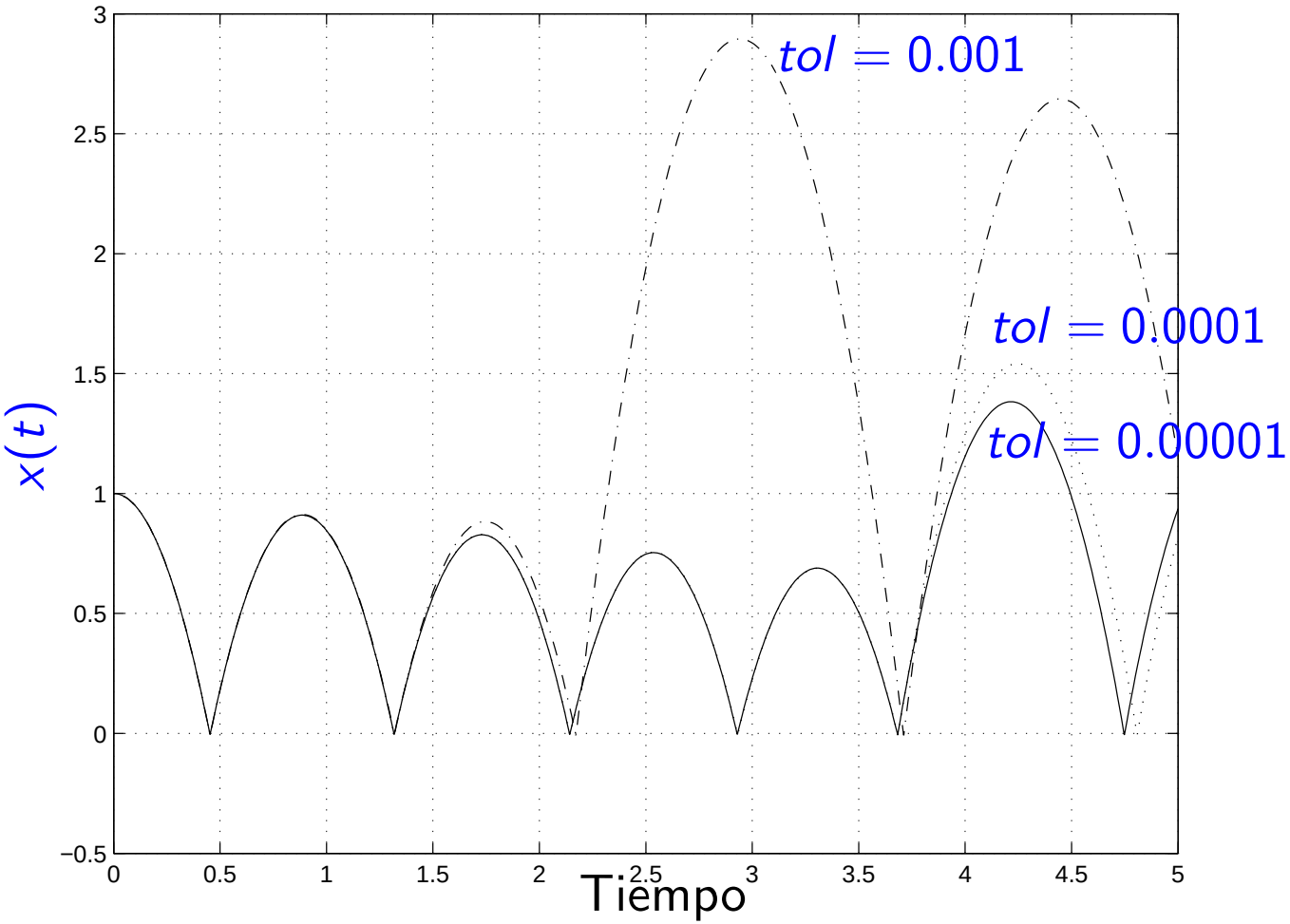

Figura 1.15: Simulación con RK23 de la pelotita rebotando.

Sistemas con Discontinuidades

- En ambos casos el problema es que se dieron pasos que atravesaron la discontinuidad. Es decir, en t_k teníamos $x(t_k) > 0$ y en t_{k+1} resultó $x(t_{k+1}) < 0$. Como la función $\mathbf{f}$ resulta discontinua entre t_k y t_{k+1}, se produce un error muy grande.
- La manera de corregir esto es evitando la situación anterior.
- Si nos encontramos con que $x(t_k) > 0$ y $x(t_{k+1}) < 0$ hay que volver atrás y buscar el punto donde se produce la discontinuidad, esto es, el valor de $\tilde{t}$ para el cual $x(\tilde{t}) = 0$.
- Una vez detectado este punto, se debe realizar un paso de valor $h = \tilde{t} - t_k$ y desde este nuevo punto reiniciar la simulación.

Este procedimiento es la base para la Detección y Tratamiento de discontinuidades, también llamadas Cruces por Cero.

Lecturas de Profundización

- Los contenidos de este apunte están principalmente basados en las notas de clases de la materia de doctorado Simulación de Sistemas Continuos [Kof06], que a su vez están basadas principalmente en el libro [CK06].
- Por otro lado, los libros [HNW00] y [HW91] son referencia obligada en la disciplina para quienes tengan interés en las propiedades más matemáticas de los algoritmos.
- Finalmente, una colección muy importante de algoritmos y programas para implementar diversos métodos numéricos (no sólo para ecuaciones diferenciales, sino para problemas más generales) puede encontrarse en el libro clásico [PTVF07].

Referencias


François Cellier and Ernesto Kofman.
Continuous System Simulation.
Springer, New York, 2006.


Ernst Hairer, Syvert Nørsett, and Gerhard Wanner.
Solving Ordinary Differential Equations I: Nonstiff Problems.
Springer, Berlin, Germany, 2^{nd} edition, 2000.


Ernst Hairer and Gerhard Wanner.
Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems.
Springer, Berlin, 1991.


Ernesto Kofman.
Simulación de Sistemas Continuos. Notas de Clase.
Departamento de Control, FCEIA, UNR. Disponible online en
`http://www.fceia.unr.edu.ar/control/ssc/notas_ssc.pdf`, 2006.


William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery.
Numerical Recipes: The Art of Scientific Computing.
Cambridge University Press, New York, 3^{rd} edition, 2007.