

Ecuaciones diferenciales

Métodos numéricos de resolución de EDOs

Índice

Esquema.	2
Ideas clave	3
7.1 Introducción y objetivos	3
7.2 Problemas de valores iniciales	5
7.3 Problemas de contorno	11
7.4 Referencias bibliográficas	21

Problemas de valores iniciales

- Método de Euler
- Método de Runge-Kutta
- Otros métodos (Verlet y Numerov)

Control adaptativo del paso de integración

Problemas de contorno

- Método de disparo
- Métodos de relajación

Cálculo de los ceros de una función

- Método de la bisección
- Método de Newton

7.1 Introducción y objetivos

A estas alturas del curso habrá quedado claro que de la infinidad de tipos de ecuaciones diferenciales que existen, tan solo una pequeña fracción de ellas se puede resolver de manera exacta. Para la inmensa mayoría (por ejemplo, las ecuaciones no lineales), no existen métodos de resolución analíticos. Afortunadamente, en estos casos siempre se puede recurrir a métodos numéricos.

Si el cálculo analítico se basa en las operaciones con intervalos infinitesimales $\Delta x \rightarrow 0$, el cálculo numérico consiste en quedarse en un valor finito de Δx , al que llamamos *paso de integración* h . El cálculo numérico sustituye una operación analítica (difícil o imposible) por muchas operaciones algebraicas sencillas, lo cual permite atacar prácticamente cualquier problema. El precio a pagar es un error en el resultado del cálculo que depende del tamaño del paso de integración. Típicamente, el *orden del error* cometido es proporcional a una potencia de h , lo cual expresamos como $O(h^n)$. Un método de integración se llama de orden n si su término de error es $O(h^{n+1})$.

Ejemplo 1. Error numérico

Supongamos que queremos calcular el valor de una determinada integral con un método numérico de segundo orden. El término de error es $O(h^3)$, por tanto, si el paso de integración es $h = 0.1$, el error será del orden de 0.001, esto es del 0.1 %; y si $h = 0.01$, el error será del orden del 0.0001 %.

Un concepto similar lo encontramos al truncar el desarrollo de Taylor de una función en torno a un punto x_0 . Si nos quedamos en el término de segundo orden,

el error cometido es del orden de $(x - x_0)^3$:

$$f(x) = f(x_0) + \left. \frac{df}{dx} \right|_{x_0} (x - x_0) + \frac{1}{2} \left. \frac{d^2f}{dx^2} \right|_{x_0} (x - x_0)^2 + O((x - x_0)^3) .$$

Evidentemente, cuanto más pequeño sea el valor de h más se aproximará la solución numérica a la solución real, sin embargo, un paso de integración más pequeño significa mayor número de operaciones y por tanto mayor tiempo de computación. Los algoritmos numéricos que estudiaremos en este tema tratan de encontrar un equilibrio entre el tiempo de computación y el error cometido, de manera que ambos sean aceptables. Como veremos, el tamaño de h no es el único factor que determina la precisión del cálculo ya que la forma del algoritmo es esencial, de modo que unos métodos son más eficientes que otros.

Los algoritmos numéricos para resolver de manera aproximada problemas matemáticos y geométricos han existido desde la antigüedad. Sin embargo, fue tras la invención de la computadora electrónica y la expansión de uso en las ciencias, la ingeniería y las finanzas, cuando adquirieron más relevancia, ya que el carácter repetitivo de las operaciones numéricas se presta especialmente al cálculo con ordenadores. Actualmente resulta a veces más conveniente utilizar métodos numéricos incluso cuando la ecuación admite soluciones exactas.

En este tema vamos a ver las técnicas más comunes de integración de ecuaciones diferenciales ordinarias, ya sea para resolver problemas de valores iniciales o problemas de contorno. Estudiaremos las ideas que sustentan el método así como la estructura del algoritmo, de modo que el alumno pueda implementarlos en un programa, ya sea escribiendo el código desde cero o utilizando rutinas ya existentes. Lo que no vamos a ver en este tema es la implementación concreta de estos algoritmos en un determinado lenguaje de programación, lo cual se sale del ámbito de este curso. El libro ([Press et al., 2007](#)) contiene una descripción exhaustiva de los métodos que vamos a ver en este tema, así como otros más sofisticados, e incluye una serie de rutinas para cada uno de los algoritmos escritas en lenguaje C. Otras ediciones del mismo libro contienen rutinas escritas en otros lenguajes de programación como Fortran y C++ . En ([Cordero et al., 2006](#)) puedes encontrar además problemas de cálculo numérico resueltos.

Existen paquetes de software matemático y cálculo simbólico tales como Mathematica, Maple, etc. que ofrecen la posibilidad de resolver ciertos tipos de ecuaciones diferenciales. Aquí tampoco nos planteamos enseñar a utilizar un determinado paquete de software, sin embargo esperamos que los conceptos aprendidos en este curso ayuden al alumno a entender estos programas e interpretar sus resultados correctamente.

Los objetivos de este tema son:

- ▶ Comprender la estructura de los algoritmos numéricos más comunes para la resolución de problemas de valores iniciales y problemas de contorno.
- ▶ Ser capaz de implementar dichos algoritmos en un programa.
- ▶ Entender el origen de los errores en los diferentes algoritmos de integración numérica de ecuaciones diferenciales, la relación entre paso de integración, precisión del cálculo y tiempo de computación, así como los métodos para optimizar la eficiencia del cálculo.

7.2 Problemas de valores iniciales

Comenzamos con el problemas de valores iniciales para un sistema de ecuaciones ordinarias de primer orden:

$$\begin{cases} \mathbf{y}' = \mathbf{f}(x, \mathbf{y}) \\ \mathbf{y}(x_0) = \mathbf{y}_0 \end{cases}, \quad (1)$$

en el que se nos dan los valores de todas las funciones $y_i(x)$ en un determinado punto x_0 . El problema consiste en encontrar los valores de las funciones $y_i(x)$ en un punto final x_f o en una determinada serie de puntos intermedios $\{x_i\}$. Recordemos que un problema de valores iniciales para una ecuación de orden n se puede expresar como un sistema de ecuaciones de primer orden como la [Ecuación \(1\)](#) mediante un cambio de variables.

La idea general tras cualquier rutina para resolver problemas de valores iniciales es

escribir las derivadas $\frac{dy}{dx}$ como los cocientes de dos cantidades finitas $\frac{\Delta y}{\Delta x}$ y multiplicar las ecuaciones por Δx . Esto genera una serie de fórmulas algebraicas que nos dan el cambio en la variable dependiente y cuando la variable independiente x da un salto Δx . Haciendo el paso Δx suficientemente pequeño obtenemos una buena aproximación a la ecuación diferencial original. En línea con la mayor parte de la literatura sobre el tema llamaremos h al paso de integración ($\Delta x \equiv h$).

La implementación literal de este procedimiento da lugar al *método de Euler*. Este método no se recomienda para ningún uso práctico ya que existen métodos mucho más precisos por el mismo esfuerzo computacional, sin embargo es conceptualmente importante, por lo que lo describimos a continuación.

Método de Euler

Todos los métodos prácticos de resolución numérica de ecuaciones diferenciales se reducen a la misma idea: añadir a las funciones $y(x)$ pequeños incrementos Δy correspondientes la derivada de $y(x)$ (el lado derecho de la [Ecuación \(1\)](#)) multiplicados por el paso de integración h .

Si discretizamos la variable x en una serie de puntos separados por la distancia h de manera que $x_{n+1} = x_n + h$, y llamamos $y_n \equiv y(x_n)$ tenemos que la fórmula del método de Euler es:

$$y_{n+1} = y_n + hf(x_n, y_n), \quad (2)$$

la cual avanza la solución de x_n a x_{n+1} .

Esta fórmula es asimétrica ya que avanza la solución a lo largo del intervalo h pero solamente utiliza la información sobre la derivada de la función al comienzo del intervalo ([Figura 1\(a\)](#)). Este método introduce un error del orden de h^2 y no suele utilizarse en la práctica ya que no es muy preciso comparado con otros métodos con un paso de integración similar y además no es muy estable.

Método de Runge-Kutta

Ahora supongamos que en vez de evaluar $y'(x)$ al comienzo del intervalo lo hacemos en un punto intermedio, y utilizamos ese valor de la derivada para calcular el paso completo de y_n a y_{n+1} . El procedimiento, ilustrado en la [Figura 1\(b\)](#), es el siguiente:

1. Damos un paso al estilo de la [Ecuación \(2\)](#) para estimar el salto de y_n a y_{n+1} , al cual llamamos k_1 . Es decir, $k_1 = hf(x_n, y_n)$.
2. Calculamos la pendiente de la solución en el punto intermedio $x_n + \frac{h}{2}, y_n + \frac{k_1}{2}$, la cual viene dada por $y' = f(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})$.
3. Utilizamos esta pendiente para calcular el paso completo de y_n a y_{n+1} , al cual llamamos $k_2 = hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2})$.

Así llegamos a:

$$y_{n+1} = y_n + k_2 + O(h^3). \quad (3)$$

Al contrario que la fórmula de Euler, este método es simétrico y la simetrización cancela los errores de primer orden, lo cual convierte a este en un método de segundo orden. Esta técnica de integración se llama *método de Runge-Kutta de segundo orden* o *método del punto intermedio*.

Pero no hace falta quedarse en segundo orden, y podemos continuar el procedimiento eliminando los errores de mayor orden. El método más utilizado es el *Runge-Kutta de cuarto orden*, cuya fórmula es la siguiente:

$$\begin{aligned} k_1 &= hf(x_n, y_n), \\ k_2 &= hf(x_n + \frac{h}{2}, y_n + \frac{k_1}{2}), \\ k_3 &= hf(x_n + \frac{h}{2}, y_n + \frac{k_2}{2}), \\ k_4 &= hf(x_n + h, y_n + k_3), \\ y_{n+1} &= y_n + \frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6} + O(h^5). \end{aligned} \quad (4)$$

El método de Runge-Kutta de cuarto orden requiere de cuatro evaluaciones de la derivada de $y(x)$ por cada paso h . Este método será superior al método de segundo orden

(Ecuación (3)) siempre que sea posible utilizar un paso al menos el doble de largo, lo cual no siempre es posible. Por tanto, el hecho de que un algoritmo sea de mayor orden no significa necesariamente que sea más preciso, aunque normalmente lo es.

Para comprobar la precisión del cálculo podemos reducir el paso de integración a la mitad, repetir el cálculo, y comparar los resultados. Si la diferencia es menor que la precisión requerida entonces damos el cálculo por bueno. Si bien este procedimiento es fácil desde el punto de vista del usuario, desde luego no es el más rápido. En la siguiente sección veremos un procedimiento más eficiente para garantizar la precisión del cálculo.

El método de integración de Runge-Kutta es fácil de implementar y funciona prácticamente siempre, por lo que es el más utilizado en los problemas más comunes, aunque no es necesariamente el más rápido. Existen otros métodos más sofisticados como el de *Bulirsch-Stoer*, que no veremos aquí, pero es bueno saber que existen cuando necesitamos un método de integración más eficientes con un alto grado de precisión.

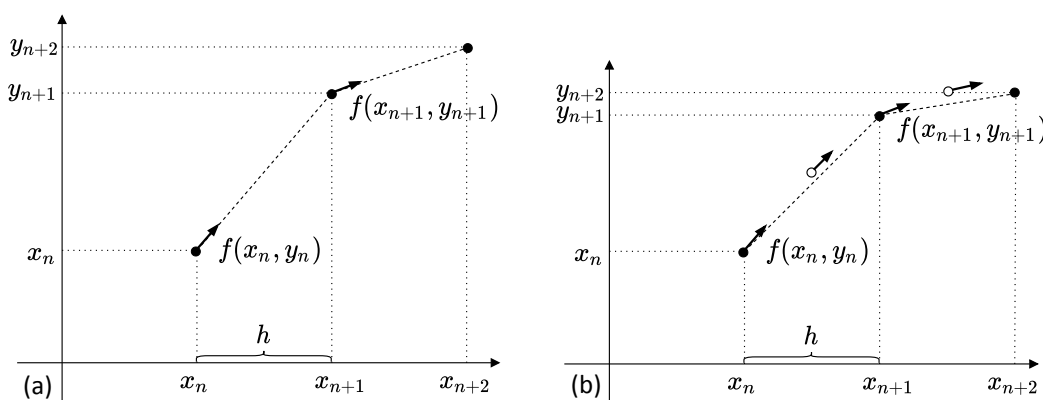


Figura 1: Representación gráfica del método de Euler (a), y de un Runge-Kutta de segundo orden (b).

El método de Runge-Kutta no exige ningún conocimiento previo sobre la solución. Todos los pasos de integración tiene la misma longitud y cada intervalo se trata de idéntica manera. Este procedimiento es matemáticamente correcto, ya que cada punto de la solución se puede utilizar como punto inicial. Además, el hecho de que todos los pasos se traten de manera idéntica lo hacen muy fácil de implementar en un programa. Sin embargo, en un programa de integración serio se considera esencial implementar algún procedimiento de control del paso de integración, de manera que adopte un

tamaño más pequeño en los intervalos en los que la función solución varía más rápidamente, y por el contrario, se pueda relajar en las zonas en las que el comportamiento de la solución es más suave. De esta manera conseguimos un balance óptimo entre la precisión y la velocidad del programa.

Otros métodos

Existen otros métodos sencillos especialmente adaptados a ecuaciones de segundo orden, que son las que aparecen con más frecuencia en la física. Por ejemplo, el *método de Verlet* para integrar ecuaciones de movimiento de la forma:

$$\ddot{\mathbf{x}} = \mathbf{f}(\mathbf{x}),$$

con condiciones iniciales $\mathbf{x}(t_0) = \mathbf{x}_0$ y $\dot{\mathbf{x}}(t_0) = \mathbf{v}_0$, sustituye la segunda derivada por:

$$\frac{\Delta^2 \mathbf{x}_n}{\Delta t^2} = \frac{\mathbf{x}_{n+1} - 2\mathbf{x}_n + \mathbf{x}_{n-1}}{\Delta t^2},$$

dando lugar a:

$$\mathbf{x}_{n+1} = 2\mathbf{x}_n - \mathbf{x}_{n-1} + \mathbf{f}(\mathbf{x}_n)\Delta t^2 + O(\Delta t^4).$$

Este método ofrece buena estabilidad numérica y preserva la reversibilidad temporal, importante en sistemas dinámicos.

El *método de Numerov* es otro método para resolver ecuaciones de segundo orden en las que no aparecen términos de primer orden:

$$y''(x) = f(x, y).$$

Este método da lugar a la fórmula:

$$y_{n+1} = 2y_n - y_{n-1} + \frac{h^2}{12} (f_{n+1} + 10f_n + f_{n-1}) + O(h^6),$$

y para ecuaciones lineales de la forma:

$$y''(x) = -g(x)y(x) + s(x),$$

el método de Numerov se reduce a:

$$y_{n+1} \left(1 + \frac{h^2}{12} g_{n+1} \right) = 2y_n \left(1 - \frac{5h^2}{12} g_n \right) - y_{n-1} \left(1 + \frac{h^2}{12} g_{n-1} \right) + \frac{h^2}{12} (s_{n+1} + 10s_n + s_{n-1}) + O(h^6).$$

Control adaptativo del paso de integración

Un buen integrador numérico de ecuaciones diferenciales ordinarias debe ejercer algún tipo de control adaptativo sobre su propio progreso, haciendo cambios frecuentes en el paso de integración. Normalmente el propósito del control adaptativo del paso de integración es conseguir un nivel de precisión predeterminado con un mínimo esfuerzo computacional. En los terrenos más abruptos será necesario dar muchos pasos pequeños, mientras que en las zonas más suaves po-

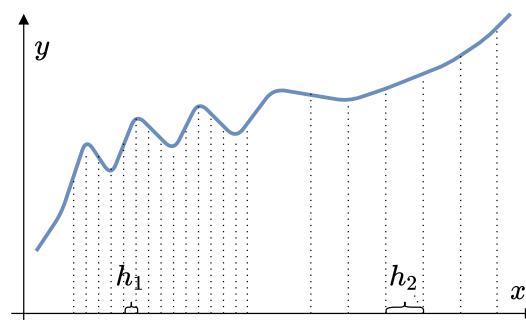


Figura 2: Integración numérica con paso variable. Elaboración propia.

dremos avanzar a zancadas (Figura 2). Este procedimiento aumenta considerablemente la eficiencia del cálculo, es decir, podemos obtener el mismo nivel de precisión que en un programa sin control adaptativo del paso pero varios órdenes de magnitud más deprisa. Para implementar un control adaptativo del paso de integración es necesario estimar el error cometido en cada paso. En el método de Runge-Kutta de cuarto orden, la manera más directa de obtener esta información es doblando el paso, es decir, tomamos cada paso dos veces, uno con un paso h y otro con dos pasos de $h/2$. La diferencia Δ entre las dos estimaciones numéricas es un buen indicador del error cometido:

$$\Delta \equiv y_2 - y_1,$$

donde y_2 e y_1 son las estimaciones de la solución con paso h y $h/2$ respectivamente. Es esta diferencia la que intentaremos mantener dentro del nivel de precisión deseado, ni muy grande ni muy pequeño, y lo hacemos ajustando el valor de h .

Según la Ecuación (4), el error Δ es proporcional a h^5 . Por tanto, si tomamos un paso

h_1 y produce un error Δ_1 , el paso h_0 que producirá un error Δ_0 es:

$$h_0 = h_1 \left| \frac{\Delta_0}{\Delta_1} \right|^{\frac{1}{5}}. \quad (5)$$

Supongamos que Δ_0 es el nivel de precisión requerido. Entonces, si Δ_1 es mayor que Δ_0 la Ecuación (5) nos dice cuánto hay que reducir el paso de integración para alcanzar la precisión deseada en el siguiente intento. Por el contrario, si Δ_1 es menor que Δ_0 la ecuación nos dice cuánto podemos aumentar el paso de integración de manera segura en el siguiente paso.

Esta notación esconde el hecho de que normalmente Δ_0 es un *vector* de precisiones deseadas (una por cada ecuación de un sistema de ecuaciones). Por lo general el requisito será que todas las ecuaciones se resuelvan dentro de sus márgenes de error permitidos, por tanto el tamaño de paso se ajustará de acuerdo a la ecuación más exigente.

7.3 Problemas de contorno

Consideramos ahora los algoritmos numéricos para resolver problemas de contorno. Los más comunes son aquellos en los que las condiciones de contorno están definidas en dos puntos, normalmente el punto inicial y el final de la integración, pero también pueden estar especificados en el interior del intervalo de integración.

La principal diferencia entre los problemas de valores iniciales y los problemas de contorno es que en el primer caso éramos capaces de comenzar con una solución aceptable (la que cumple los valores iniciales) y avanzar a lo largo del intervalo de integración hasta el final, mientras que en los problemas de contorno, las condiciones impuestas en el punto inicial no determinan una solución única y una elección «aleatoria» entre las soluciones que cumplen las condiciones impuestas en el punto inicial no satisfará por lo general las condiciones impuestas en el otro punto (o puntos).

El problema de dos puntos de contorno «estándar» tiene la forma siguiente. Buscamos

una solución a un conjunto de N ecuaciones diferenciales acopladas de primer orden, las cuales satisfacen n_1 condiciones de contorno en el primer punto del intervalo, x_1 , y las restantes $n_2 = N - n_1$ condiciones en el punto final x_2 . Recordemos que cualquier ecuación de orden N se puede reescribir como un sistema de N ecuaciones acopladas de primer orden.

Las ecuaciones son:

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_N), \quad i = 1, 2, \dots, N. \quad (6)$$

En el punto x_1 la solución debe satisfacer:

$$C_{1j}(x_1, y_1, y_2, \dots, y_N) = 0, \quad j = 1, 2, \dots, n_1, \quad (7)$$

mientras que en x_2 debe cumplir:

$$C_{2k}(x_2, y_1, y_2, \dots, y_N) = 0, \quad k = 1, 2, \dots, n_2. \quad (8)$$

Hay dos clases de métodos numéricos para resolver los problemas de dos puntos de contorno, el *método de disparo*, y los *métodos de relajación*, representados esquemáticamente en la [Figura 3](#). En el método de disparo, elegimos valores para todas las variables dependientes en uno de los puntos de contorno. Estas variables deben cumplir las condiciones impuestas en ese punto pero, por lo demás, dependerán de un parámetro independiente, cuyo valor asignamos inicialmente de manera aleatoria. A continuación integramos las ecuaciones utilizando métodos de valores iniciales, hasta llegar al otro extremo (o a cualquiera de los puntos interiores en los que se impongan condiciones). Por lo general, encontraremos que la «solución» calculada con la estimación aleatoria del parámetro libre no satisface la segunda condición de contorno. El problema consiste entonces en ajustar este parámetro hasta dar con la solución que cumpla ambas condiciones. Este procedimiento es equivalente a encontrar los ceros de un sistema multi-dimensional, es decir, encontrar los valores de los parámetros que anulan las discrepancias en el punto de contorno opuesto.

El nombre de este método se entiende si equiparamos el proceso de integración al de

encontrar la trayectoria del proyectil que acierte en un objetivo determinado desde una distancia dada. Primero lanzamos un disparo de prueba, y si bien nos pasamos de largo o bien nos quedamos cortos ajustamos la altura del «cañón» hasta que conseguimos dar en la «diana».

Otra variante del método de disparo consiste en suponer el valor de los parámetros libres a ambos lados del intervalo de integración, integrar las ecuaciones hasta un punto intermedio común, y ajustar los parámetros hasta que las soluciones se unan «suavemente» en dicho punto.

En los métodos de disparo, todas las «soluciones» de prueba satisfacen la ecuación diferencial dada, pero solamente cumplen las condiciones de contorno requeridas una vez que ha finalizado el proceso de iteración.

Los métodos de relajación utilizan una aproximación diferente. En este caso las ecuaciones diferenciales se sustituyen por ecuaciones en diferencias finitas definidas sobre una malla de puntos que cubre el intervalo de integración. Una «solución» de prueba consiste en asignar valores a las variables dependientes en cada uno de los puntos de la malla. Para empezar, no es necesario que estos puntos satisfagan las ecuaciones en diferencias finitas, ni siquiera las condiciones de contorno. La iteración, que ahora llamamos relajación, consiste en ajustar todos los valores en la malla para hacer que se vayan acercando sucesivamente a las soluciones de la ecuación en diferencias finitas, al mismo tiempo cumpliendo las condiciones de contorno. Por ejemplo, si el problema consiste en dos ecuaciones acopladas y una malla de mil puntos, tenemos que suponer y ajustar dos mil variables representando la solución.

Los métodos de relajación funcionan mejor que los de disparo cuando las condiciones de contorno son especialmente delicadas o sutiles, o cuando consisten en relaciones algebraicas complicadas que no se pueden resolver de forma cerrada. Los métodos de relajación funcionan mejor cuando la solución es suave y no oscila demasiado. Tales oscilaciones requerirían muchos puntos de malla para poderlos representar adecuadamente (ver la [Figura 2](#)). El número y posición de los puntos necesarios puede no conocerse a priori. En ese caso los métodos de disparo funcionan mejor, ya que los métodos de integración con paso variable se ajustan naturalmente a las características

de la solución. El secreto de los métodos de relajación es hacer una buena suposición inicial.

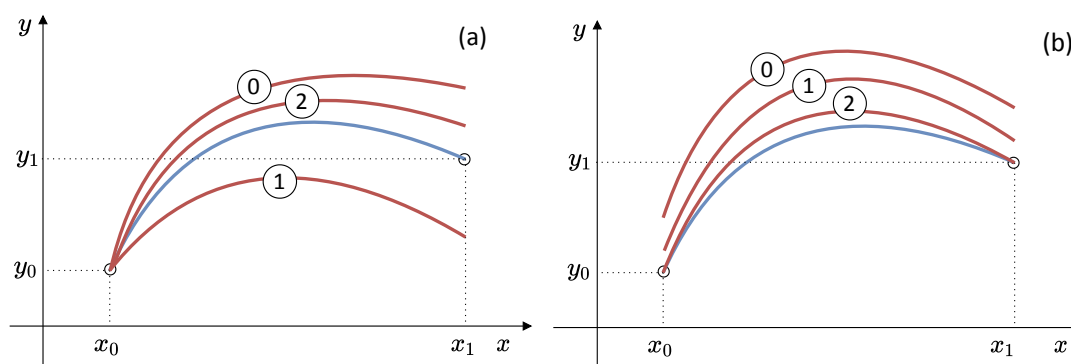


Figura 3: Ilustración de los métodos de resolución numérica de una ecuación diferencial ordinaria con condiciones de contorno $y(x_0) = y_0$ y $y(x_1) = y_1$. (a) Método de disparo; (b) Método de relajación. La trayectoria 0 es «solución» de prueba inicial, y las 1, 2, ... son las sucesivas iteraciones convergentes en la solución (trayectoria azul).

Problemas de autovalores

Los problemas de autovalores representan un caso particular que se puede reducir a un problema de contorno estándar. En este caso el lado derecho del sistema de ecuaciones depende de un parámetro λ :

$$\frac{dy_i}{dx} = f_i(x, y_1, y_2, \dots, y_N, \lambda), \quad i = 1, 2, \dots, N,$$

y uno debe satisfacer $N + 1$ condiciones de contorno en vez de solamente N .

Este problema solamente tiene solución para ciertos valores de λ , los autovalores, y se reduce al problema de contorno estándar introduciendo una nueva variable dependiente $y_{N+1} \equiv \lambda$, y una ecuación adicional:

$$\frac{dy_{N+1}}{dx} = 0,$$

que no es otra cosa que el requisito de que λ sea constante.

Veamos ahora la estructura básica de los métodos de disparo y relajación para la resolución de problemas de contorno.

Método de disparo

Supongamos que queremos resolver el siguiente problema:

$$\begin{cases} y'' = f(x, y, y') \\ y(x_0) = y_0 \\ y(x_1) = y_1 \end{cases},$$

y supongamos que $y(x, a)$ es la solución del problema de valores iniciales:

$$\begin{cases} y'' = f(x, y, y') \\ y(x_0) = y_0 \\ y'(x_0) = a \end{cases}.$$

Si $y(x_1, a) = y_1$, entonces $y(x, a)$ también es solución del problema de contorno.

El método de disparo consiste en resolver el problema de valores iniciales para valores diferentes de a hasta que uno encuentra la solución $y(x, a)$ que satisface las condiciones de contorno impuestas. Es decir, uno «dispara» en diferentes direcciones desde uno de los puntos del contorno hasta que «acierta» con el otro punto del contorno. La solución (o soluciones) corresponde a la raíz (o raíces) de la función $F(a)$:

$$F(a) = y(x_1, a) - y_1,$$

las cuales se pueden encontrar numéricamente utilizando el *método de la bisección* o el *método de Newton*.

Método de la bisección

El método de la bisección es un método numérico para encontrar de manera aproximada los ceros de una función. En el caso más sencillo de una función de una sola variable $f(x)$, suponemos que conocemos dos valores de la función con signo opuesto: $f(a_1)$ y $f(b_1)$, y que la función es continua y está definida en el intervalo $[a_1, b_1]$, por tanto la función debe tener al menos un cero en el intervalo (a_1, b_1) . El método

consiste en dividir este intervalo en dos y elegir el sub-intervalo en el que la función cambia de signo, y por tanto debe contener un cero. Este proceso se repite hasta que el intervalo es suficientemente pequeño (Figura 4).

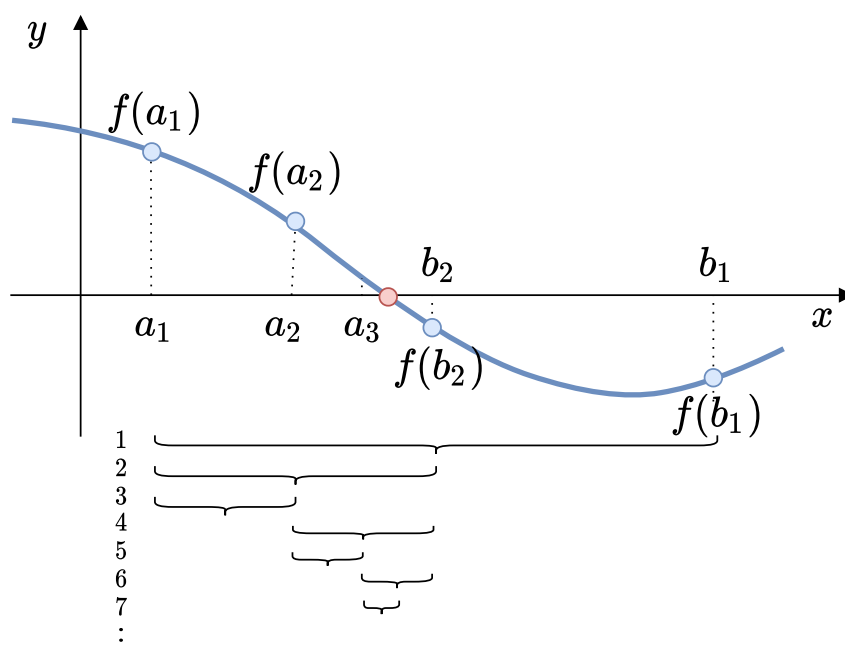


Figura 4: Método de la bisección.

Este método es el más sencillo pero también es relativamente lento. Por eso se suele utilizar para obtener una estimación del intervalo en el que se encuentra la solución, y utilizar este intervalo como punto de partida para métodos más rápidos.

Método de Newton

El método de Newton, también conocido como método de Newton-Raphson, es otro algoritmo para encontrar los ceros de una función, por lo general más rápido que el método de la bisección. La versión más sencilla comienza con una función $f(x)$, su derivada $f'(x)$, y una estimación inicial x_0 para el cero de $f(x)$. Si la estimación inicial es suficientemente buena (lo cual se puede asegurar mediante el método de la bisección) entonces el punto x_1 definido como:

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)},$$

es una mejor aproximación al cero que x_0 . El proceso se repite como:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)},$$

hasta llegar a un valor lo bastante preciso (Figura 5).

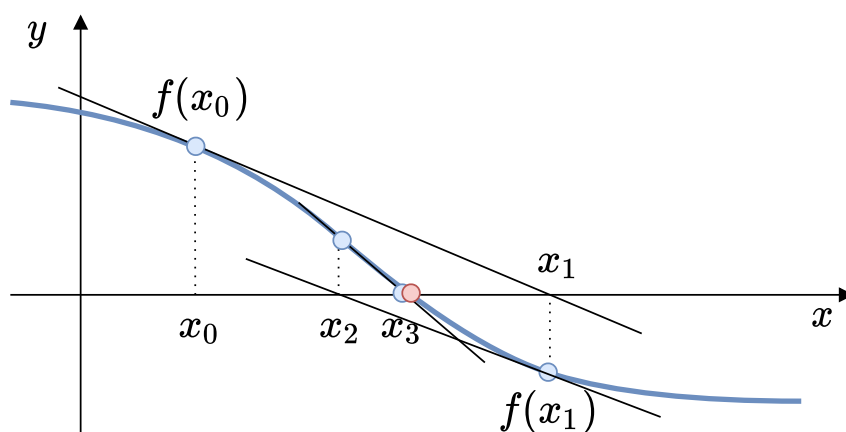


Figura 5: Método de Newton.

En este vídeo puedes ver una demostración práctica del método para resolver un problema de contorno:



Accede al vídeo: Método de disparo.

Método de relajación

En los métodos de relajación, sustituimos las ecuaciones diferenciales por ecuaciones en diferencias finitas definidas sobre una malla de puntos que cubre el dominio de interés. Por ejemplo, podemos reemplazar una ecuación general de primer orden:

$$\frac{dy}{dx} = f(x, y), \quad (9)$$

por una ecuación algebraica que relacione los valores de la función en dos puntos x_k y x_{k-1} :

$$y_k - y_{k-1} = (x_k - x_{k-1}) f\left(\frac{1}{2}(x_k + x_{k-1}), \frac{1}{2}(y_k + y_{k-1})\right). \quad (10)$$

La forma de convertir la ecuación diferencial [Ecuación \(9\)](#) en una ecuación de diferencias finitas no es única, [Ecuación \(10\)](#) es tan solo un ejemplo.

Cuando el problema implica un sistema de N ecuaciones acopladas de primer orden representadas por ecuaciones en diferencias finitas sobre una malla de M puntos, la solución consiste en valores para las N funciones dependientes en cada uno de los M puntos de la malla, es decir $N \times M$ variables en total. El método de relajación determina la solución comenzando con una función de prueba y mejorándola iterativamente. Como las iteraciones convergen hacia la solución, el resultado se dice que «relaja» en la solución verdadera.

Varios procedimientos de iteración son posibles, pero para la mayor parte de problemas el método de Newton multidimensional funciona bien.

Este método produce una ecuación de matrices que hay que resolver, pero las matrices adquieren una forma diagonal por bloques que permite invertirlas de manera más eficiente (tanto en tiempo de computación como en consumo de memoria) que si fuera una matriz general de tamaño $(MN) \times (MN)$. Como MN puede ser del orden de 10^3 , este hecho es esencial para la viabilidad del método.

Vamos a desarrollar un método general para representar ecuaciones diferenciales ordinarias como ecuaciones en diferencias finitas.

El problema que queremos resolver es el expresado por la [Ecuación \(6\)](#) con las condiciones de contorno de la [Ecuación \(7\)](#) y la [Ecuación \(8\)](#), donde tenemos N ecuaciones de primer orden acopladas que satisfacen n_1 condiciones de contorno en x_1 y $n_2 = N - n_1$ condiciones en x_2 .

Comenzamos definiendo una malla con un conjunto de puntos $k = 1, 2, \dots, M$ a los cuales asignamos valores de la variable independiente x_k . En particular, x_1 es el contorno inicial y x_M es el contorno final. Utilizamos la notación y_k para referirnos al conjunto completo de variables dependientes y_1, y_2, \dots, y_N en el punto x_k . En un punto arbitrario k en el interior de la malla aproximamos el conjunto de N ecuaciones dife-

renciales de primer orden por relaciones algebraicas de la forma:

$$\underbrace{y_k - y_{k-1} - (x_k - x_{k-1})f_k(x_k, x_{k-1}, y_k, y_{k-1})}_{E_k} = 0 \Rightarrow \quad (11)$$

$$E_k = 0, \quad k = 2, 3, \dots, M.$$

La notación indica que f_k se puede evaluar usando la información de dos puntos consecutivos x_k, x_{k-1} . Las ecuaciones en diferencias finitas designadas como E_k proporcionan N ecuaciones acoplando $2N$ variables en los puntos $k, k-1$. Existen $M-1$ puntos ($k = 2, 3, \dots, M$), en los cuales se pueden aplicar las ecuaciones de la forma de la [Ecuación \(11\)](#). Por tanto, las ecuaciones en diferencias finitas proporcionan un total de $(M-1)N$ ecuaciones para las MN incógnitas. Las N ecuaciones restantes provienen de las condiciones de contorno.

En el primer punto del contorno tenemos:

$$0 = \mathbf{E}_1 \equiv \mathbf{C}_1(x_1, \mathbf{y}_1), \quad (12)$$

y en el segundo:

$$0 = \mathbf{E}_{M+1} \equiv \mathbf{C}_2(x_M, \mathbf{y}_M). \quad (13)$$

Los vectores \mathbf{E}_1 y \mathbf{C}_1 tienen solamente n_1 componentes no-nulas, correspondientes a las n_1 condiciones de contorno en x_1 . Resulta conveniente hacer que estas componentes no-nulas sean las últimas n_1 componentes. Es decir, $E_{j,1} \neq 0$ solo para $j = n_2 + 1, n_2 + 2, \dots, N$. En el otro extremo, solo las primeras n_2 componentes de \mathbf{E}_{M+1} y \mathbf{C}_2 son distintas de cero: $E_{j,M+1} \neq 0$ solo para $j = 1, 2, \dots, n_2$.

La «solución» del problema de diferencias finitas expresado por la [Ecuación \(11\)](#), [Ecuación \(12\)](#) y [Ecuación \(13\)](#), consiste en un conjunto de variables $y_{j,k}$, esto es, los valores de las N variables y_j en los M puntos x_k de la malla. El algoritmo que describimos más abajo requiere una suposición inicial para las $y_{j,k}$. A continuación determinamos los incrementos $\Delta y_{j,k}$ de manera que $y_{j,k} + \Delta y_{j,k}$ es una aproximación mejorada a la solución.

Las ecuaciones para los incrementos se obtienen desarrollando las ecuaciones de dife-

rencias finitas en series de Taylor hasta primer orden respecto a pequeñas variaciones Δy_k . En un punto interior $k = 2, 3, \dots, M$ se obtiene:

$$\mathbf{E}_k(\mathbf{y}_k + \Delta \mathbf{y}_k, \mathbf{y}_{k-1} + \Delta \mathbf{y}_{k-1}) \approx \mathbf{E}_k(\mathbf{y}_k, \mathbf{y}_{k-1}) + \sum_{n=1}^N \frac{\partial \mathbf{E}_k}{\partial y_{n,k-1}} \Delta y_{n,k-1} + \sum_{n=1}^N \frac{\partial \mathbf{E}_k}{\partial y_{n,k}} \Delta y_{n,k}.$$

Para llegar a la solución queremos que el valor actualizado $\mathbf{E}(\mathbf{y} + \Delta \mathbf{y})$ sea cero, de manera que el conjunto de ecuaciones en un punto interior se puede escribir en forma de matriz como:

$$\sum_{n=1}^N S_{j,n} \Delta y_{n,k-1} + \sum_{n=N+1}^{2N} S_{j,n} \Delta y_{n-N,k} = -E_{j,k}, \quad j = 1, 2, \dots, N, \quad (14)$$

donde:

$$S_{j,n} = \frac{\partial E_{j,k}}{\partial y_{n,k-1}}, \quad S_{j,n+N} = \frac{\partial E_{j,k}}{\partial y_{n,k}}, \quad n = 1, 2, \dots, N. \quad (15)$$

La cantidad $S_{j,n}$ es una matriz $N \times 2N$ en cada punto k . Cada punto interior por tanto proporciona un bloque de N ecuaciones acoplando $2N$ correcciones a las variables de la solución en los puntos $k, k-1$.

De manera similar, las relaciones algebraicas en los puntos de contorno se pueden desarrollar en series de Taylor hasta primer orden para obtener incrementos que mejoren la «solución». Como E_1 depende solo de y_1 , obtenemos para el primer punto de contorno:

$$\sum_{n=1}^N S_{j,n} \Delta y_{n,1} = -E_{j,1}, \quad j = n_2 + 1, n_2 + 2, \dots, N, \quad (16)$$

donde:

$$S_{j,n} = \frac{\partial E_{j,1}}{\partial y_{n,1}}, \quad n = 1, 2, \dots, N, \quad (17)$$

y para el otro extremo,

$$\sum_{n=1}^N S_{j,n} \Delta y_{n,M} = -E_{j,M+1}, \quad j = 1, 2, \dots, n_2, \quad (18)$$

donde:

$$S_{j,n} = \frac{\partial E_{j,M+1}}{\partial y_{n,M}}, \quad n = 1, 2, \dots, N. \quad (19)$$

De esta manera, tenemos en la [Ecuación \(14\)](#), [Ecuación \(15\)](#), [Ecuación \(16\)](#), [Ecuación \(17\)](#), [Ecuación \(18\)](#), y [Ecuación \(19\)](#) un conjunto de ecuaciones algebraicas lineales que deben resolverse para las correcciones Δy de manera iterativa hasta que las correcciones son lo suficientemente pequeñas.

7.4 Referencias bibliográficas

Cordero, A., Hueso, J. L., Martínez, E., & Torregrosa, J. R. (2006). *Problemas resueltos de métodos numéricos*. Thomson.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. (2007). *Numerical recipes 3rd edition: The art of scientific computing*. Cambridge university press.