Simulación de Sistemas Continuos y a Tramos

Prof. Dr. François E. Cellier
Institut für Computational Science
ETH Zürich

26 de junio 2007

Introducción

Introducción

Hasta ahora vimos *algoritmos para la simulación numérica de sistemas dinámicos en un solo paso*. Toda la información que el algoritmo necesitaba lo obtenía localmente.

Introducción

Hasta ahora vimos *algoritmos para la simulación numérica de sistemas dinámicos en un solo paso*. Toda la información que el algoritmo necesitaba lo obtenía localmente.

El precio que pagamos era que los métodos avanzados de órdenes elevados necesitaban varias evaluaciones del modelo durante el paso. Hablamos de algoritmos con varias "etapas".

Introducción

Hasta ahora vimos *algoritmos para la simulación numérica de sistemas dinámicos en un solo paso*. Toda la información que el algoritmo necesitaba lo obtenía localmente.

El precio que pagamos era que los métodos avanzados de órdenes elevados necesitaban varias evaluaciones del modelo durante el paso. Hablamos de algoritmos con varias "etapas".

Quizás esa manera de resolver el problema sea ineficiente. Al final de cada paso tenemos mucha información que podríamos utilizar durante el cálculo asociado al próximo paso.

Introducción

Hasta ahora vimos *algoritmos para la simulación numérica de sistemas dinámicos en un solo paso*. Toda la información que el algoritmo necesitaba lo obtenía localmente.

El precio que pagamos era que los métodos avanzados de órdenes elevados necesitaban varias evaluaciones del modelo durante el paso. Hablamos de algoritmos con varias "etapas".

Quizás esa manera de resolver el problema sea ineficiente. Al final de cada paso tenemos mucha información que podríamos utilizar durante el cálculo asociado al próximo paso.

Existen otros métodos avanzados de órdenes elevados que utilizan información del pasado y que pueden efectuar simulaciones muy precisas con una sola evaluación del modelo en cada paso. Esos algoritmos se llaman *métodos lineales de la integración numérica en múltiples pasos*.

Los Polinomios de Newton-Gregory

Para analizar *métodos en múltiples pasos* necesitamos unas herramientas que introducimos ahora.

Los Polinomios de Newton-Gregory

Para analizar *métodos en múltiples pasos* necesitamos unas herramientas que introducimos ahora.

Tenemos una función $f(t)$ con muestreo en los instantes de tiempo, t_0, $t_1 > t_0$, $t_2 > t_1$, $\ldots$. La función asume los valores f_0, f_1, f_2, $\ldots$ en estos momentos.

Los Polinomios de Newton-Gregory

Para analizar *métodos en múltiples pasos* necesitamos unas herramientas que introducimos ahora.

Tenemos una función $f(t)$ con muestreo en los instantes de tiempo, t_0, $t_1 > t_0$, $t_2 > t_1$, $\ldots$. La función asume los valores f_0, f_1, f_2, $\ldots$ en estos momentos.

Empezamos con la introducción del *operador de la diferencia hacia adelante*, Δ:

$$\Delta f_0 = f_1 - f_0$$
$$\Delta^2 f_0 = \Delta(\Delta f_0) = \Delta(f_1 - f_0) = \Delta f_1 - \Delta f_0 = f_2 - 2f_1 + f_0$$
$$\Delta^3 f_0 = \Delta(\Delta^2 f_0) = f_3 - 3f_2 + 3f_1 - f_0$$
$$\text{etc.}$$

Los Polinomios de Newton-Gregory

Para analizar *métodos en múltiples pasos* necesitamos unas herramientas que introducimos ahora.

Tenemos una función $f(t)$ con muestreo en los instantes de tiempo, t_0, $t_1 > t_0$, $t_2 > t_1$, ... La función asume los valores f_0, f_1, f_2, ... en estos momentos.

Empezamos con la introducción del *operador de la diferencia hacia adelante*, Δ:

$$\begin{aligned}
\Delta f_0 &= f_1 - f_0 \\
\Delta^2 f_0 &= \Delta(\Delta f_0) = \Delta(f_1 - f_0) = \Delta f_1 - \Delta f_0 = f_2 - 2f_1 + f_0 \\
\Delta^3 f_0 &= \Delta(\Delta^2 f_0) = f_3 - 3f_2 + 3f_1 - f_0 \\
&\text{etc.}
\end{aligned}$$

En general:

$$\begin{aligned}
\Delta^n f_i &= f_{i+n} - n \cdot f_{i+n-1} + \frac{n(n-1)}{2!} \cdot f_{i+n-2} - \frac{n(n-1)(n-2)}{3!} \cdot f_{i+n-3} + \dots \\
&= \binom{n}{0} f_{i+n} - \binom{n}{1} f_{i+n-1} + \binom{n}{2} f_{i+n-2} - \binom{n}{3} f_{i+n-3} + \dots \pm \binom{n}{n} f_i
\end{aligned}$$

Los Polinomios de Newton-Gregory II

Suponiendo ahora un *muestreo equidistante*, es decir $t_1 = t_0 + h$, $t_2 = t_0 + 2h$, $\ldots$, $t_n = t_0 + n \cdot h$.

Los Polinomios de Newton-Gregory II

Suponiendo ahora un *muestreo equidistante*, es decir $t_1 = t_0 + h$, $t_2 = t_0 + 2h$, $\ldots$, $t_n = t_0 + n \cdot h$.

Introducimos una *variable de tiempo normalizada*, s:

$$s = \frac{t - t_0}{h}$$

de tal manera que $t = t_0 \Leftrightarrow s = 0.0$, $t = t_1 \Leftrightarrow s = 1.0$, $\ldots$.

Los Polinomios de Newton-Gregory II

Suponiendo ahora un *muestreo equidistante*, es decir $t_1 = t_0 + h$, $t_2 = t_0 + 2h$, $\ldots$, $t_n = t_0 + n \cdot h$.

Introducimos una *variable de tiempo normalizada*, s:

$$s = \frac{t - t_0}{h}$$

de tal manera que $t = t_0 \Leftrightarrow s = 0.0$, $t = t_1 \Leftrightarrow s = 1.0$, $\ldots$.

Se puede definir un *polinomio de interpolación* de orden n que pasa por los $n + 1$ puntos f_0, f_1, $\ldots$, f_n:

$$f(s) \approx \binom{s}{0} f_0 + \binom{s}{1} \Delta f_0 + \binom{s}{2} \Delta^2 f_0 + \cdots + \binom{s}{n} \Delta^n f_0$$

Los Polinomios de Newton-Gregory II

Suponiendo ahora un *muestreo equidistante*, es decir $t_1 = t_0 + h$, $t_2 = t_0 + 2h$, $\ldots$, $t_n = t_0 + n \cdot h$.

Introducimos una *variable de tiempo normalizada*, s:

$$s = \frac{t - t_0}{h}$$

de tal manera que $t = t_0 \Leftrightarrow s = 0.0$, $t = t_1 \Leftrightarrow s = 1.0$, $\ldots$.

Se puede definir un *polinomio de interpolación* de orden n que pasa por los $n + 1$ puntos f_0, f_1, $\ldots$, f_n:

$$f(s) \approx \binom{s}{0} f_0 + \binom{s}{1} \Delta f_0 + \binom{s}{2} \Delta^2 f_0 + \cdots + \binom{s}{n} \Delta^n f_0$$

Ese polinomio se llama el *polinomio de interpolación de Newton-Gregory hacia adelante*. Es muy fácil mostrar que ese polinomio de orden n pasa por los $n+1$ puntos f_0, f_1, $\ldots$, f_n.

Los Polinomios de Newton-Gregory III

Es importante mencionar que la variable s puede asumir *valores reales* no solamente *valores enteros*. Por ejemplo:

$$\binom{s}{3}_{s=1.5} \equiv \left[\frac{s(s-1)(s-2)}{3!}\right]_{s=1.5} = -\frac{1}{16}$$

Los Polinomios de Newton-Gregory III

Es importante mencionar que la variable s puede asumir *valores reales* no solamente *valores enteros*. Por ejemplo:

$$\binom{s}{3}_{s=1.5} \equiv \left[\frac{s(s-1)(s-2)}{3!}\right]_{s=1.5} = -\frac{1}{16}$$

A veces es más útil trabajar con otro polinomio de interpolación:

$$f(s) \approx f_0 + \binom{s}{1}\Delta f_{-1} + \binom{s+1}{2}\Delta^2 f_{-2} + \binom{s+2}{3}\Delta^3 f_{-3} + \cdots + \binom{s+n-1}{n}\Delta^n f_{-n}$$

Los Polinomios de Newton-Gregory III

Es importante mencionar que la variable s puede asumir *valores reales* no solamente *valores enteros*. Por ejemplo:

$$\binom{s}{3}_{s=1.5} \equiv \left[\frac{s(s-1)(s-2)}{3!}\right]_{s=1.5} = -\frac{1}{16}$$

A veces es más útil trabajar con otro polinomio de interpolación:

$$f(s) \approx f_0 + \binom{s}{1}\Delta f_{-1} + \binom{s+1}{2}\Delta^2 f_{-2} + \binom{s+2}{3}\Delta^3 f_{-3} + \cdots + \binom{s+n-1}{n}\Delta^n f_{-n}$$

Ese polinomio se llama el *polinomio de interpolación de Newton-Gregory hacia atrás*. Es igualmente fácil mostrar que ese polinomio de orden n pasa por los $n+1$ puntos f_0, f_{-1}, $\ldots$, f_{-n}.

Los Polinomios de Newton-Gregory IV

Introducimos ahora un segundo operador, el *operador de la diferencia hacia atrás*, ∇:

$$\nabla f_i = f_i - f_{i-1}$$
$$\nabla^2 f_i = \nabla(\nabla f_i) = \nabla(f_i - f_{i-1}) = \nabla f_i - \nabla f_{i-1} = f_i - 2\ f_{i-1} + f_{i-2}$$
$$\nabla^3 f_i = \nabla(\nabla^2 f_i) = f_i - 3f_{i-1} + 3f_{i-2} - f_{i-3}$$
$$\text{etc.}$$

Los Polinomios de Newton-Gregory IV

Introducimos ahora un segundo operador, el *operador de la diferencia hacia atrás*, ∇:

$$\begin{aligned}
\nabla f_i &= f_i - f_{i-1} \\
\nabla^2 f_i &= \nabla(\nabla f_i) = \nabla(f_i - f_{i-1}) = \nabla f_i - \nabla f_{i-1} = f_i - 2\, f_{i-1} + f_{i-2} \\
\nabla^3 f_i &= \nabla(\nabla^2 f_i) = f_i - 3f_{i-1} + 3f_{i-2} - f_{i-3} \\
&\text{etc.}
\end{aligned}$$

En general:

$$\nabla^n f_i = \binom{n}{0} f_i - \binom{n}{1} f_{i-1} + \binom{n}{2} f_{i-2} - \binom{n}{3} f_{i-3} + \cdots \pm \binom{n}{n} f_{i-n}$$

Los Polinomios de Newton-Gregory IV

Introducimos ahora un segundo operador, el *operador de la diferencia hacia atrás*, ∇:

$$\begin{aligned}
\nabla f_i &= f_i - f_{i-1} \\
\nabla^2 f_i &= \nabla(\nabla f_i) = \nabla(f_i - f_{i-1}) = \nabla f_i - \nabla f_{i-1} = f_i - 2\, f_{i-1} + f_{i-2} \\
\nabla^3 f_i &= \nabla(\nabla^2 f_i) = f_i - 3f_{i-1} + 3f_{i-2} - f_{i-3} \\
&\text{etc.}
\end{aligned}$$

En general:

$$\nabla^n f_i = \binom{n}{0} f_i - \binom{n}{1} f_{i-1} + \binom{n}{2} f_{i-2} - \binom{n}{3} f_{i-3} + \cdots \pm \binom{n}{n} f_{i-n}$$

El *polinomio de interpolación de Newton-Gregory hacia atrás* también puede expresarse usando el operador ∇:

$$f(s) \approx f_0 + \binom{s}{1} \nabla f_0 + \binom{s+1}{2} \nabla^2 f_0 + \binom{s+2}{3} \nabla^3 f_0 + \cdots + \binom{s+n-1}{n} \nabla^n f_0$$

Los Polinomios de Newton-Gregory V

También es útil otro operador más, el *operador del desplazamiento*, $\mathcal{E}$:

$$\mathcal{E} f_i = f_{i+1}$$
$$\mathcal{E}^2 f_i = \mathcal{E}(\mathcal{E} f_i) = \mathcal{E}(f_{i+1}) = f_{i+2}$$
$$\mathcal{E}^3 f_i = \mathcal{E}(\mathcal{E}^2 f_i) = \mathcal{E}(f_{i+2}) = f_{i+3}$$
$$\text{etc.}$$

Los Polinomios de Newton-Gregory V

También es útil otro operador más, el *operador del desplazamiento*, $\mathcal{E}$:

$$\begin{aligned}
\mathcal{E} f_i &= f_{i+1} \\
\mathcal{E}^2 f_i &= \mathcal{E}(\mathcal{E} f_i) = \mathcal{E}(f_{i+1}) = f_{i+2} \\
\mathcal{E}^3 f_i &= \mathcal{E}(\mathcal{E}^2 f_i) = \mathcal{E}(f_{i+2}) = f_{i+3} \\
&\text{etc.}
\end{aligned}$$

Obviamente:

$$\begin{aligned}
\Delta f_i &= \mathcal{E} f_i - f_i = (\mathcal{E} - 1) f_i \\
\nabla f_i &= f_i - \mathcal{E}^{-1} f_i = (1 - \mathcal{E}^{-1}) f_i \\
\mathcal{E}(\nabla f_i) &= \mathcal{E}(f_i - f_{i-1}) = f_{i+1} - f_i = \Delta f_i
\end{aligned}$$

Los Polinomios de Newton-Gregory V

También es útil otro operador más, el *operador del desplazamiento*, $\mathcal{E}$:

$$\begin{aligned}\mathcal{E}f_i &= f_{i+1}\\ \mathcal{E}^2 f_i &= \mathcal{E}(\mathcal{E}f_i) = \mathcal{E}(f_{i+1}) = f_{i+2}\\ \mathcal{E}^3 f_i &= \mathcal{E}(\mathcal{E}^2 f_i) = \mathcal{E}(f_{i+2}) = f_{i+3}\\ &\text{etc.}\end{aligned}$$

Obviamente:

$$\begin{aligned}\Delta f_i &= \mathcal{E}f_i - f_i = (\mathcal{E}-1)f_i\\ \nabla f_i &= f_i - \mathcal{E}^{-1}f_i = (1-\mathcal{E}^{-1})f_i\\ \mathcal{E}(\nabla f_i) &= \mathcal{E}(f_i - f_{i-1}) = f_{i+1} - f_i = \Delta f_i\end{aligned}$$

Abstrayendo:

$$\begin{aligned}\Delta &= \mathcal{E}-1\\ \nabla &= 1-\mathcal{E}^{-1}\\ \Delta &= \mathcal{E}\nabla\end{aligned}$$

Los Polinomios de Newton-Gregory VI

Los tres operadores Δ, ∇ y $\mathcal{E}$ son *operadores lineales*. Entonces se pueden usar en expresiones algebraicas. En particular:

$$\Delta^n = (\mathcal{E} - 1)^n = \mathcal{E}^n - n\mathcal{E}^{n-1} + \binom{n}{2}\mathcal{E}^{n-2} - + \cdots \pm \binom{n}{n-1}\mathcal{E} \mp 1$$

Los Polinomios de Newton-Gregory VI

Los tres operadores Δ, ∇ y $\mathcal{E}$ son *operadores lineales*. Entonces se pueden usar en expresiones algebraicas. En particular:

$$\Delta^n = (\mathcal{E} - 1)^n = \mathcal{E}^n - n\mathcal{E}^{n-1} + \binom{n}{2}\mathcal{E}^{n-2} - + \cdots \pm \binom{n}{n-1}\mathcal{E} \mp 1$$

Usando el cálculo de operadores, la derivación de los polinomios de Newton-Gregory se simplifica:

$$f(s) \approx \mathcal{E}^s f_0 = (1 + \Delta)^s f_0 = \left[1 + \binom{s}{1}\Delta + \binom{s}{2}\Delta^2 + \binom{s}{3}\Delta^3 + \ldots\right] f_0$$

y:

$$f(s) \approx (1 - \nabla)^{-s} f_0 = \left[1 + \binom{s}{1}\nabla + \binom{s+1}{2}\nabla^2 + \binom{s+2}{3}\nabla^3 + \ldots\right] f_0$$

Los Polinomios de Newton-Gregory VII

También la *diferenciación* es una *operación lineal*. Entonces:

$$\begin{aligned} \dot{f}(t) &= \frac{d}{dt} f(t) = \frac{\partial}{\partial s} f(s) \cdot \frac{ds}{dt} \\ &\approx \frac{1}{h} \cdot \frac{\partial}{\partial s} \left(f_0 + s\Delta f_0 + \frac{s(s-1)}{2!} \Delta^2 f_0 + \dots \right) \end{aligned}$$

En particular:

$$\dot{f}(t_0) \approx \frac{1}{h} \cdot \left(\Delta f_0 - \frac{1}{2} \Delta^2 f_0 + \frac{1}{3} \Delta^3 f_0 - \dots \pm \frac{1}{n} \Delta^n f_0 \right)$$

Los Polinomios de Newton-Gregory VII

También la *diferenciación* es una *operación lineal*. Entonces:

$$\begin{aligned} \dot{f}(t) &= \frac{d}{dt} f(t) = \frac{\partial}{\partial s} f(s) \cdot \frac{ds}{dt} \\ &\approx \frac{1}{h} \cdot \frac{\partial}{\partial s} \left(f_0 + s\Delta f_0 + \frac{s(s-1)}{2!} \Delta^2 f_0 + \dots \right) \end{aligned}$$

En particular:

$$\dot{f}(t_0) \approx \frac{1}{h} \cdot \left(\Delta f_0 - \frac{1}{2} \Delta^2 f_0 + \frac{1}{3} \Delta^3 f_0 - \dots \pm \frac{1}{n} \Delta^n f_0 \right)$$

Tiene sentido introducir otro operador más, el *operador de la diferenciación*, $\mathcal{D}$:

$$\mathcal{D} = \frac{1}{h} \cdot \left(\Delta - \frac{1}{2} \Delta^2 + \frac{1}{3} \Delta^3 - \dots \pm \frac{1}{n} \Delta^n \right)$$

Los Polinomios de Newton-Gregory VII

También la *diferenciación* es una *operación lineal*. Entonces:

$$\begin{aligned} \dot{f}(t) &= \frac{d}{dt} f(t) = \frac{\partial}{\partial s} f(s) \cdot \frac{ds}{dt} \\ &\approx \frac{1}{h} \cdot \frac{\partial}{\partial s} \left(f_0 + s\Delta f_0 + \frac{s(s-1)}{2!} \Delta^2 f_0 + \dots \right) \end{aligned}$$

En particular:

$$\dot{f}(t_0) \approx \frac{1}{h} \cdot \left(\Delta f_0 - \frac{1}{2}\Delta^2 f_0 + \frac{1}{3}\Delta^3 f_0 - \dots \pm \frac{1}{n}\Delta^n f_0 \right)$$

Tiene sentido introducir otro operador más, el *operador de la diferenciación*, $\mathcal{D}$:

$$\mathcal{D} = \frac{1}{h} \cdot \left(\Delta - \frac{1}{2}\Delta^2 + \frac{1}{3}\Delta^3 - \dots \pm \frac{1}{n}\Delta^n \right)$$

Entonces, la segunda derivada puede obtenerse de la manera siguiente:

$$\begin{aligned} \mathcal{D}^2 &= \frac{1}{h^2} \cdot \left(\Delta - \frac{1}{2}\Delta^2 + \frac{1}{3}\Delta^3 - \dots \pm \frac{1}{n}\Delta^n \right)^2 \\ &= \frac{1}{h^2} \cdot \left(\Delta^2 - \Delta^3 + \frac{11}{12}\Delta^4 - \frac{5}{6}\Delta^5 + \dots \right) \end{aligned}$$

Los Métodos Lineales de Integración en Múltiples Pasos

Todas las familias de métodos lineales en múltiples pasos para la integración numérica que se usan en la simulación de sistemas dinámicos pueden derivarse de forma elegante recurriendo a los polinomios de Newton-Gregory.

Para ello aproximamos la función misma por un polinomio de Newton-Gregory y diferenciamos ese polinomio con el tiempo o, al revés, aproximamos la primera derivada por un polinomio de Newton-Gregory e integramos ese polinomio sobre el tiempo.

Las Fórmulas Explícitas de Adams-Bashforth

Formulamos un *polinomio de Newton-Gregory hacia atrás* de la derivada $\dot{\mathbf{x}}$ alrededor el instante en el tiempo t_k:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots$$

con:

$$\mathbf{f}_k = \dot{\mathbf{x}}(t_k) = \mathbf{f}(\mathbf{x}(t_k), t_k)$$

Las Fórmulas Explícitas de Adams-Bashforth

Formulamos un *polinomio de Newton-Gregory hacia atrás* de la derivada $\dot{\mathbf{x}}$ alrededor el instante en el tiempo t_k:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots$$

con:

$$\mathbf{f}_k = \dot{\mathbf{x}}(t_k) = \mathbf{f}(\mathbf{x}(t_k), t_k)$$

Integramos ese polinomio en el intervalo $t \in [t_k, t_{k+1}]$:

$$\int_{t_k}^{t_{k+1}} \dot{\mathbf{x}}(t)dt = \mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)$$

$$= \int_{t_k}^{t_{k+1}} \left[\mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots\right] dt$$

$$= \int_{0.0}^{1.0} \left[\mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots\right] \cdot \frac{dt}{ds} \cdot ds$$

Las Fórmulas Explícitas de Adams-Bashforth II

Entonces:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h \int_0^1 \left[\mathbf{f}_k + s\nabla\mathbf{f}_k + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{f}_k + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{f}_k + \ldots \right] ds$$

y por eso:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h\left(\mathbf{f}_k + \frac{1}{2}\nabla\mathbf{f}_k + \frac{5}{12}\nabla^2\mathbf{f}_k + \frac{3}{8}\nabla^3\mathbf{f}_k + \ldots\right)$$

Las Fórmulas Explícitas de Adams-Bashforth II

Entonces:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h \int_0^1 \left[\mathbf{f}_k + s\nabla\mathbf{f}_k + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{f}_k + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{f}_k + \ldots \right] ds$$

y por eso:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h\left(\mathbf{f}_k + \frac{1}{2}\nabla\mathbf{f}_k + \frac{5}{12}\nabla^2\mathbf{f}_k + \frac{3}{8}\nabla^3\mathbf{f}_k + \ldots\right)$$

Si truncamos después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(23\mathbf{f}_k - 16\mathbf{f}_{k-1} + 5\mathbf{f}_{k-2}\right)$$

el bien conocido *algoritmo de Adams-Bashforth de tercer orden (AB3)*.

Las Fórmulas Explícitas de Adams-Bashforth II

Entonces:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h \int_0^1 \left[\mathbf{f}_k + s\nabla\mathbf{f}_k + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{f}_k + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{f}_k + \ldots \right] ds$$

y por eso:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h\left(\mathbf{f}_k + \frac{1}{2}\nabla\mathbf{f}_k + \frac{5}{12}\nabla^2\mathbf{f}_k + \frac{3}{8}\nabla^3\mathbf{f}_k + \ldots\right)$$

Si truncamos después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(23\mathbf{f}_k - 16\mathbf{f}_{k-1} + 5\mathbf{f}_{k-2}\right)$$

el bien conocido *algoritmo de Adams-Bashforth de tercer orden (AB3)*.

Si truncamos solamente después del término cúbico obtenemos:

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

es decir el *algoritmo de Adams-Bashforth de cuarto orden (AB4)*.

Las Fórmulas Explícitas de Adams-Bashforth III

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.
- Cada uno usa *una sola evaluación del modelo* para cada paso.

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.
- Cada uno usa *una sola evaluación del modelo* para cada paso.
- Los algoritmos AB utilizan *información sobre las derivadas del pasado*. El algoritmo AB del orden n usa $n-1$ derivadas de pasos anteriores.

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.
- Cada uno usa *una sola evaluación del modelo* para cada paso.
- Los algoritmos AB utilizan *información sobre las derivadas del pasado*. El algoritmo AB del orden n usa $n-1$ derivadas de pasos anteriores.
- El algoritmo AB del orden n puede usarse solamente después de $n-1$ pasos anteriores. Es decir, esos algoritmos, salvo el algoritmo AB1, no pueden usarse inmediatamente.

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.
- Cada uno usa *una sola evaluación del modelo* para cada paso.
- Los algoritmos AB utilizan *información sobre las derivadas del pasado*. El algoritmo AB del orden n usa $n-1$ derivadas de pasos anteriores.
- El algoritmo AB del orden n puede usarse solamente después de $n-1$ pasos anteriores. Es decir, esos algoritmos, salvo el algoritmo AB1, no pueden usarse inmediatamente.
- El *control del paso* es complicado por la necesidad de usar información del pasado. Recordemos que los polinomios de Newton-Gregory fueron desarrollados bajo la suposición de un *muestreo equidistante*.

Las Fórmulas Explícitas de Adams-Bashforth III

- Los *algoritmos del tipo Adams-Bashforth* son *algoritmos explícitos*.
- Cada uno usa *una sola evaluación del modelo* para cada paso.
- Los algoritmos AB utilizan *información sobre las derivadas del pasado*. El algoritmo AB del orden n usa $n-1$ derivadas de pasos anteriores.
- El algoritmo AB del orden n puede usarse solamente después de $n-1$ pasos anteriores. Es decir, esos algoritmos, salvo el algoritmo AB1, no pueden usarse inmediatamente.
- El *control del paso* es complicado por la necesidad de usar información del pasado. Recordemos que los polinomios de Newton-Gregory fueron desarrollados bajo la suposición de un *muestreo equidistante*.
- Las fórmulas AB fueron derivadas bajo la *suposición de linealidad*. No es cierto que AB3 también es un algoritmo de tercer orden en la simulación de un sistema no lineal.

Las Fórmulas Explícitas de Adams-Bashforth IV

Los algoritmos AB pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 2 & 12 & 24 & 720 & 1440 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & -1 & 0 & 0 & 0 & 0 \\ 23 & -16 & 5 & 0 & 0 & 0 \\ 55 & -59 & 37 & -9 & 0 & 0 \\ 1901 & -2774 & 2616 & -1274 & 251 & 0 \\ 4277 & -7923 & 9982 & -7298 & 2877 & -475 \end{pmatrix}$$

Cada regla especifica uno de los algoritmos.

Las Fórmulas Explícitas de Adams-Bashforth IV

Los algoritmos AB pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 2 & 12 & 24 & 720 & 1440 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & -1 & 0 & 0 & 0 & 0 \\ 23 & -16 & 5 & 0 & 0 & 0 \\ 55 & -59 & 37 & -9 & 0 & 0 \\ 1901 & -2774 & 2616 & -1274 & 251 & 0 \\ 4277 & -7923 & 9982 & -7298 & 2877 & -475 \end{pmatrix}$$

Cada regla especifica uno de los algoritmos.

El *algoritmo AB1* es el algoritmo:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{1}(1\mathbf{f}_k)$$

es decir, se trata del *algoritmo FE*.

El Dominio de la Estabilidad

Queremos encontrar el *dominio de la estabilidad del algoritmo AB3*:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(23\mathbf{f}_k - 16\mathbf{f}_{k-1} + 5\mathbf{f}_{k-2}\right)$$

Aplicamos ese algoritmo a nuestro sistema lineal:

$$\mathbf{x}(t_{k+1}) = \left[\mathbf{I}^{(\mathbf{n})} + \frac{23}{12}\mathbf{A}h\right] \cdot \mathbf{x}(t_k) - \frac{4}{3}\mathbf{A}h \cdot \mathbf{x}(t_{k-1}) + \frac{5}{12}\mathbf{A}h \cdot \mathbf{x}(t_{k-2})$$

El Dominio de la Estabilidad

Queremos encontrar el *dominio de la estabilidad del algoritmo AB3*:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(23\mathbf{f}_k - 16\mathbf{f}_{k-1} + 5\mathbf{f}_{k-2}\right)$$

Aplicamos ese algoritmo a nuestro sistema lineal:

$$\mathbf{x}(t_{k+1}) = \left[\mathbf{I}^{(\mathbf{n})} + \frac{23}{12}\mathbf{A}h\right] \cdot \mathbf{x}(t_k) - \frac{4}{3}\mathbf{A}h \cdot \mathbf{x}(t_{k-1}) + \frac{5}{12}\mathbf{A}h \cdot \mathbf{x}(t_{k-2})$$

Sustituimos:

$$\begin{aligned}\mathbf{z_1}(t_k) &= \mathbf{x}(t_{k-2})\\ \mathbf{z_2}(t_k) &= \mathbf{x}(t_{k-1})\\ \mathbf{z_3}(t_k) &= \mathbf{x}(t_k)\end{aligned}$$

Entonces:

$$\begin{aligned}\mathbf{z_1}(t_{k+1}) &= \mathbf{z_2}(t_k)\\ \mathbf{z_2}(t_{k+1}) &= \mathbf{z_3}(t_k)\\ \mathbf{z_3}(t_{k+1}) &= \frac{5}{12}\mathbf{A}h \cdot \mathbf{z_1}(t_k) - \frac{4}{3}\mathbf{A}h \cdot \mathbf{z_2}(t_k) + \left[\mathbf{I}^{(\mathbf{n})} + \frac{23}{12}\mathbf{A}h\right] \cdot \mathbf{z_3}(t_k)\end{aligned}$$

El Dominio de la Estabilidad II

Por eso podemos escribir:

$$\mathbf{z}(t_{k+1}) = \begin{pmatrix} \mathbf{O^{(n)}} & \mathbf{I^{(n)}} & \mathbf{O^{(n)}} \\ \mathbf{O^{(n)}} & \mathbf{O^{(n)}} & \mathbf{I^{(n)}} \\ \frac{5}{12}\mathbf{A}h & -\frac{4}{3}\mathbf{A}h & (\mathbf{I^{(n)}} + \frac{23}{12}\mathbf{A}h) \end{pmatrix} \cdot \mathbf{z}(t_k)$$

El Dominio de la Estabilidad II

Por eso podemos escribir:

$$\mathbf{z}(t_{k+1}) = \begin{pmatrix} \mathbf{O^{(n)}} & \mathbf{I^{(n)}} & \mathbf{O^{(n)}} \\ \mathbf{O^{(n)}} & \mathbf{O^{(n)}} & \mathbf{I^{(n)}} \\ \frac{5}{12}\mathbf{A}h & -\frac{4}{3}\mathbf{A}h & (\mathbf{I^{(n)}} + \frac{23}{12}\mathbf{A}h) \end{pmatrix} \cdot \mathbf{z}(t_k)$$

Es decir:

$$\mathbf{z}(t_{k+1}) = \mathbf{F} \cdot \mathbf{z}(t_k)$$

con:

$$\mathbf{F} = \begin{pmatrix} \mathbf{O^{(n)}} & \mathbf{I^{(n)}} & \mathbf{O^{(n)}} \\ \mathbf{O^{(n)}} & \mathbf{O^{(n)}} & \mathbf{I^{(n)}} \\ \frac{5}{12}\mathbf{A}h & -\frac{4}{3}\mathbf{A}h & (\mathbf{I^{(n)}} + \frac{23}{12}\mathbf{A}h) \end{pmatrix}$$

El Dominio de la Estabilidad II

Por eso podemos escribir:

$$\mathbf{z}(t_{k+1}) = \begin{pmatrix} \mathbf{O^{(n)}} & \mathbf{I^{(n)}} & \mathbf{O^{(n)}} \\ \mathbf{O^{(n)}} & \mathbf{O^{(n)}} & \mathbf{I^{(n)}} \\ \frac{5}{12}\mathbf{A}h & -\frac{4}{3}\mathbf{A}h & (\mathbf{I^{(n)}} + \frac{23}{12}\mathbf{A}h) \end{pmatrix} \cdot \mathbf{z}(t_k)$$

Es decir:

$$\mathbf{z}(t_{k+1}) = \mathbf{F} \cdot \mathbf{z}(t_k)$$

con:

$$\mathbf{F} = \begin{pmatrix} \mathbf{O^{(n)}} & \mathbf{I^{(n)}} & \mathbf{O^{(n)}} \\ \mathbf{O^{(n)}} & \mathbf{O^{(n)}} & \mathbf{I^{(n)}} \\ \frac{5}{12}\mathbf{A}h & -\frac{4}{3}\mathbf{A}h & (\mathbf{I^{(n)}} + \frac{23}{12}\mathbf{A}h) \end{pmatrix}$$

La matriz $\mathbf{F}$ es tres veces más grande que la matriz $\mathbf{A}$. Entonces tiene tres veces más autovalores.

Dominios de la Estabilidad de los Algoritmos AB

Ahora estamos listo para dibujar los *dominios de la estabilidad de los algoritmos AB*.

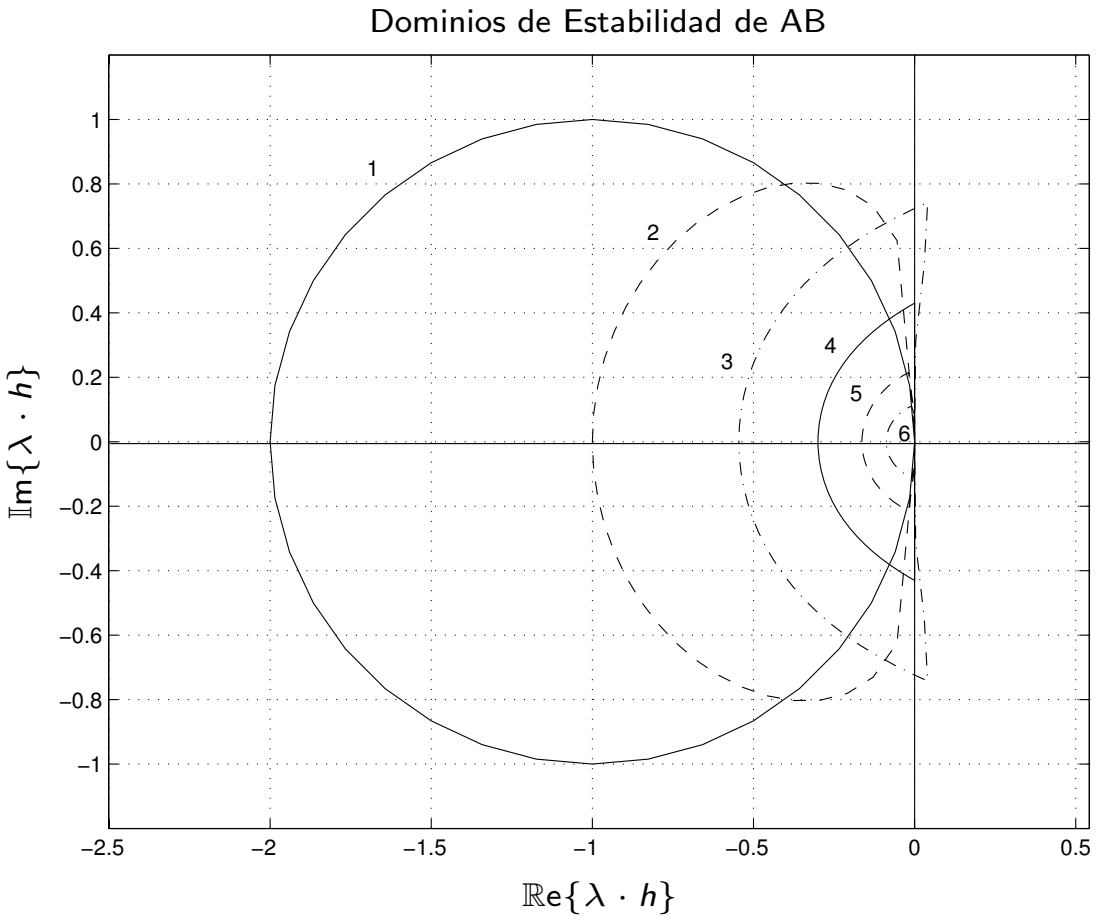


Figure: Dominios de la estabilidad de los algoritmos AB

Dominios de la Estabilidad de los Algoritmos AB II

Dominios de la Estabilidad de los Algoritmos AB II

- Aunque los dominios de la estabilidad de los algoritmos AB de órdenes elevados aproximan mejor el eje imaginario en la proximidad del origen, el tamaño de los dominios de la estabilidad numérica disminuye en lugar de crecer para los algoritmos de órdenes más altos.

Dominios de la Estabilidad de los Algoritmos AB II

- Aunque los dominios de la estabilidad de los algoritmos AB de órdenes elevados aproximan mejor el eje imaginario en la proximidad del origen, el tamaño de los dominios de la estabilidad numérica disminuye en lugar de crecer para los algoritmos de órdenes más altos.

- No existen algoritmos AB estables para órdenes más grandes que seis.

Dominios de la Estabilidad de los Algoritmos AB II

- Aunque los dominios de la estabilidad de los algoritmos AB de órdenes elevados aproximan mejor el eje imaginario en la proximidad del origen, el tamaño de los dominios de la estabilidad numérica disminuye en lugar de crecer para los algoritmos de órdenes más altos.
- No existen algoritmos AB estables para órdenes más grandes que seis.
- Queremos usar algoritmos de órdenes elevados para mejorar la precisión de las simulaciones mientras que usamos pasos de integración más grandes. En realidad tenemos que reducir los pasos a causa de problemas con la estabilidad numérica.

Dominios de la Estabilidad de los Algoritmos AB II

- Aunque los dominios de la estabilidad de los algoritmos AB de órdenes elevados aproximan mejor el eje imaginario en la proximidad del origen, el tamaño de los dominios de la estabilidad numérica disminuye en lugar de crecer para los algoritmos de órdenes más altos.
- No existen algoritmos AB estables para órdenes más grandes que seis.
- Queremos usar algoritmos de órdenes elevados para mejorar la precisión de las simulaciones mientras que usamos pasos de integración más grandes. En realidad tenemos que reducir los pasos a causa de problemas con la estabilidad numérica.
- Aunque la carga computacional asociada con un solo paso es mucho más baja en el caso de los algoritmos AB en comparación con la de los algoritmos RK, tenemos que usar pasos mucho más pequeños por razones de la estabilidad numérica limitada de estos algoritmos. Por eso no es cierto que los algoritmos AB sean más económicos que los algoritmos RK.

Dominios de la Estabilidad de los Algoritmos AB III

¿Qué pasó?

Dominios de la Estabilidad de los Algoritmos AB III

¿Qué pasó?

Estamos buscando *polinomios de interpolación* de órdenes grandes pasando por un número de puntos extendidos de forma equidistante. Después usamos esos polinomios para una *extrapolación*.

Dominios de la Estabilidad de los Algoritmos AB III

¿Qué pasó?

Estamos buscando *polinomios de interpolación* de órdenes grandes pasando por un número de puntos extendidos de forma equidistante. Después usamos esos polinomios para una *extrapolación*.

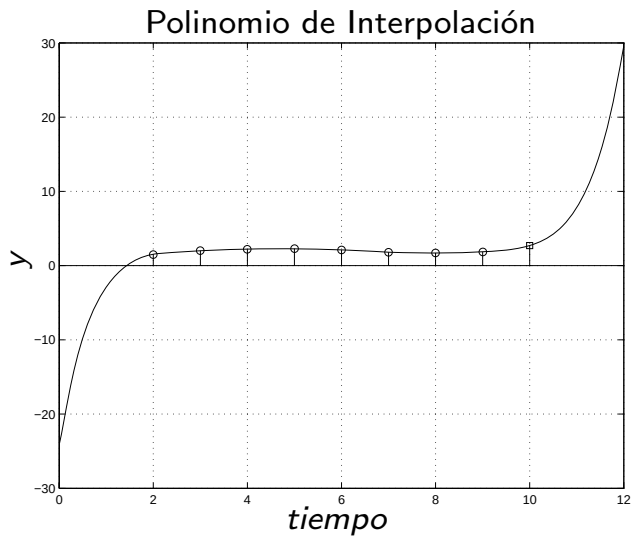


Figure: Polinomio de interpolación usado para extrapolación

Dominios de la Estabilidad de los Algoritmos AB III

¿Qué pasó?

Estamos buscando *polinomios de interpolación* de órdenes grandes pasando por un número de puntos extendidos de forma equidistante. Después usamos esos polinomios para una *extrapolación*.

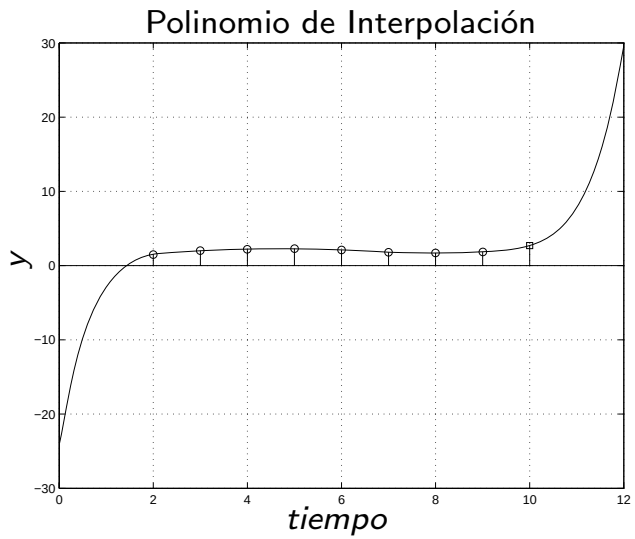


Figure: Polinomio de interpolación usado para extrapolación

Eso no funciona muy bien.

Las Fórmulas Implícitas de Adams-Moulton

Si usamos *métodos implícitos* en lugar de explícitos, podemos *interpolar* en lugar de extrapolar. Esto podra servirnos.

Las Fórmulas Implícitas de Adams-Moulton

Si usamos *métodos implícitos* en lugar de explícitos, podemos *interpolar* en lugar de extrapolar. Esto podra servirnos.

Formulamos ahora un *polinomio de Newton-Gregory hacia atrás* de la derivada $\dot{\mathbf{x}}$ alrededor el instante en el tiempo t_{k+1}:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Las Fórmulas Implícitas de Adams-Moulton

Si usamos *métodos implícitos* en lugar de explícitos, podemos *interpolar* en lugar de extrapolar. Esto podra servirnos.

Formulamos ahora un *polinomio de Newton-Gregory hacia atrás* de la derivada $\dot{\mathbf{x}}$ alrededor el instante en el tiempo t_{k+1}:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Integramos ese polinomio en el intervalo $t \in [t_k, t_{k+1}]$. Esta vez corresponde al intervalo $s \in [-1.0, 0.0]$.

Las Fórmulas Implícitas de Adams-Moulton

Si usamos *métodos implícitos* en lugar de explícitos, podemos *interpolar* en lugar de extrapolar. Esto podra servirnos.

Formulamos ahora un *polinomio de Newton-Gregory hacia atrás* de la derivada $\dot{\mathbf{x}}$ alrededor el instante en el tiempo t_{k+1}:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Integramos ese polinomio en el intervalo $t \in [t_k, t_{k+1}]$. Esta vez corresponde al intervalo $s \in [-1.0, 0.0]$.

Resulta la familia de fórmulas:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + h\left(\mathbf{f}_{k+1} - \frac{1}{2}\nabla\mathbf{f}_{k+1} - \frac{1}{12}\nabla^2\mathbf{f}_{k+1} - \frac{1}{24}\nabla^3\mathbf{f}_{k+1} + \dots\right)$$

Las Fórmulas Implícitas de Adams-Moulton II

Truncando después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(5\mathbf{f}_{k+1} + 8\mathbf{f}_k - \mathbf{f}_{k-1}\right)$$

el bien conocido *algoritmo de Adams-Moulton de tercer orden (AM3)*.

Las Fórmulas Implícitas de Adams-Moulton II

Truncando después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(5\mathbf{f}_{k+1} + 8\mathbf{f}_k - \mathbf{f}_{k-1}\right)$$

el bien conocido *algoritmo de Adams-Moulton de tercer orden (AM3)*.

Si truncamos después del término cúbico obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{24}\left(9\mathbf{f}_{k+1} + 19\mathbf{f}_k - 5\mathbf{f}_{k-1} + \mathbf{f}_{k-2}\right)$$

es decir el *algoritmo de Adams-Moulton de cuarto orden (AM4)*.

Las Fórmulas Implícitas de Adams-Moulton II

Truncando después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(5\mathbf{f}_{k+1} + 8\mathbf{f}_k - \mathbf{f}_{k-1}\right)$$

el bien conocido *algoritmo de Adams-Moulton de tercer orden (AM3)*.

Si truncamos después del término cúbico obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{24}\left(9\mathbf{f}_{k+1} + 19\mathbf{f}_k - 5\mathbf{f}_{k-1} + \mathbf{f}_{k-2}\right)$$

es decir el *algoritmo de Adams-Moulton de cuarto orden (AM4)*.

El *algoritmo AM1* ya se conoce bajo el nombre *algoritmo BE*.

Las Fórmulas Implícitas de Adams-Moulton II

Truncando después del término cuadrado obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{12}\left(5\mathbf{f}_{k+1} + 8\mathbf{f}_k - \mathbf{f}_{k-1}\right)$$

el bien conocido *algoritmo de Adams-Moulton de tercer orden (AM3)*.

Si truncamos después del término cúbico obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_k) + \frac{h}{24}\left(9\mathbf{f}_{k+1} + 19\mathbf{f}_k - 5\mathbf{f}_{k-1} + \mathbf{f}_{k-2}\right)$$

es decir el *algoritmo de Adams-Moulton de cuarto orden (AM4)*.

El *algoritmo AM1* ya se conoce bajo el nombre *algoritmo BE*.

Todos los algoritmos AM pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 2 & 12 & 24 & 720 & 1440 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 5 & 8 & -1 & 0 & 0 & 0 \\ 9 & 19 & -5 & 1 & 0 & 0 \\ 251 & 646 & -264 & 106 & -19 & 0 \\ 475 & 1427 & -798 & 482 & -173 & 27 \end{pmatrix}$$

Dominios de la Estabilidad de los Algoritmos AM

Podemos dibujar los *dominios de la estabilidad de los algoritmos AM*.

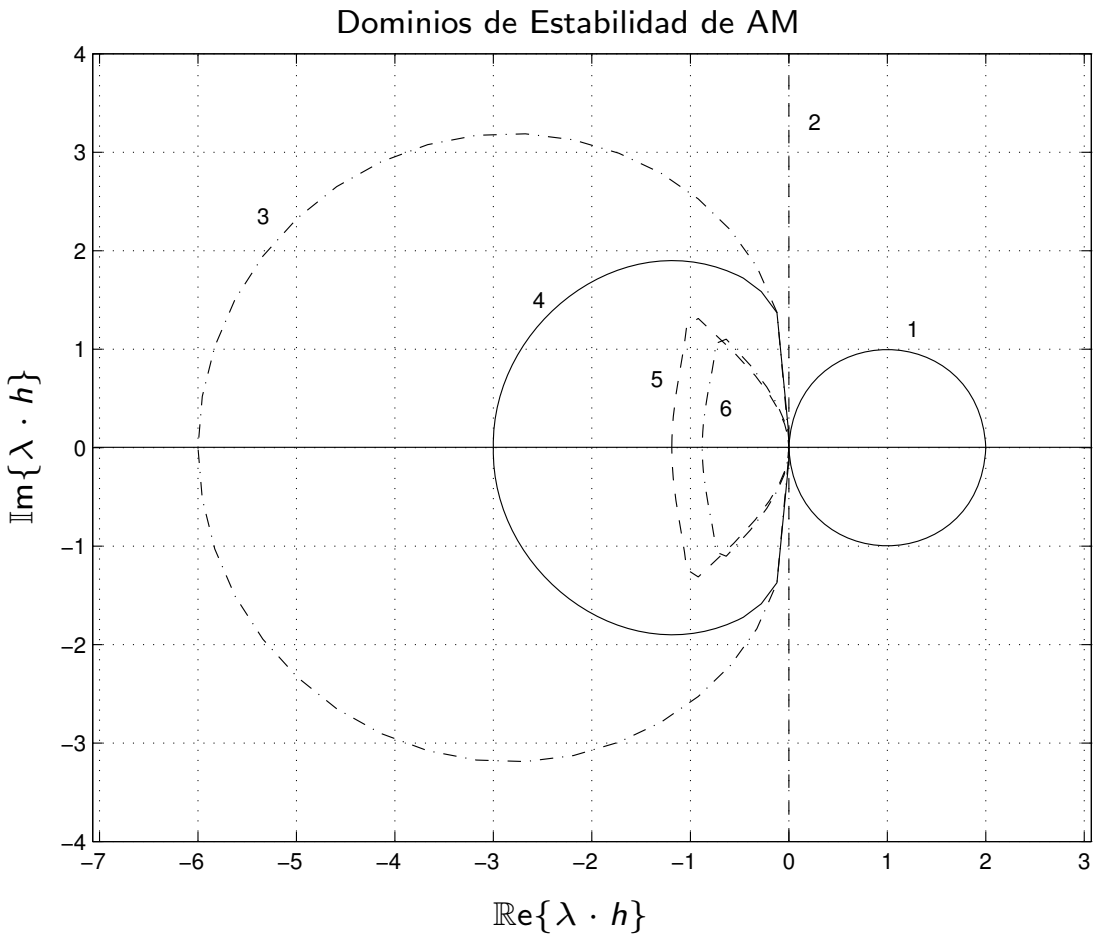


Figure: Dominios de la estabilidad de los algoritmos AM

Dominios de la Estabilidad de los Algoritmos AM II

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.
- Los dominios de la estabilidad numérica de los demás algoritmos AM se cierran en la parte izquierda del plano complejo. Esos algoritmos, aunque implícitos, no sirven entonces para la simulación de sistemas rígidos.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.
- Los dominios de la estabilidad numérica de los demás algoritmos AM se cierran en la parte izquierda del plano complejo. Esos algoritmos, aunque implícitos, no sirven entonces para la simulación de sistemas rígidos.
- Tenemos el mismo problema que antes. Los dominios de los algoritmos AM de órdenes elevados, aunque más grandes que los de los algoritmos AB, disminuyen en lugar de crecer.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.
- Los dominios de la estabilidad numérica de los demás algoritmos AM se cierran en la parte izquierda del plano complejo. Esos algoritmos, aunque implícitos, no sirven entonces para la simulación de sistemas rígidos.
- Tenemos el mismo problema que antes. Los dominios de los algoritmos AM de órdenes elevados, aunque más grandes que los de los algoritmos AB, disminuyen en lugar de crecer.
- No existen algoritmos AM estables para órdenes más grandes que seis.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.
- Los dominios de la estabilidad numérica de los demás algoritmos AM se cierran en la parte izquierda del plano complejo. Esos algoritmos, aunque implícitos, no sirven entonces para la simulación de sistemas rígidos.
- Tenemos el mismo problema que antes. Los dominios de los algoritmos AM de órdenes elevados, aunque más grandes que los de los algoritmos AB, disminuyen en lugar de crecer.
- No existen algoritmos AM estables para órdenes más grandes que seis.
- En promedio se necesitan tres iteraciones de Newton durante cada paso de integración. Entonces se necesitan tres evaluaciones del modelo para cada paso.

Dominios de la Estabilidad de los Algoritmos AM II

- El algoritmo AM1=BE=BRK1 es L-estable.
- El algoritmo AM2=TR=BRK2 es F-estable.
- Los dominios de la estabilidad numérica de los demás algoritmos AM se cierran en la parte izquierda del plano complejo. Esos algoritmos, aunque implícitos, no sirven entonces para la simulación de sistemas rígidos.
- Tenemos el mismo problema que antes. Los dominios de los algoritmos AM de órdenes elevados, aunque más grandes que los de los algoritmos AB, disminuyen en lugar de crecer.
- No existen algoritmos AM estables para órdenes más grandes que seis.
- En promedio se necesitan tres iteraciones de Newton durante cada paso de integración. Entonces se necesitan tres evaluaciones del modelo para cada paso.
- A causa de los dominios de estabilidad más grandes de los algoritmos AM, pueden usarse en promedio pasos aproximadamente tres veces más grandes. La eficiencia de los algoritmos AB y AM es entonces aproximadamente la misma.

Las Fórmulas Explícitas de Adams-Bashforth-Moulton

A veces se usa un método combinado consistente de una predicción AB seguido por una corrección AM, por ejemplo:

predicción:
$$\dot{\mathbf{x}}_{\mathbf{k}} = \mathbf{f}(\mathbf{x}_{\mathbf{k}}, t_k)$$
$$\mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \frac{h}{12}(23\dot{\mathbf{x}}_{\mathbf{k}} - 16\dot{\mathbf{x}}_{\mathbf{k-1}} + 5\dot{\mathbf{x}}_{\mathbf{k-2}})$$

corrección:
$$\dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{f}(\mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}}, t_{k+1})$$
$$\mathbf{x}^{\mathbf{C}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \frac{h}{12}(5\dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} + 8\dot{\mathbf{x}}_{\mathbf{k}} - \dot{\mathbf{x}}_{\mathbf{k-1}})$$

Las Fórmulas Explícitas de Adams-Bashforth-Moulton

A veces se usa un método combinado consistente de una predicción AB seguido por una corrección AM, por ejemplo:

$$\begin{aligned}
&\text{predicción:} && \dot{\mathbf{x}}_{\mathbf{k}} = \mathbf{f}(\mathbf{x}_{\mathbf{k}}, t_k) \\
& && \mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \tfrac{h}{12}(23\dot{\mathbf{x}}_{\mathbf{k}} - 16\dot{\mathbf{x}}_{\mathbf{k-1}} + 5\dot{\mathbf{x}}_{\mathbf{k-2}}) \\
&\text{corrección:} && \dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{f}(\mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}}, t_{k+1}) \\
& && \mathbf{x}^{\mathbf{C}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \tfrac{h}{12}(5\dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} + 8\dot{\mathbf{x}}_{\mathbf{k}} - \dot{\mathbf{x}}_{\mathbf{k-1}})
\end{aligned}$$

Estos métodos se llaman *algoritmos predictor corrector de Adams-Bashforth-Moulton (ABM)*.

Las Fórmulas Explícitas de Adams-Bashforth-Moulton

A veces se usa un método combinado consistente de una predicción AB seguido por una corrección AM, por ejemplo:

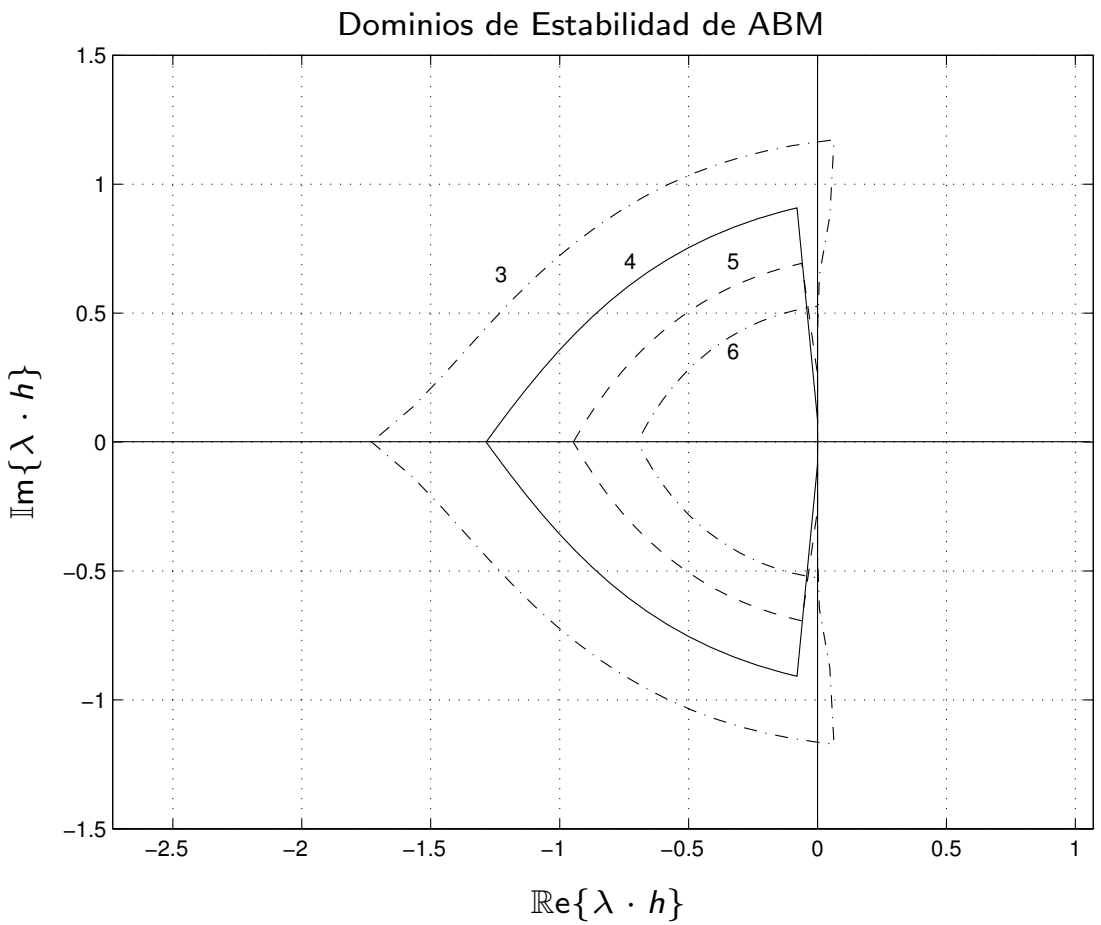
$$\begin{aligned} &\text{predicción:} && \dot{\mathbf{x}}_{\mathbf{k}} = \mathbf{f}(\mathbf{x}_{\mathbf{k}}, t_k) \\ & && \mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \tfrac{h}{12}(23\dot{\mathbf{x}}_{\mathbf{k}} - 16\dot{\mathbf{x}}_{\mathbf{k-1}} + 5\dot{\mathbf{x}}_{\mathbf{k-2}}) \\ &\text{corrección:} && \dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} = \mathbf{f}(\mathbf{x}^{\mathbf{P}}_{\mathbf{k+1}}, t_{k+1}) \\ & && \mathbf{x}^{\mathbf{C}}_{\mathbf{k+1}} = \mathbf{x}_{\mathbf{k}} + \tfrac{h}{12}(5\dot{\mathbf{x}}^{\mathbf{P}}_{\mathbf{k+1}} + 8\dot{\mathbf{x}}_{\mathbf{k}} - \dot{\mathbf{x}}_{\mathbf{k-1}}) \end{aligned}$$

Estos métodos se llaman *algoritmos predictor corrector de Adams-Bashforth-Moulton (ABM)*.

El método combinado es un *algoritmo explícito*.

Dominios de la Estabilidad de los Algoritmos ABM

Podemos dibujar los *dominios de la estabilidad de los algoritmos predictor corrector ABM*.



Dominios de la Estabilidad de los Algoritmos ABM II

Dominios de la Estabilidad de los Algoritmos ABM II

- Se necesitan dos evaluaciones del modelo para cada paso, una para la predicción y otra para la corrección.

Dominios de la Estabilidad de los Algoritmos ABM II

- Se necesitan dos evaluaciones del modelo para cada paso, una para la predicción y otra para la corrección.
- A causa de los dominios de estabilidad más grandes de los algoritmos ABM en comparación con los algoritmos AB, se pueden usar en promedio pasos aproximadamente dos veces más grandes. La eficiencia de los algoritmos AB y ABM es entonces aproximadamente la misma.

Las Fórmulas Implícitas de Diferencias hacia Atrás

Hasta ahora no encontramos ninguna familia de fórmulas que puedan usarse en la simulación de sistemas rígidos. Introducimos una de tales familias ahora.

Las Fórmulas Implícitas de Diferencias hacia Atrás

Hasta ahora no encontramos ninguna familia de fórmulas que puedan usarse en la simulación de sistemas rígidos. Introducimos una de tales familias ahora.

Formulamos un *polinomio de Newton-Gregory hacia atrás* de las variables del estado $\mathbf{x}$ alrededor el instante en el tiempo t_{k+1}:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + \binom{s}{1}\nabla\mathbf{x}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{x}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{x}_{k+1} + \dots$$

o:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + s\nabla\mathbf{x}_{k+1} + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{x}_{k+1} + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{x}_{k+1} + \dots$$

Las Fórmulas Implícitas de Diferencias hacia Atrás

Hasta ahora no encontramos ninguna familia de fórmulas que puedan usarse en la simulación de sistemas rígidos. Introducimos una de tales familias ahora.

Formulamos un *polinomio de Newton-Gregory hacia atrás* de las variables del estado $\mathbf{x}$ alrededor el instante en el tiempo t_{k+1}:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + \binom{s}{1}\nabla\mathbf{x}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{x}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{x}_{k+1} + \dots$$

o:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + s\nabla\mathbf{x}_{k+1} + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{x}_{k+1} + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{x}_{k+1} + \dots$$

Diferenciamos ese polinomio con respecto a la variable t:

$$\dot{\mathbf{x}}(t) = \frac{1}{h}\left[\nabla\mathbf{x}_{k+1} + \left(s + \frac{1}{2}\right)\nabla^2\mathbf{x}_{k+1} + \left(\frac{s^2}{2} + s + \frac{1}{3}\right)\nabla^3\mathbf{x}_{k+1} + \dots\right]$$

Las Fórmulas Implícitas de Diferencias hacia Atrás

Hasta ahora no encontramos ninguna familia de fórmulas que puedan usarse en la simulación de sistemas rígidos. Introducimos una de tales familias ahora.

Formulamos un *polinomio de Newton-Gregory hacia atrás* de las variables del estado $\mathbf{x}$ alrededor el instante en el tiempo t_{k+1}:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + \binom{s}{1}\nabla\mathbf{x}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{x}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{x}_{k+1} + \dots$$

o:

$$\mathbf{x}(t) = \mathbf{x}_{k+1} + s\nabla\mathbf{x}_{k+1} + \left(\frac{s^2}{2} + \frac{s}{2}\right)\nabla^2\mathbf{x}_{k+1} + \left(\frac{s^3}{6} + \frac{s^2}{2} + \frac{s}{3}\right)\nabla^3\mathbf{x}_{k+1} + \dots$$

Diferenciamos ese polinomio con respecto a la variable t:

$$\dot{\mathbf{x}}(t) = \frac{1}{h}\left[\nabla\mathbf{x}_{k+1} + \left(s + \frac{1}{2}\right)\nabla^2\mathbf{x}_{k+1} + \left(\frac{s^2}{2} + s + \frac{1}{3}\right)\nabla^3\mathbf{x}_{k+1} + \dots\right]$$

Evaluamos por $s = 0.0$:

$$\dot{\mathbf{x}}(t_{k+1}) = \frac{1}{h}\left[\nabla\mathbf{x}_{k+1} + \frac{1}{2}\nabla^2\mathbf{x}_{k+1} + \frac{1}{3}\nabla^3\mathbf{x}_{k+1} + \dots\right]$$

Las Fórmulas Implícitas de Diferencias hacia Atrás II

Truncando después del término cúbico obtenemos:

$$h \cdot \mathbf{f}_{k+1} = \frac{11}{6}\mathbf{x}_{k+1} - 3\mathbf{x}_k + \frac{3}{2}\mathbf{x}_{k-1} - \frac{1}{3}\mathbf{x}_{k-2}$$

Las Fórmulas Implícitas de Diferencias hacia Atrás II

Truncando después del término cúbico obtenemos:

$$h \cdot \mathbf{f}_{k+1} = \frac{11}{6}\mathbf{x}_{k+1} - 3\mathbf{x}_k + \frac{3}{2}\mathbf{x}_{k-1} - \frac{1}{3}\mathbf{x}_{k-2}$$

Podemos resolver la ecuación por la variable del estado en el instante t_{k+1}:

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

Las Fórmulas Implícitas de Diferencias hacia Atrás II

Truncando después del término cúbico obtenemos:

$$h \cdot \mathbf{f}_{k+1} = \frac{11}{6}\mathbf{x}_{k+1} - 3\mathbf{x}_k + \frac{3}{2}\mathbf{x}_{k-1} - \frac{1}{3}\mathbf{x}_{k-2}$$

Podemos resolver la ecuación por la variable del estado en el instante t_{k+1}:

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

Resulta la bien conocida *fórmula de diferencias hacia atrás de tercer orden (BDF3)*.

Las Fórmulas Implícitas de Diferencias hacia Atrás II

Truncando después del término cúbico obtenemos:

$$h \cdot \mathbf{f}_{k+1} = \frac{11}{6}\mathbf{x}_{k+1} - 3\mathbf{x}_k + \frac{3}{2}\mathbf{x}_{k-1} - \frac{1}{3}\mathbf{x}_{k-2}$$

Podemos resolver la ecuación por la variable del estado en el instante t_{k+1}:

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

Resulta la bien conocida *fórmula de diferencias hacia atrás de tercer orden (BDF3)*.

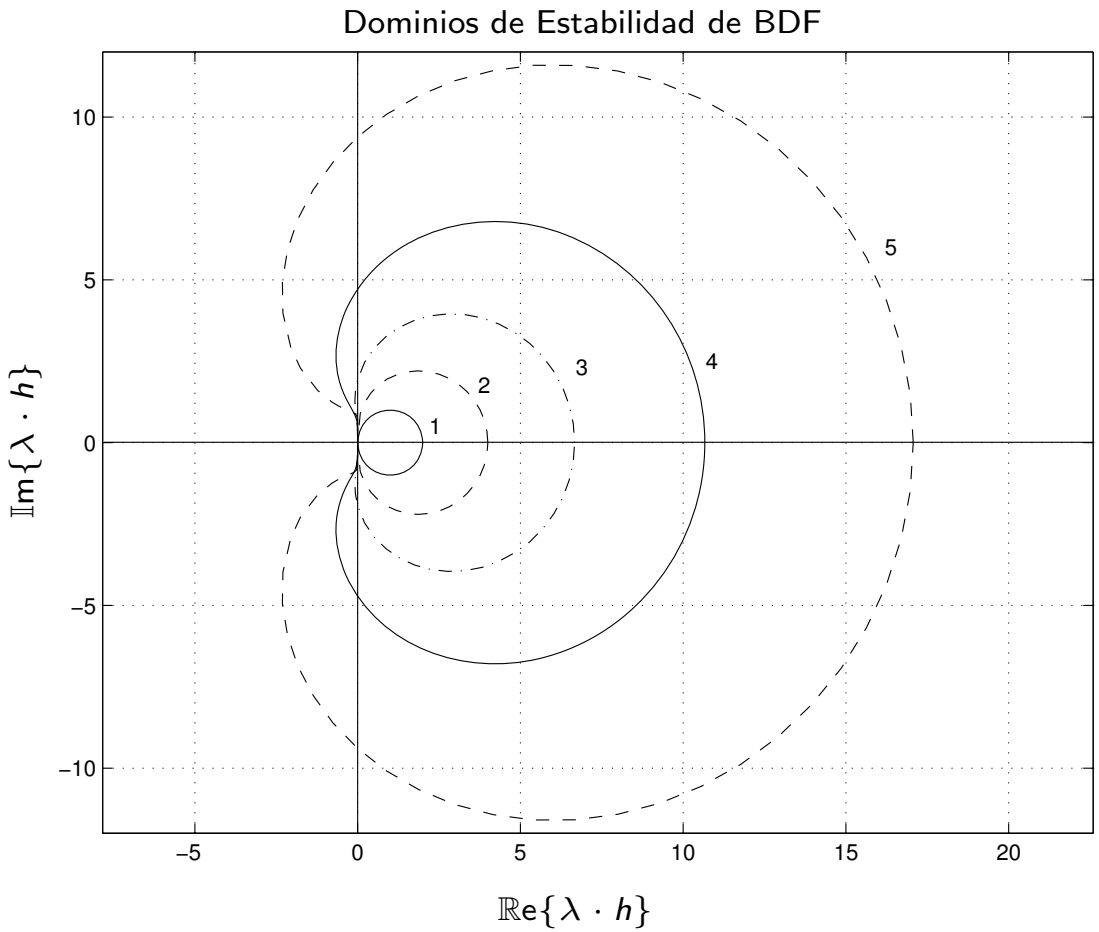
Todos los algoritmos BDF pueden caracterizarse por un vector α especificando el factor asociado con la derivada y por una matriz β que contiene los pesos de las variables del estado en el pasado:

$$\alpha = \begin{pmatrix} 1 & 2/3 & 6/11 & 12/25 & 60/137 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 4/3 & -1/3 & 0 & 0 & 0 \\ 18/11 & -9/11 & 2/11 & 0 & 0 \\ 48/25 & -36/25 & 16/25 & -3/25 & 0 \\ 300/137 & -300/137 & 200/137 & -75/137 & 12/137 \end{pmatrix}$$

Dominios de la Estabilidad de los Algoritmos BDF

Podemos dibujar los *dominios de la estabilidad de los algoritmos de diferencias hacia atrás BDF*.



Dominios de la Estabilidad de los Algoritmos BDF II

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.
- Los demás algoritmos BDF no son L-estables. Ni siquiera son A-estables. Una parte del plano complejo a la izquierda del eje imaginario permanece inestable.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.
- Los demás algoritmos BDF no son L-estables. Ni siquiera son A-estables. Una parte del plano complejo a la izquierda del eje imaginario permanece inestable.
- No existen algoritmos BDF estables para órdenes más grandes que seis.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.
- Los demás algoritmos BDF no son L-estables. Ni siquiera son A-estables. Una parte del plano complejo a la izquierda del eje imaginario permanece inestable.
- No existen algoritmos BDF estables para órdenes más grandes que seis.
- Ya el algoritmo BDF6 no debe usarse salvo para la simulación de sistemas rígidos sin oscilaciones, por ejemplo sistemas térmicos o químicos, porque la parte inestable a la izquierda del eje imaginario del plano complejo es demasiado grande.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.
- Los demás algoritmos BDF no son L-estables. Ni siquiera son A-estables. Una parte del plano complejo a la izquierda del eje imaginario permanece inestable.
- No existen algoritmos BDF estables para órdenes más grandes que seis.
- Ya el algoritmo BDF6 no debe usarse salvo para la simulación de sistemas rígidos sin oscilaciones, por ejemplo sistemas térmicos o químicos, porque la parte inestable a la izquierda del eje imaginario del plano complejo es demasiado grande.
- Los algoritmos BDF son implícitos. También pueden derivarse algoritmos BDF explícitos, pero desafortunadamente son inestables en el entero plano complejo.

Dominios de la Estabilidad de los Algoritmos BDF II

- El algoritmo BDF1=BE=BRK1=AM1 es L-estable.
- El algoritmo BDF2 también es L-estable.
- Los demás algoritmos BDF no son L-estables. Ni siquiera son A-estables. Una parte del plano complejo a la izquierda del eje imaginario permanece inestable.
- No existen algoritmos BDF estables para órdenes más grandes que seis.
- Ya el algoritmo BDF6 no debe usarse salvo para la simulación de sistemas rígidos sin oscilaciones, por ejemplo sistemas térmicos o químicos, porque la parte inestable a la izquierda del eje imaginario del plano complejo es demasiado grande.
- Los algoritmos BDF son implícitos. También pueden derivarse algoritmos BDF explícitos, pero desafortunadamente son inestables en el entero plano complejo.
- **Gracias a sus calidades en la simulación de sistemas rígidos y a causa de su simplicidad, los algoritmos BDF están entre los más usados en la simulación de sistemas dinámicos.**

Las Fórmulas Explícitas de Nyström

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AB:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots$$

Las Fórmulas Explícitas de Nyström

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AB:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots$$

Esta vez integramos ese polinomio en el intervalo $t \in [t_{k-1}, t_{k+1}]$, es decir en el intervalo $s \in [-1.0, +1.0]$. Encontramos la familia de fórmulas:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + h\left(2\mathbf{f}_k + \frac{1}{3}\nabla^2\mathbf{f}_k + \frac{1}{3}\nabla^3\mathbf{f}_k + \dots\right)$$

Las Fórmulas Explícitas de Nyström

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AB:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k + \binom{s}{1}\nabla\mathbf{f}_k + \binom{s+1}{2}\nabla^2\mathbf{f}_k + \binom{s+2}{3}\nabla^3\mathbf{f}_k + \dots$$

Esta vez integramos ese polinomio en el intervalo $t \in [t_{k-1}, t_{k+1}]$, es decir en el intervalo $s \in [-1.0, +1.0]$. Encontramos la familia de fórmulas:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + h\left(2\mathbf{f}_k + \frac{1}{3}\nabla^2\mathbf{f}_k + \frac{1}{3}\nabla^3\mathbf{f}_k + \dots\right)$$

Truncando después del término cúbico obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + \frac{h}{3}\left(8\mathbf{f}_k - 5\mathbf{f}_{k-1} + 4\mathbf{f}_{k-2} - \mathbf{f}_{k-3}\right)$$

Ese algoritmo se llama *algoritmo de Nyström de cuarto orden (Ny4)*.

Las Fórmulas Explícitas de Nyström II

Los algoritmos de Nyström pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 7 & -2 & 1 & 0 & 0 \\ 8 & -5 & 4 & -1 & 0 \\ 269 & -266 & 294 & -146 & 29 \end{pmatrix}$$

Las Fórmulas Explícitas de Nyström II

Los algoritmos de Nyström pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 7 & -2 & 1 & 0 & 0 \\ 8 & -5 & 4 & -1 & 0 \\ 269 & -266 & 294 & -146 & 29 \end{pmatrix}$$

Desafortunadamente todos los algoritmos de la familia Nyström son inestables.

Las Fórmulas Explícitas de Nyström II

Los algoritmos de Nyström pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 \\ 7 & -2 & 1 & 0 & 0 \\ 8 & -5 & 4 & -1 & 0 \\ 269 & -266 & 294 & -146 & 29 \end{pmatrix}$$

Desafortunadamente todos los algoritmos de la familia Nyström son inestables.

Sin embargo pueden usarse a veces como *componentes de métodos combinados o cíclicos*.

Las Fórmulas Implícitas de Milne

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AM:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Las Fórmulas Implícitas de Milne

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AM:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Integramos ese polinomio en el intervalo $t \in [t_{k-1}, t_{k+1}]$, es decir en el intervalo $s \in [-2.0, 0.0]$. Encontramos la familia de fórmulas:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + h\left(2\mathbf{f}_{k+1} - 2\nabla\mathbf{f}_{k+1} + \frac{1}{3}\nabla^2\mathbf{f}_{k+1} + 0\nabla^3\mathbf{f}_{k+1} + \dots\right)$$

Las Fórmulas Implícitas de Milne

Empezamos con el mismo polinomio de Newton-Gregory hacia atrás que ya usamos en la derivación de los algoritmos AM:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_{k+1} + \binom{s}{1}\nabla\mathbf{f}_{k+1} + \binom{s+1}{2}\nabla^2\mathbf{f}_{k+1} + \binom{s+2}{3}\nabla^3\mathbf{f}_{k+1} + \dots$$

Integramos ese polinomio en el intervalo $t \in [t_{k-1}, t_{k+1}]$, es decir en el intervalo $s \in [-2.0, 0.0]$. Encontramos la familia de fórmulas:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + h\left(2\mathbf{f}_{k+1} - 2\nabla\mathbf{f}_{k+1} + \frac{1}{3}\nabla^2\mathbf{f}_{k+1} + 0\nabla^3\mathbf{f}_{k+1} + \dots\right)$$

Truncando después del término cúbico (cuadrado en realidad) obtenemos:

$$\mathbf{x}(t_{k+1}) = \mathbf{x}(t_{k-1}) + \frac{h}{3}\left(\mathbf{f}_{k+1} + 4\mathbf{f}_k + \mathbf{f}_{k-1}\right)$$

Ese algoritmo se llama *algoritmo de Milne de cuarto orden (Mi4)*.

Las Fórmulas Explícitas de Milne II

Los algoritmos de Milne pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 29 & 124 & 24 & 4 & -1 \end{pmatrix}$$

Las Fórmulas Explícitas de Milne II

Los algoritmos de Milne pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 29 & 124 & 24 & 4 & -1 \end{pmatrix}$$

Desafortunadamente todos los algoritmos de la familia Milne son también inestables.

Las Fórmulas Explícitas de Milne II

Los algoritmos de Milne pueden caracterizarse por un vector α especificando el factor asociado con el paso h y por una matriz β que contiene los pesos de las derivadas:

$$\alpha = \begin{pmatrix} 1 & 1 & 3 & 3 & 90 \end{pmatrix}^T$$

$$\beta = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 \\ 29 & 124 & 24 & 4 & -1 \end{pmatrix}$$

Desafortunadamente todos los algoritmos de la familia Milne son también inestables.

También esos algoritmos se usan a veces como *componentes de métodos combinados o cíclicos*.

Conclusiones

Conclusiones

- En esta presentación introdujimos *métodos lineales en múltiples pasos* para la simulación de sistemas dinámicos.

Conclusiones

- En esta presentación introdujimos *métodos lineales en múltiples pasos* para la simulación de sistemas dinámicos.
- Empezamos con las familias de los *algoritmos de Adams*. Existen entre ellos *algoritmos explícitos* y *algoritmos implícitos*.

Conclusiones

- En esta presentación introdujimos *métodos lineales en múltiples pasos* para la simulación de sistemas dinámicos.
- Empezamos con las familias de los *algoritmos de Adams*. Existen entre ellos *algoritmos explícitos* y *algoritmos implícitos*.
- Analizamos sus *propiedades de la estabilidad numérica*. Enseñamos que estos métodos no son muy útiles. En la práctica de los ingenieros no se usan mucho salvo para la simulación de sistemas lineales no rígidos sin discontinuidades.

Conclusiones

- En esta presentación introdujimos *métodos lineales en múltiples pasos* para la simulación de sistemas dinámicos.
- Empezamos con las familias de los *algoritmos de Adams*. Existen entre ellos *algoritmos explícitos* y *algoritmos implícitos*.
- Analizamos sus *propiedades de la estabilidad numérica*. Enseñamos que estos métodos no son muy útiles. En la práctica de los ingenieros no se usan mucho salvo para la simulación de sistemas lineales no rígidos sin discontinuidades.
- Después introdujimos la familia de los *algoritmos de diferencias hacia atrás*. Estos algoritmos son muy útiles para la simulación de sistemas rígidos. Están entre los algoritmos más utilizados en los entornos de la simulación de sistemas dinámicos.