Simulación de sistemas por cuantificación de estados.

JCC 2010 - FCEIA - UNR

Ernesto Kofman - Federico Bergero

Laboratorio de Sistemas Dinámicos y Procesamiento de la Información
FCEIA - UNR.
CIFASIS - CONICET

Organización de la Presentación

Organización de la Presentación

Sistemas Continuos

Son sistemas cuyas variables evolucionan de forma continua en el tiempo

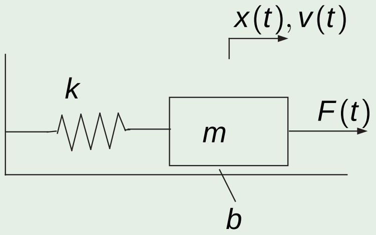
Esto incluye:

- sistemas físicos (mecánicos, eléctricos, hidráulicos, etc.),
- procesos químicos,
- dinámica de poblaciones,
- algunos modelos económicos,
- etc.

Estos sistemas pueden en general modelarse mediante Ecuaciones Diferenciales.

Sistemas Continuos – Ejemplo

Sistema Masa–Resorte



Modelo del sistema según las leyes de Newton.(de segundo orden):

$$\begin{aligned} \dot{x}(t) &= v(t) \\ \dot{v}(t) &= -\frac{k}{m}x(t) - \frac{b}{m}v(t) + \frac{1}{m}F(t) \end{aligned} \tag{1}$$

Sistemas Continuos – Ejemplo (cont)

Si nos interesa predecir el comportamiento del sistema, debemos resolver la Ecuación Diferencial (1).

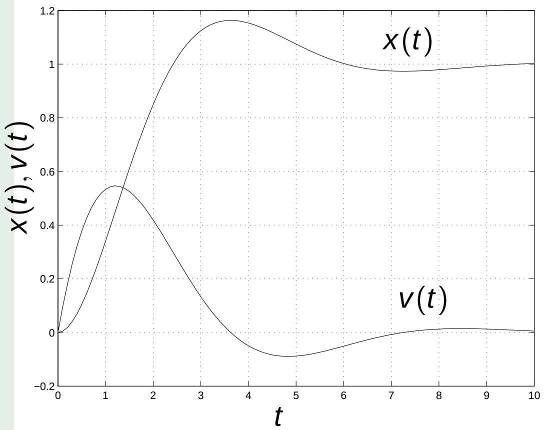
Por ejemplo, para los parámetros $k = b = m = 1$, tomando $F(t) = 1$ para $t \geq 0$ y las condiciones iniciales $x(0) = 0$ y $v(0) = 0$, la solución analítica está dada por

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

para todo $t \geq 0$

Sistemas Continuos – Ejemplo (cont)

Solución de la Ecuación (1)



Sistemas Continuos – Ecuaciones de Estado

En general, los sistemas continuos con parámetros concentrados pueden describirse mediante Ecuaciones Diferenciales Ordinarias.(EDOs)

De aquí en más, escribiremos las EDOs como Ecuaciones de Estado:

$$\begin{aligned} \dot{x}_1(t) &= f_1(x_1(t), \cdots, x_n(t), t) \\ &\vdots \\ \dot{x}_n(t) &= f_n(x_1(t), \cdots, x_n(t), t) \end{aligned} \tag{3}$$

donde $x_1, x_2, \cdots, x_n$ se denominan variables de estado y n es el orden del sistema.

Sistemas Continuos – Solución de las EDOs

La Ecuación de Estados (en forma vectorial)

$$\dot{x}(t) = f(x(t), t) \tag{4}$$

con condición inicial

$$x(t_0) = x_0 \tag{5}$$

en general no puede resolverse de manera analítica (salvo en casos lineales o algunos casos no lineales muy simples).

Por este motivo, para conocer la evolución de las variables del sistema $x_i(t)$ debe recurrirse a la integración numérica.

Métodos de Integración Numérica

Consideremos el sistema

$$\dot{x}(t) = f(x(t), t) \tag{6}$$

con la condición inicial $x(t_0) = x_0$ conocida.

El objetivo de los métodos de integración numérica es obtener una solución aproximada en los instantes de tiempo $t_1, t_2, \cdots, t_N$.

$$\tilde{x}_1 \approx x(t_1),\ \tilde{x}_2 \approx x(t_2), \cdots, \tilde{x}_N \approx x(t_N),$$

La distancia $h_k \triangleq t_{k+1} - t_k$ se denomina paso de integración, y puede ser constante o variable, según el método.

Método de Euler

Aproximando la derivada por el cociente incremental, puede escribirse

$$\frac{x(t_{k+1}) - x(t_k)}{t_{k+1} - t_k} \approx \dot{x}(t_k) = f(x(t_k), t_k)$$

Tomando $h \triangleq t_{k+1} - t_k$ (h fijo) puede despejarse

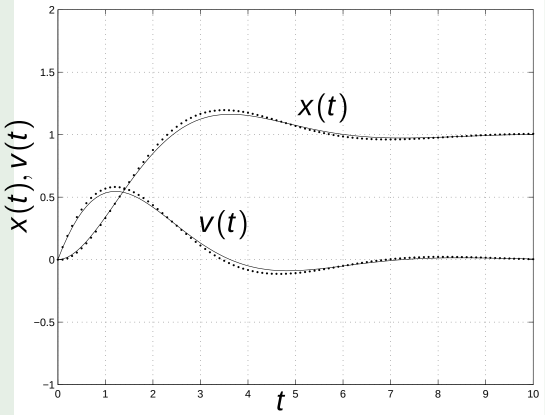
$$x_{k+1} = x_k + h\, f(x_k, t_k) \tag{7}$$

Luego, conociendo x_0, pueden obtenerse $x_1, x_2, \cdots, x_N$ de forma iterativa.

La fórmula de Euler (7) define una Ecuación en Diferencias (Sistema de Tiempo Discreto).

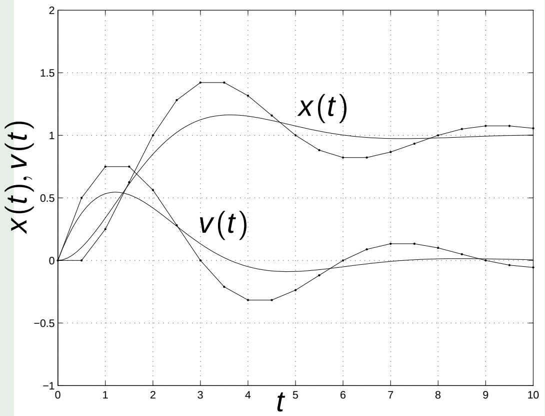
Método de Euler – Ejemplo

Solución con Euler de la Ecuación (1) ($h = 0.1$)



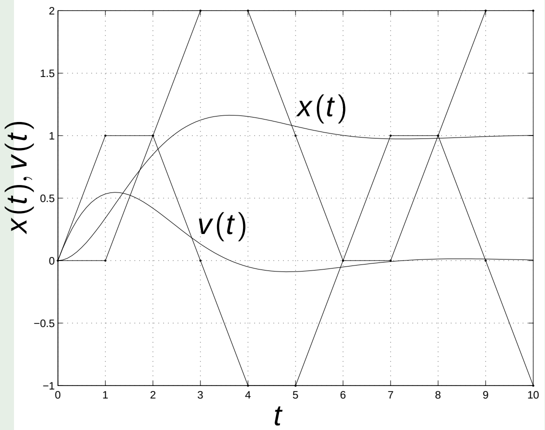
Método de Euler – Ejemplo

Solución con Euler de la Ecuación (1) ($h = 0.5$)



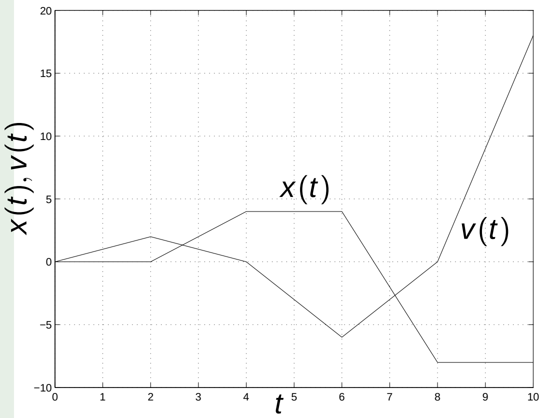
Método de Euler – Ejemplo

Solución con Euler de la Ecuación (1) ($h = 1$)



Método de Euler – Ejemplo

Solución con Euler de la Ecuación (1) ($h = 2$)



Error y Estabilidad Numérica

En todos los casos, la solución numérica tuvo un error apreciable.

El error local por truncamiento es el que se comete de un paso al siguiente. En general aumenta al aumentar el paso h.

Además, con $h = 2$ la solución numérica se tornó inestable.

Una solución es numéricamente estable si no diverge cuando $k \to \infty$

Es deseable que la estabilidad numérica coincida con la estabilidad analítica de la solución. Evidentemente, en el método de Euler esto depende del paso h.

Orden de Precisión

La expansión en serie de Taylor de la solución exacta de la EDO (6) en torno a x_k es:

$$x_{k+1} = x_k + h \cdot f(x_k, t_k) + \frac{h^2}{2!}\frac{df}{dt}(x_k, t_k) + \frac{h^3}{3!}\frac{d^2 f}{dt^2}(x_k, t_k) + \ldots \quad (8)$$

El orden de precisión de un método es la máxima potencia de h hasta la cual coinciden las soluciones exacta y numérica.

El método de Euler es entonces un método de primer orden

Cuanto mayor es el orden de un método, menor es el error local por truncamiento.

Métodos Monopaso

Son métodos que calculan x_{k+1} utilizando únicamente información sobre x_k. (Métodos de Runge–Kutta)

- Forward Euler (primer orden):

$$x_{k+1} = x_k + h \cdot f(x_k, t_k)$$

- Backward Euler (primer orden):

$$x_{k+1} = x_k + h \cdot f(x_{k+1}, t_{k+1})$$

- Regla Trapezoidal (segundo orden):

$$x_{k+1} = x_k + 0.5 \cdot h \cdot [f(x_{k+1}, t_{k+1}) + f(x_k, t_k)]$$

- Heun (segundo orden):

$$k_1 = f(x_k, t_k),\ k_2 = f(x_k + h \cdot k_1, t_k + h),\ x_{k+1} = x_k + 0.5 \cdot h \cdot (k_1 + k_2)$$

Métodos Multipaso

Son métodos que calculan x_{k+1} utilizando información sobre x_k y sobre algunos puntos anteriores (x_{k-1}, etc).

- Adams–Bashforth 3 (tercer orden):

$$x_{k+1} = x_k + \frac{h}{12}(23 \cdot f_k - 16 \cdot f_{k-1} + 5 \cdot f_{k-2})$$

- Backward Difference Formulae (BDF) 3 (tercer orden):

$$x_{k+1} = \frac{18}{11}x_k - \frac{9}{11}x_{k-1} + \frac{2}{11}x_{k-2} + \frac{6}{11}h \cdot f_{k+1}$$

Nota: llamamos $f_k \triangleq f(x_k, t_k)$.

Métodos Implícitos

Los métodos implícitos utilizan información del futuro para calcular x_{k+1}, y por lo tanto requieren resolver una ecuación en cada paso.

Los métodos de Backward Euler, la Regla Trapezoidal y BDF3 son ejemplos de métodos implícitos.

Los métodos implícitos tienen grandes ventajas en relación a la estabilidad numérica.

Como contrapartida, su implementación requiere de algoritmos iterativos para resolver la ecuación implícita en cada paso.

Sistemas Stiff (Rígidos)

Son sistemas que contienen simultáneamente dinámica lenta y dinámica rápida.

En principio, la idea sería usar un paso chico al comienzo y luego agrandarlo cuando la dinámica rápida desaparece.

El problema es que los métodos explícitos se tornan numéricamente inestables al agrandar el paso h.

Por esto, con los sistemas stiff deben utilizarse exclusivamente métodos implícitos provistos de algoritmos de control de paso.

Sistemas Marginalmente Estables

Son sistemas que están en el límite de la estabilidad analítica.

Ej: el sistema masa resorte (1) sin fricción ($b = 0$), sistemas de dinámica celeste, etc. En estos casos:

- Los métodos explícitos resultan numéricamente inestables.
- Los métodos implícitos en general resultan numéricamente estables.

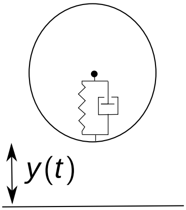
Se necesita utilizar métodos implícitos especiales denominados F–estables como la Regla Trapezoidal

Sistemas Discontinuos

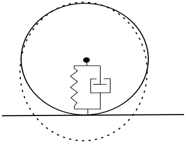
Un modelo simple de una pelota que cae y rebota contra el piso es el siguiente:

$$\dot{y}(t) = v(t)$$

$$\dot{v}(t) = \begin{cases} -g & \text{si } y(t) > 0 \\ -g - \frac{k}{m} \cdot y(t) - \frac{b}{m} \cdot v(t) & \text{si } y(t) \leq 0 \end{cases}$$



Esta EDO tiene una discontinuidad en $y = 0$.



Los métodos de integración pueden cometer errores inaceptables. Es necesario detectar los instantes en que $y(t) = 0$ y recomenzar la simulación a partir de allí.

Resumen métodos clásicos

En general todos los métodos de integración clásicos estiman el valor de los estados en un futuro a partir de la información sobre el valor actual de los estados, es decir:

Dado x en $t = t_k$, los métodos estiman iterativamente el valor de x en sucesivos t_{k+1} cada uno con una aproximación distinta.

Organización de la Presentación

Ejemplo Introductorio

Consideremos el sistema de segundo orden:

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= -x_1(t) \end{aligned} \tag{9}$$

y la siguiente *aproximación*:

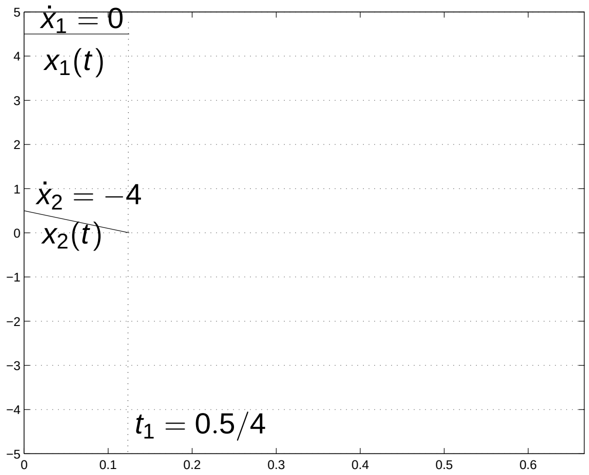
$$\begin{aligned} \dot{x}_1(t) &= \text{floor}(x_2(t)) = q_2(t) \\ \dot{x}_2(t) &= -\text{floor}(x_1(t)) = -q_1(t) \end{aligned} \tag{10}$$

Ejemplo Introductorio

La Ecuación

$$
\begin{aligned}
\dot{x}_1(t) &= q_2(t) \\
\dot{x}_2(t) &= -q_1(t)
\end{aligned}
$$

puede resolverse. Consideremos las c.i. $x_1(0) = 4.5$, $x_2(0) = 0.5$:

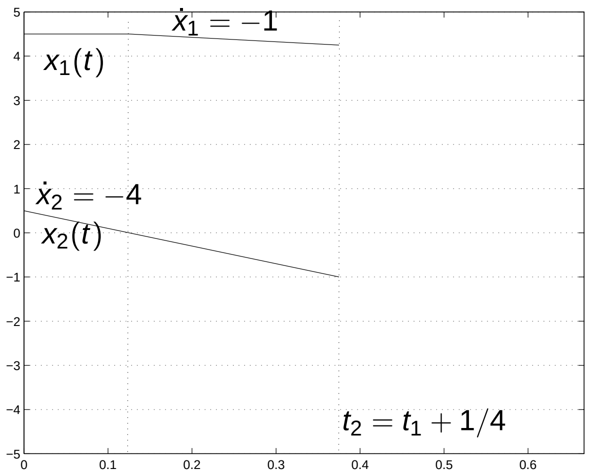


Ejemplo Introductorio

La Ecuación

$$\begin{aligned}\dot{x}_1(t) &= q_2(t)\\ \dot{x}_2(t) &= -q_1(t)\end{aligned}$$

puede resolverse. Consideremos las c.i. $x_1(0) = 4.5$, $x_2(0) = 0.5$:

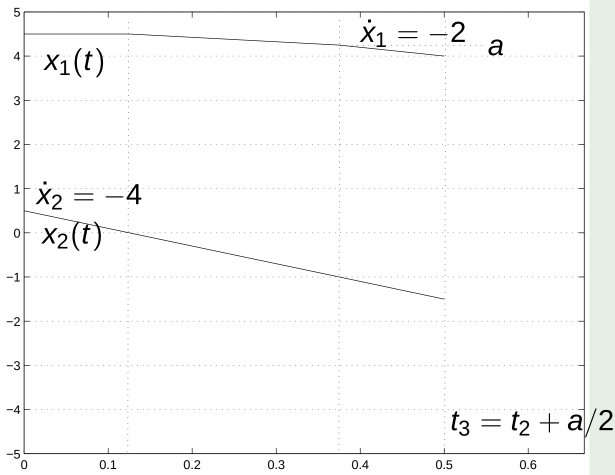


Ejemplo Introductorio

La Ecuación

$$\begin{aligned}\dot{x}_1(t) &= q_2(t)\\ \dot{x}_2(t) &= -q_1(t)\end{aligned}$$

puede resolverse. Consideremos las c.i. $x_1(0) = 4.5$, $x_2(0) = 0.5$:

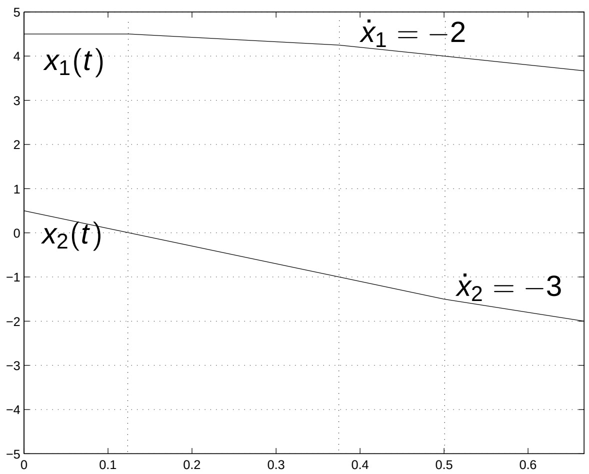


Ejemplo Introductorio

La Ecuación

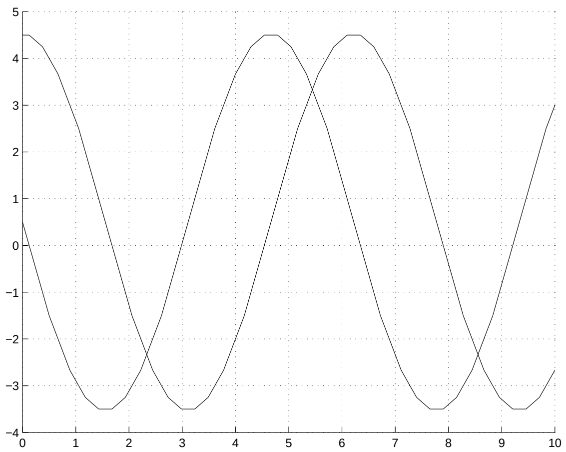
$$\begin{aligned}\dot{x}_1(t) &= q_2(t)\\ \dot{x}_2(t) &= -q_1(t)\end{aligned}$$

puede resolverse. Consideremos las c.i. $x_1(0) = 4.5$, $x_2(0) = 0.5$:



Ejemplo Introductorio

Solución del Sistema Cuantificado



Sistemas Cuantificados y DEVS.

A diferencia de los métodos de integración vistos, el sistema aproximado (10) no puede escribirse como una Ecuación en Diferencias:

$$x(t_{k+1}) = f(x(t_k), t_k)$$

sino como un *Sistema Cuantificado*:

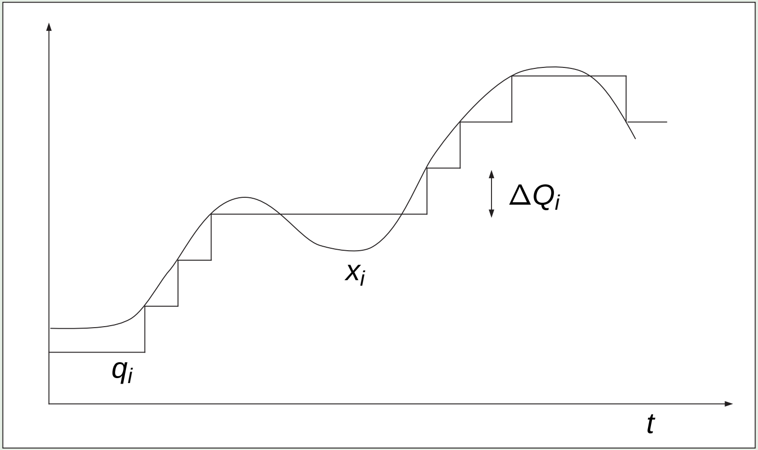
$$\dot{x}(t) = f(q(t), t)$$

Como veremos, los Sistemas Cuantificados son equivalentes a modelos de Eventos Discretos DEVS

$$M = (X, Y, S, \delta_{\mathrm{int}}, \delta_{\mathrm{ext}}, \lambda, ta)$$

Método de QSS

Función de Cuantificación con Histéresis



Método de QSS

Definición

Dado un sistema

$$\dot{x}(t) = f(x(t), u(t)) \tag{11}$$

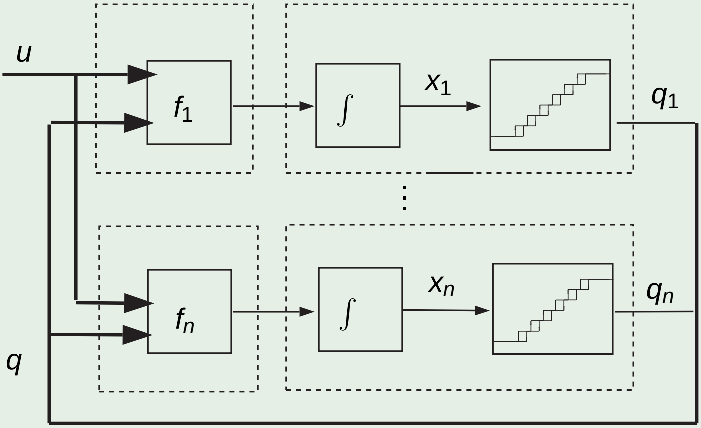
con $x \in \mathbb{R}^n$, $u \in \mathbb{R}^m$ y $f : \mathbb{R}^n \to \mathbb{R}^n$, la aproximación QSS está dada por

$$\dot{x}(t) = f(q(t), u(t)) \tag{12}$$

donde $q(t)$ y $x(t)$ están vinculadas componente a componente por funciones de cuantificación con histéresis.

El QSS (12) es equivalente a un modelo DEVS *Legítimo*.

QSS – Diagrama de Bloques



Propiedades de QSS

Definiendo $\Delta x(t) \triangleq q(t) - x(t)$, la Ec.(12) puede reescribirse como

$$\dot{x}(t) = f(x(t) + \Delta x(t), u(t))$$

que es muy similar a (11), excepto por la perturbación acotada Δx. Luego resulta:

- Convergencia: El error tiende a 0 cuando la cuantificación $\Delta Q \rightarrow 0$.
- Estabilidad *práctica*: Cuando el sistema original es estable, las soluciones quedan en un entorno del punto de equilibrio.
- **Cota de Error Global Calculable!**: En sistemas lineales, se puede acotar el error cometido en función de la cuantificación.

QSS – Ejemplo

La aproximación QSS del sistema masa resorte (1) puede escribirse como

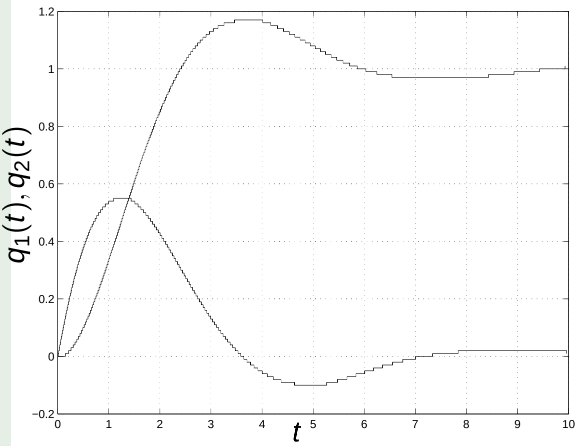
$$\begin{aligned} \dot{x}_1(t) &= q_2(t) \\ \dot{x}_2(t) &= -\frac{k}{m} q_1(t) - \frac{b}{m} q_2(t) + \frac{1}{m} F(t) \end{aligned} \tag{13}$$

Para los parámetros utilizados ($m = b = k = 1$), la cota de error global puede calcularse como

$$\begin{bmatrix} |e_1(t)| \\ |e_2(t)| \end{bmatrix} \leq 2.3094 \cdot \begin{bmatrix} \Delta Q_1 + \Delta Q_2 \\ \Delta Q_1 + \Delta Q_2 \end{bmatrix} \tag{14}$$

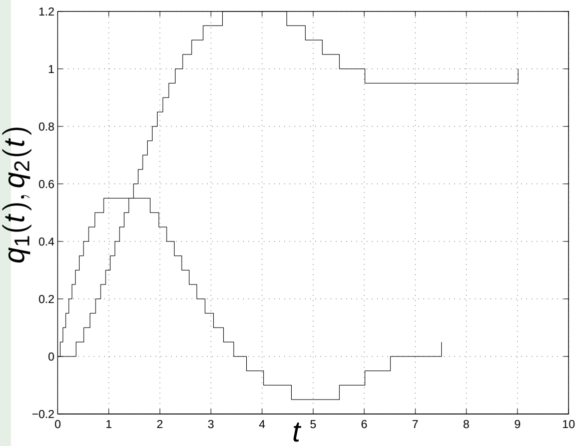
QSS – Ejemplo

Solución con $\Delta Q = 0.01$



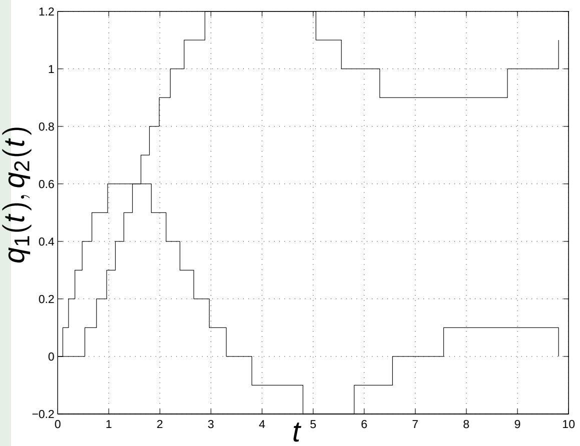
QSS – Ejemplo

Solución con $\Delta Q = 0.05$



QSS – Ejemplo

Solución con $\Delta Q = 0.1$



QSS – Características

Ventajas

- Estabilidad y Cota de Error.
- Descentralización (sólo cálculos locales). Explota ralitud
- Grandes ventajas para simular sistemas discontinuos

Desventajas

- Aparición de oscilaciones. Problemas en sistemas stiff.
- Necesidad de elegir el quantum.
- El número de pasos crece linealmente con la precisión.

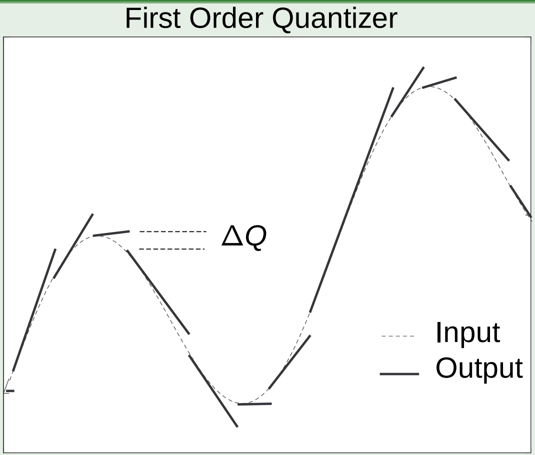
Resumen métodos de cuantificación

Los métodos de integración por cuantificación, (a diferencia de los clásicos) estiman **en qué momento** los estados asumirán un cierto valor:

Dado x en $t = t_k$, los métodos por cuantificación estiman en que t_{k+1} el estado x llega a x'.

Método de QSS2

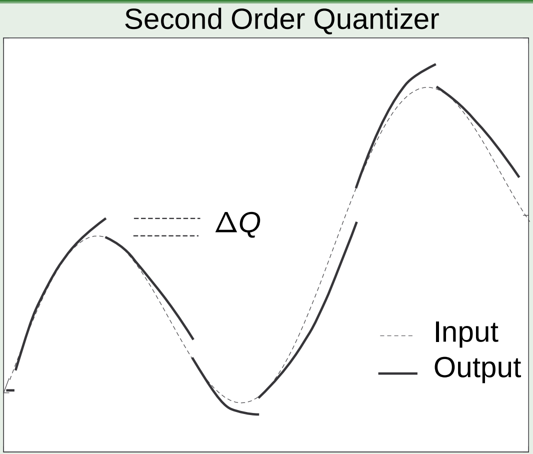
Cuantificación de primer orden



- Mismas propiedades y ventajas que QSS.
- Método de segundo orden.
- El número de pasos crece con la raíz cuadrada de la precisión.

Método de QSS3

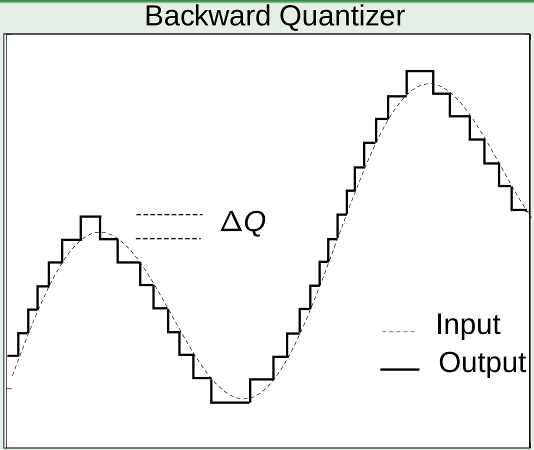
Cuantificación de segundo orden



- Mismas propiedades y ventajas que QSS.
- Método de tercer orden.
- El número de pasos crece con la raíz cúbica de la precisión.

Método de Backward QSS

Cuantificación *Backward*

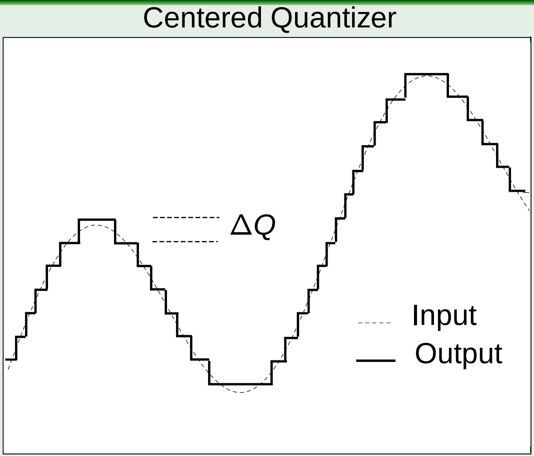


BQSS:

- Similares propiedades y ventajas que QSS.
- Método de primer orden.
- No produce oscilaciones, y sirve para sistemas stiff.

Método de Centered QSS

Cuantificación *Centrada*



CQSS:

- Similares propiedades y ventajas que QSS.
- Método de primer orden.
- Es F-estable y sirve para sistemas marginalmente estables.

Principales Aplicaciones de los Métodos de QSS

- En sistemas de Electrónica de Potencia permiten acelerar las simulaciones en más de un orden de magnitud respecto de los métodos clásicos.
- En Redes Neuronales Pulsantes (Spiking Neural Networks) logran un costo lineal con el tamaño de la red. Este costo es cuadrático en los métodos clásicos.
- Pueden utilizarse para discretizar controladores garantizando estabilidad práctica de manera incondicional.
- Reducen notablemente el tiempo de simulación en sistemas con retardos (Delay Differential Equations).

Organización de la Presentación

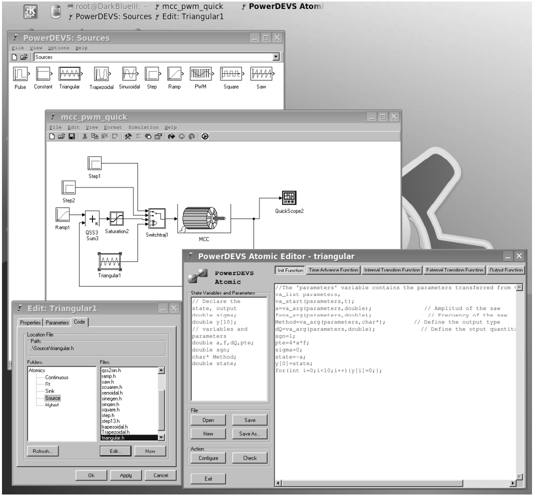
Implementación de los Métodos

- Los métodos de integración QSS pueden ser implementados en cualquier simulador DEVS o pueden ser implementados como un solver stand-alone.
- En nuestro grupo de investigación desarrollamos una herramienta para la simulación de sistemas híbridos basado en DEVS, llamada PowerDEVS.
- En PowerDEVS se implementan todas las variantes de los métodos de QSS, brindando la posibilidad de simular los modelos descriptos de modo off-line o en tiempo real (bajo un RTOS).

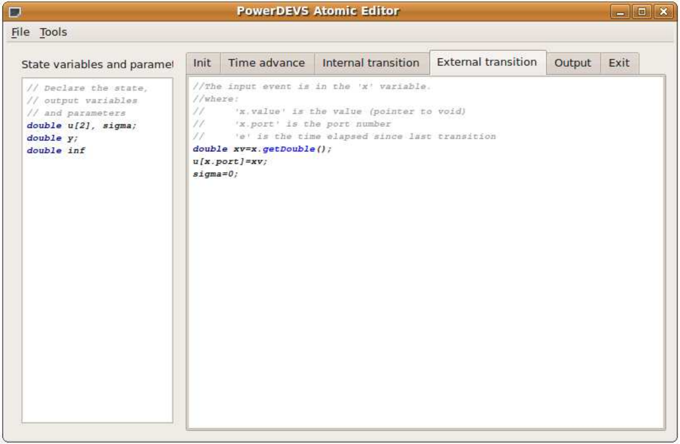
PowerDEVS

- PowerDEVS es una herramienta de propósito genera para el modelado y simulación DEVS orientada a sistemas híbridos.
- Ofrece una simple GUI donde se pueden describir modelos DEVS arrastrando y conectando bloques predefinidos de una librería.
- El usuario avanzado puede también definir sus propios bloques en C++, y usarlos en modelos DEVS.
- Una vez terminado, el modelo DEVS puede ser simulado en tiempo real o en modo off-line.

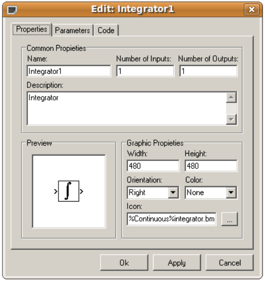
PowerDEVS - Captura

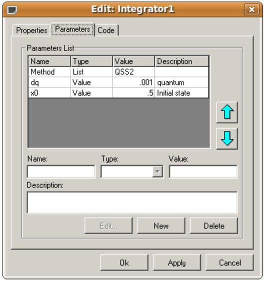


PowerDEVS - Editor de atómicos

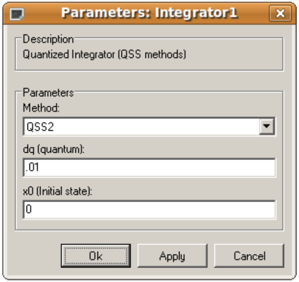


PowerDEVS - Edición de bloques



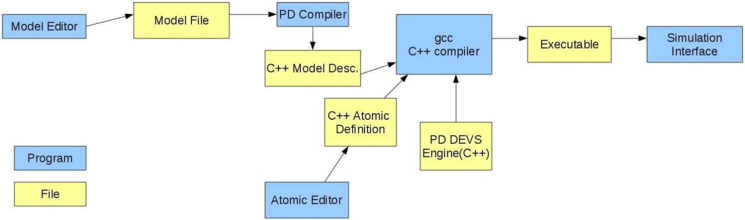


PowerDEVS - Parámetros



PowerDEVS - Interfaz de simulación

PowerDEVS - Internamente



PowerDEVS - Librería

PowerDEVS posee una librería de modelos atómicos y acoplados predefinidos que incluyen:

Basic Elements: Esta es la única librería fundamental de PowerDEVS. Todas las otras librerías utilizan de alguna forma bloques de esta librería. Contiene modelos atómicos y acoplados vacíos y dos objetos especiales llamados puertos de entrada y de salida utilizado para el modelado de modelos acoplados.

Continuous: Esta librería contiene los bloques que procesan señales continuas basadas en la aproximación de QSS. Incluye integradores, ganancias, funciones no lineales, multiplicadores, etc.

PowerDEVS - Librería

Discrete: Esta librería incluye modelos para simular sistemas de tiempos discretos en PowerDEVS. Aquí se encuentra el bloque de retraso discreto.

Hybrid: En esta categoría existen un conjunto de modelo que combinan funciones discretas y continuas utilizados para simular sistemas híbridos como quantificadores (QSS), switches, comparadores, sampleadores, etc.

Real Time: Esta librería contiene bloques específicos para hacer uso de las características de tiempo real de PowerDEVS como sincronización y manejo de interrupciones.

PowerDEVS - Librería

Source: Aquí encontramos diferentes generadores de señales continuas (aproximadas por QSS) como senoidales, rampas, pulsos, ondas cuadradas, etc.

Sink: Esta librería contiene diferente bloques sumideros donde los resultados de la simulación pueden ser enviados para su visualización o procesamiento.

PowerDEVS - Librería

Todos los bloques de la librería de PowerDEVS están implementados para soportar los distintos métodos de QSS, es decir, procesan señales cuantificadas a través de la cuantificación de estados de primer, segundo o tercer orden.

PowerDEVS - RTAI

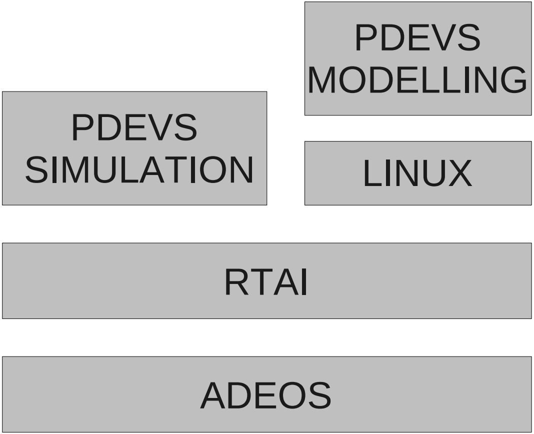
RTAI

RTAI es un RTOS que soporta varias arquitecturas (i386, PowerPC, ARM). Es una extensión del kernel de Linux que permite ejecutar tareas de tiempo real concurrentemente con procesos de Linux normales.

PowerDEVS puede generar binarios para ser ejecutados en RTAI. Un modelo en tiempo real puede:

- Usar sincronización temporal precisa ($\approx$ 2 micro segundos) para emitir eventos.
- Acceder dispositivos de hardware y manejar IRQ's.
- Utilizar un reloj de tiempo real con precisión de nano segundos.

PowerDEVS - RTAI



PowerDEVS - RTAI

PowerDEVS ofrece una manera de modelar, diseñar y simular sistemas híbridos en tiempo real como:

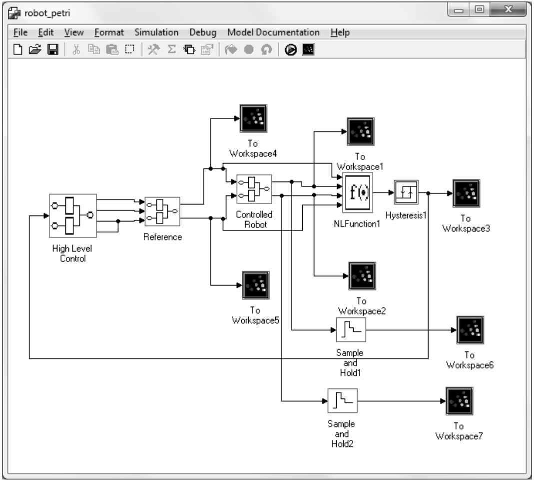
- Controladores digitales.
- Simulaciones Hardware-in-the-loop y Man-in-the-loop.
- Detección de fallas y monitoreo de sistemas.

PowerDEVS & Scilab

Scilab puede ser usado junto con PowerDEVS como un workspace, esto es:

- Los parámetros de los modelos de PowerDEVS pueden ser expresiones de Scilab como *“3*u*%pi”*.
- Los modelos atómicos puede llamar cualquier método de Scilab en su implementación interna, por ejemplo para calcular la inversa de una matriz.
- Los resultados de la simulación puede ser enviados a Scilab para su futuro uso en análisis estadístico, visualización, etc.

PowerDEVS & Scilab



Organización de la Presentación

Trabajo Actual en el Tema

Nuestro grupo en el CIFASIS investiga sobre:

- Desarrollo de métodos de QSS.
- Aplicación de los métodos en sistemas híbridos: Electrónica de potencia, redes de datos, spiking neural networks, etc.
- Implementación de lo métodos en tiempo real para su uso en control automático.
- Simulación de modelos descriptos en Modelica a través del uso de los métodos de QSS.
- Desarrollo y mantenimiento de la herramienta PowerDEVS.

Trabajo Actual en el Tema

Algunos proyectos relacionados con la LCC y con Ing. Electrónica:

- Implementación de los métodos QSS como un solver stand-alone.
- Modelado y Simulación de Redes de Petri utilizando el formalismo DEVS.
- Desarrollo de una nueva GUI para la edición de modelos de PowerDEVS.
- Implementación en hardware de un muestreador asincrónico.
- Diseño, simulación e implementación de sistemas de control de movimiento en tiempo real.
- Desarrollo de sistemas control basado

Bibliografía Sobre Métodos de QSS

E. Kofman.
A Second Order Approximation for DEVS Simulation of Continuous Systems.
Simulation, 78(2):76–89, 2002.

E. Kofman.
Quantization–Based Simulation of Differential Algebraic Equation Systems.
Simulation: Transactions of the Society for Modeling and Simulation International, 79(7):363–376, 2003.

E. Kofman.
Discrete Event Simulation of Hybrid Systems.
SIAM Journal on Scientific Computing, 25(5):1771–1797, 2004.

E. Kofman and S. Junco.
Quantized State Systems. A DEVS Approach for Continuous System Simulation.
Transactions of SCS, 18(3):123–132, 2001.

B. Zeigler, T.G. Kim, and H. Praehofer.
Theory of Modeling and Simulation. Second edition.
Academic Press, New York, 2000.

Bibliografía Sobre Métodos de QSS

F. Bergero and E. Kofman.
PowerDEVS. A Tool for Hybrid System Modeling and Real Time Simulation.
Simulation: Transactions of the Society for Modeling and Simulation International, 2010. in Press.

R. Castro, E. Kofman, and F. Cellier.
Quantization-Based Integration Methods for Delay Differential Equations.
Simulation Modelling Practice and Theory, 19:314–336, 2010.

F. Cellier and E. Kofman.
Continuous System Simulation.
Springer, New York, 2006.

E. Kofman.
A Third Order Discrete Event Simulation Method for Continuous System Simulation.
Latin American Applied Research, 36(2):101–108, 2006.

G. Migoni and E. Kofman.
Linearly Implicit Discrete Event Methods for Stiff ODEs.
Latin American Applied Research, 39(3):245–254, 2009.

Simulación de sistemas por cuantificación de estados.

JCC 2010 - FCEIA - UNR

Ernesto Kofman - Federico Bergero

Laboratorio de Sistemas Dinámicos y Procesamiento de la Información
FCEIA - UNR.
CIFASIS - CONICET