

Un modelo CellDEVS para la Aplicación del Método de Líneas a Ecuaciones Diferenciales Parciales

Matías Alejandro Nitsche

7 de septiembre de 2009

Índice

1. Introducción	3
2. Ecuaciones Diferenciales	3
2.1. Condiciones Iniciales y de Borde	4
3. El Método de Líneas	4
3.1. Primer Paso: Discretizando Variables	4
3.2. Segundo Paso: Aproximaciones de Derivadas	5
3.3. Tercer Paso: Reescritura de la Ecuación	6
4. Mol con CellIDEVS	6
4.1. Integración de ODEs	7
5. Un Ejemplo: Ecuación de Calor	8
5.1. Aplicación de MOL	8
5.2. Modelo CellIDEVS	9
5.2.1. Reglas	10
5.2.2. Vecindad	11
5.3. Estabilidad Numérica	11
6. Resultados	11
7. Discusión	13
A. Método de Elementos Finitos	14
B. Código de conversión DRW→Matlab	14

1. Introducción

Los modelos de sistemas continuos son aquellos en los cuales las magnitudes o *estados* asociados a los mismos pertenecen a un conjunto de valores no-discretos. Estos modelos en general están formulados matemáticamente en términos de ecuaciones diferenciales, las cuales pueden ser, o bien *Ecuaciones Diferenciales Ordinarias* (ODEs) o *Ecuaciones Diferenciales Parciales* (PDEs). Éstas últimas son las más comunes ya que los modelos suelen describir como varían los procesos respecto del tiempo y de alguna otra variable (el espacio, por ejemplo). Dado que estas ecuaciones en general no pueden ser resueltas analíticamente, en la gran mayoría de los casos se debe recurrir inevitablemente a métodos numéricos.

Existen diversos métodos de aproximación numérica para atacar problemas descritos en términos de PDEs, entre los cuales se encuentra el llamado *Método de Líneas* o MOL. Éste, permite llevar una PDE a un conjunto de ODEs que aproximen la solución. Esto permite aprovechar alguno de los numerosos métodos existentes para integración de ODEs, que ya se encuentran muy establecidos y analizados. Otro método también muy utilizado es el *Método de Elementos Finitos* o FEM, el cual será brevemente descripto y contrapuesto al *Método de Líneas*.

El objetivo principal de este trabajo es mostrar cómo puede ser resuelto un sistema de ODEs (como resultado de aplicar MOL a una PDE) utilizando el simulador CD++. De este modo se da una pauta de cómo atacar algunos problemas modelados mediante PDEs, aplicando lo que se sabe para los Sistemas de Eventos Discretos (como contraposición de los Continuos).

Desde ya se aclara que no es posible dar un esquema de resolución general para cualquier PDE, dada la complejidad de las mismas. La resolución y/o aproximación de PDEs abarca una gran teoría, por lo que este trabajo se limitará a detallar el método y sus fundamentos, para luego aplicarlo a un caso concreto. Se intentará generalizar lo más posible para posibilitar su aplicación a otros casos, pero es inevitable que queden algunos no contemplados. El *Método de Líneas* es un método más (con pros y contras), y debe ser tomado como tal.

2. Ecuaciones Diferenciales

Una ecuación diferencial es una ecuación que relaciona a una función incógnita f con sus derivadas de distintos órdenes. Cuando la función f tiene una sola variable independiente x (y, por ende, las derivadas son respecto de ésta) se puede hablar de una *Ecuación Diferencial Ordinaria* u ODE (por sus siglas en inglés). Por ejemplo, la *Segunda Ley de Newton* se puede expresar mediante una ODE de la siguiente forma:

$$m \frac{\partial^2 x(t)}{\partial t^2} = F(x(t))$$

En esta ecuación se muestra cómo depende la posición x de una partícula de masa m a lo largo del tiempo t , de la fuerza F a la que es sometida. En este caso $x(t)$ es la función incógnita.

En otros casos, cuándo la función incógnita tiene más de una variable independiente, y sus derivadas son ahora parciales (respecto de las distintas variables) se está hablando de una *Ecuación Diferencial Parcial* o PDE. Un ejemplo (que se utilizará a lo largo del trabajo) es uno caso particular de la *Ecuación de Difusión*, la llamada *Ecuación de Calor* (en una variable espacial):

$$\frac{\partial u}{\partial t} = \sigma \frac{\partial^2 u}{\partial x^2} \quad (1)$$

En este caso $u(x, t)$ es la función incógnita. Esta ecuación viene a modelar la distribución de la temperatura en el espacio x , y su variación en el tiempo t . La constante σ corresponde a la conductividad térmica propia del material en cuestión.

2.1. Condiciones Iniciales y de Borde

En general, cuando se modela un proceso mediante ecuaciones diferenciales lo que interesa es obtener la solución para un rango sobre cada una de las variables independientes. Por lo tanto, el problema debe estar asociado a valores que toma la función incógnita en los extremos de esos rangos.

En general, se distinguen dos tipos de variables: las de *valor inicial* y las de *valor de borde*. Las variables de *valor inicial* son aquellas que pertenecen típicamente a intervalos de la forma $[a, +\infty)$. Un caso claro de una variable de este tipo es t , el tiempo. Por otro lado, las de *valor de borde* son las que pertenecen a intervalos de la forma $[a, b]$. Ejemplo de estas últimas es la variable espacial x .

Los valores que toma la función incógnita para cada tipo de variables son llamadas las condiciones *iniciales* y *de borde*, para los tipos respectivos. Dada la función incógnita $u(x, t)$, una condición inicial para t tendría la siguiente forma:

$$u(x, t = a) = y_0$$

o incluso:

$$u(x, t = a) = f(x)$$

Análogamente, se tendrían dos condiciones para x : una sobre $u(x = x_0, t)$ y otra sobre $u(x = x_f, t)$, asumiendo que el rango de interés para x es $[x_0, x_f]$. En otros casos, es posible que las condiciones de borde no estén dadas directamente sobre u , sino sobre alguna de sus derivadas.

3. El Método de Líneas

Se tiene una función incógnita $u(x^{(1)}, \dots, x^{(k)}, t)$, y una PDE sobre la misma. Como se mencionó previamente, si bien las soluciones a la PDE corresponden a las posibles funciones u , al estar hablando de modelos que pretenden ser aplicados y evaluados, el interés real es evaluar u (o alguna de sus derivadas) sobre distintos rangos de interés para cada una de sus variables.

El método consiste en particionar en intervalos regulares el rango de cada variable $x^{(i)}$, obteniendo así un conjunto de valores discretos para cada variable. El segundo paso es plantear aproximaciones para las derivadas parciales respecto de los $x^{(i)}$. Las aproximaciones permitirán reemplazar las apariciones de estas derivadas parciales en la PDE original.

Las discretizaciones de las variables $x^{(i)}$ efectivamente definen una grilla de puntos en \mathbb{R}^{n-1} . Sobre esta grilla se evalúa la nueva PDE (es decir, la que incluye las aproximaciones), obteniendo así una serie de ecuaciones diferenciales cuya única variable independiente es t . Dado que la única derivada que todavía queda en cada ecuación es con respecto de esta misma variable, las ecuaciones son ahora ODEs. La solución aproximada para u se obtiene mediante algún método de integración numérica de ODEs.

A continuación se detallan los pasos mencionados.

3.1. Primer Paso: Discretizando Variables

Dados estos rangos interés previamente mencionados, el primer paso del método consiste en particionar éstos en intervalos regulares. Esto se hace para cada variable $x^{(i)}$. La última variable (será la variable t en la mayoría de los casos) se excluye de este paso, ya que justamente el método se intenta *deshacer* de todas las derivadas respecto de todas las variables menos una.

Cada rango $[x_0^{(i)}, x_f^{(i)}]$ será particionado en $n - 1$ intervalos de ancho $\Delta x^{(i)}$, donde:

$$\Delta x^{(i)} = \frac{x_f^{(i)} - x_0^{(i)}}{n - 1}$$

obteniéndose así n puntos $\{x_0^{(i)}, \dots, x_{n-1}^{(i)}\}$ (separados entre sí por $\Delta x^{(i)}$), donde $x_{n-1}^{(i)} = x_f^{(i)}$. Con este paso, se está discretizando cada variable $x^{(i)}$.

En adelante, para simplificar, se considerará una función con una única variable x , para reducir la notación. El procedimiento para funciones de más variables es fácilmente generalizable.

3.2. Segundo Paso: Aproximaciones de Derivadas

Lo que se quiere ahora es encontrar aproximaciones para las apariciones de derivadas parciales respecto de x en la PDE.

La primera aproximación que se puede obtener es la que surge casi inmediatamente de la definición de derivada parcial de primer orden:

$$\frac{\partial u}{\partial x}(x, t) = \lim_{h \rightarrow 0} \frac{u(x + h, t) - u(x, t)}{h}$$

Por lo tanto, si se considera un h fijo muy pequeño ($h > 0$), es válido decir que:

$$\frac{\partial u}{\partial x}(x, t) \approx \frac{u(x + h, t) - u(x, t)}{h} \quad (2)$$

Visto de otra forma, se está aproximando la pendiente de la recta tangente a u , en el punto (x, t) , mediante la secante que pasa por $x + h$ y x .

La previa aproximación se conoce como *forward difference* (ya que hace uso del punto $x_1 = x + h$, que está *adelante* de x en la recta real). Análogamente, existen la llamada *backward difference*:

$$\frac{\partial u}{\partial x}(x, t) \approx \frac{u(x, t) - u(x - h, t)}{h}$$

Utilizando estas dos aproximaciones, se puede obtener una tercera, comunmente conocida como *central difference*:

$$\frac{\partial u}{\partial x}(x, t) \approx \frac{u(x + h, t) - u(x - h, t)}{2h} \quad (3)$$

En general, se pueden derivar fórmulas de aproximación desarrollando, para la función u , el Polinomio de Taylor de grado n . Asimismo, se puede obtener también el error de aproximación para el mismo, notado como R_n . De este modo se puede reescribir la ecuación (2) de manera exacta como sigue:

$$\frac{\partial u}{\partial x}(x, t) = \frac{u(x + h, t) - u(x, t)}{h} + R_1(x, t)$$

En general, dado que el resto nunca será calculado, sino que será analizado asintóticamente, se usa la notación-O (conocida por su aplicación en la complejidad de algoritmos). Con ésta, se busca expresar cómo depende el resto de h . En el caso de tratarse de un resto lineal respecto de h , se habla de $O(h)$. Es decir, es un método de aproximación de primer orden. Dado que para $n = 1$, el resto es lineal con h , efectivamente:

$$\frac{\partial u}{\partial x}(x, t) = \frac{u(x + h, t) - u(x, t)}{h} + O(h)$$

La aproximación de la ecuación (3) es en realidad es un método de segundo orden, es decir $O(h^2)$. Dado que $h < 1$, un resto $O(h)$ es, en principio, menos preciso que uno de $O(h^2)$. Utilizando el Polinomio de Taylor se pueden conseguir aproximaciones de otros órdenes, en especial para cuando se tienen derivadas de segundo orden o mayores. Sin embargo, utilizar una aproximación de orden mayor no garantiza que sea un método numericamente estable, como se verá más adelante en el caso analizado.

3.3. Tercer Paso: Reescritura de la Ecuación

El último paso consiste en reemplazar las apariciones de derivadas respecto de x (del lado derecho de la PDE) por alguna de las aproximaciones. Para cada evaluación posible de x sobre los distintos puntos de la grilla discreta se tiene una ecuación distinta. Éstas tienen como única variable independiente a t , conformando así un sistema de ODEs. En otras palabras, se pasa una PDE de la forma:

$$\frac{\partial u}{\partial t}(x, t) = \dots$$

a un sistema de ODEs de la forma:

$$\frac{\partial u}{\partial t}(x_i, t) = \dots \quad (4)$$

Ahora, si se define:

$$u'_i(t) \triangleq \frac{\partial u}{\partial t}(x_i, t)$$

el lado derecho de las ecuaciones (4) será función de distintos $\{u_j\}$, dependiendo de la aproximación utilizada en cada caso. Por ejemplo, para una *forward difference*, se puede escribir:

$$u'_i(t) = f(t, u_i(t), u_{i+1}(t))$$

En realidad, como las aproximaciones no dependen t en sí, se puede decir incluso que:

$$u'_i(t) = f(u_i(t), u_{i+1}(t))$$

Para los extremos del intervalo de las variable no-temporales entran en juego las condiciones de borde. En el caso en el que éstas estén dadas en términos del valor de la función u misma, no habrá que resolver ninguna ecuación ni plantear ninguna aproximación. Sin embargo, en los casos en los que las condiciones estén en términos de derivadas de u , habrá que replantear las aproximaciones utilizadas teniendo en cuenta las mismas (evitando así, evaluar las diferencias finitas sobre puntos fuera del intervalo).

4. Mol con CellDEVS

Partiendo de una ecuación diferencial a la que se le aplicó MOL, se quiere poder definir un modelo CellDEVS en base al sistema de ODEs que se obtiene. La intención es poder simular este modelo CellDEVS en CD++ y que los resultados obtenidos permitan obtener el valor de la función incógnita sobre cada punto de la grilla.

Dada la función incógnita $u : \mathbb{R}^k \rightarrow \mathbb{R}$, el modelo CellDEVS se definirá como n -dimensional (nótese que se toman todas las variables). Cada dimensión i tendrá un largo en celdas igual a la cantidad de valores discretos que toma la variable $x^{(i)}$, una vez aplicado MOL. El espacio de coordenadas de las celdas serán tuplas de \mathbb{N}_0^k . Las reglas se definirán de manera tal de que el resultado final sea que cada coordenada $c \in \mathbb{N}_0^k, c = (c_0, \dots, c_{k-1})$ tenga el valor de $u(x_{c_0}^{(0)}, \dots, x_{c_{k-2}}^{(k-2)}, t_{c_{k-1}})$. Por ejemplo, teniendo una función $u(x, t)$, el espacio de celdas será bidimensional, como se observa en la figura 1.

Otra posibilidad es definir el espacio de celdas dentro de \mathbb{N}_0^{k-1} , excluyendo la variable temporal. En este caso, el valor de t en cada caso, estará dado por el tiempo de simulación T del modelo en sí. La ventaja en este caso, es que el espacio de celdas puede llegar a resultar bastante más chico. Sin embargo, tomar la totalidad de las dimensiones tiene el beneficio de poder observar directamente el valor de la función sobre todo su dominio. Tomar una u otra definición del espacio de celdas es indistinto. En este caso se tomó la primera por cuestiones de comodidad.

Por último, para poder integrar las ODEs es necesario o bien implementar alguno de los métodos existentes directamente dentro del conjunto de reglas, o bien extender el lenguaje del simulador. El problema

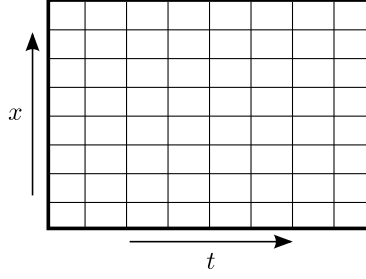


Figura 1: Espacio de celdas para el caso de una función incógnita $u(x, t)$

que esto último conlleva es que las librerías que implementan integración de ODEs están diseñadas generalmente para que se las llame de forma iterativa. Es decir, el algoritmo guarda un estado interno, que permite ir obteniendo el valor de la función sobre cada punto de integración definido. Si bien, en principio, no sería imposible hacer alguna adaptación que permita utilizar dichos algoritmos desde el lenguaje de reglas, esta tarea excede al trabajo.

4.1. Integración de ODEs

Se decidió utilizar un método de integración perteneciente a la familia Runge-Kutta. En este caso se utilizó la versión de cuarto orden (es decir, utiliza cuatro evaluaciones para aproximar la derivada).

Este método sirve para aproximar ODEs del tipo *valor inicial*. Esto es, se tiene una ecuación

$$y'(t) = f(t, y(t))$$

donde $y(t)$ es la función incógnita y $f : \mathbb{R} \times \mathbb{R} \rightarrow \mathbb{R}$. Se dice que es de *valor inicial*, ya que se tiene el dato de $y(t_0) = c$. El método plantea cómo obtener $y(t_{i+1})$ a partir de $y(t_i)$ (o sea, es un método iterativo), según:

$$y(t_{i+1}) = y(t_i) + \frac{1}{6}\Delta t(k_1 + 2k_2 + 2k_3 + k_4)$$

donde

$$\begin{aligned} \Delta t &= t_{i+1} - t_i \\ k_1 &= f(t_i, y) \\ k_2 &= f(t_i + \frac{1}{2}\Delta t, y + \frac{1}{2}\Delta t k_1) \\ k_3 &= f(t_i + \frac{1}{2}\Delta t, y + \frac{1}{2}\Delta t k_2) \\ k_4 &= f(t_i + \Delta t, y + \Delta t k_3) \end{aligned}$$

Sin embargo, para resolver sistemas de ODEs, hay que aplicar la variante correspondiente del método. En esta versión, el problema de *valor inicial* incluye n ecuaciones de la forma:

$$u_i'(t) = f_i(t, u_0, \dots, u_{n-1})$$

con n condiciones iniciales de la forma:

$$u_i(t_0) = c_i$$

El método ahora consiste en:

$$u_i(t_{j+1}) = u_i(t_j) + \frac{1}{6}\Delta t(k_{1i} + 2k_{2i} + 2k_{3i} + k_{4i})$$

donde

$$\begin{aligned}
k_{1i} &= f_i(t_j, u_0, \dots, u_{n-1}) \\
k_{2i} &= f_i(t_j + \frac{1}{2}\Delta t, u_0 + \frac{1}{2}\Delta tk_{1i}, \dots, u_{n-1} + \frac{1}{2}\Delta tk_{1i}) \\
k_{3i} &= f_i(t_j + \frac{1}{2}\Delta t, u_0 + \frac{1}{2}\Delta tk_{2i}, \dots, u_{n-1} + \frac{1}{2}\Delta tk_{2i}) \\
k_{4i} &= f_i(t_j + \Delta t, u_0 + \Delta tk_{3i}, \dots, u_{n-1} + \Delta tk_{3i})
\end{aligned}$$

En el caso de MOL, la función f será alguna combinación lineal de distintas diferencias finitas.

5. Un Ejemplo: Ecuación de Calor

En esta sección se detallará la aplicación de MOL al caso concreto de la Ecuación de Calor (ecuación (1)), para luego construir un modelo CellDEVS que, mediante su simulación en CD++, permita obtener la función incógnita. Esta ecuación es muy utilizada en diversas disciplinas, para modelar diversos procesos. Existen numerosos análisis sobre su aproximación. De todos modos, sirve como un ejemplo claro y no trivial para aplicar MOL.

La formulación completa del problema (en una dimensión espacial) es la siguiente. Se plantea la ecuación con incógnita u :

$$\frac{\partial u}{\partial t}(x, t) = \sigma \frac{\partial^2 u}{\partial x^2}(x, t)$$

sujeta a las condiciones auxiliares:

$$u(x, 0) = 100 \sin\left(\frac{\pi x}{c}\right) \quad (5)$$

$$u(0, t) = 0 \quad (6)$$

$$u(c, t) = 0 \quad (7)$$

En donde se toma que $c = 1$ y $\sigma = 1$. En otras palabras, se podría considerar que se está tratando una barra (que se puede considerar unidimensional) de longitud 1, a la cual se le aplica calor (máximo en su centro, y difundiendo hacia los lados), mientras que se mantiene sus extremos a temperatura cero. La dinámica del sistema consiste en una disipación del calor conforme al paso del tiempo.

Se pueden, entonces, considerar los siguientes intervalos de interés:

$$x \in [0, 1]$$

$$t \in [0, 1]$$

Sin embargo, se verá luego que por cuestiones numéricas, se tomará un intervalo menor para t .

5.1. Aplicación de MOL

Como fue mencionado, el primer paso consiste en discretizar todas las variables menos una, que en este caso (como en la mayoría) será t . Si se toma a n como la cantidad de valores discretos para x , el ancho de los intervalos será de:

$$\Delta x = \frac{x_f - x_0}{n - 1} = h$$

Lo que sigue es aproximar las derivadas parciales respecto de x con alguna de las diferencias finitas. Utilizando el Polinomio de Taylor, y desarrollando éste para u hasta orden 2, se obtiene la siguiente aproximación:

$$\frac{\partial^2 u}{\partial x^2}(x, t) \simeq \frac{u(x + \Delta x, t) - 2u(x, t) + u(x - \Delta x, t)}{\Delta x^2} \quad (8)$$

Ésta, se conoce como *central difference* de segundo orden. La diferencia entre esta aproximación y la de la ecuación (3) consiste en que ésta última se obtiene a partir de las diferencias *forward* y *backward*, mientras que la de la ecuación (8) se obtiene directamente del Polinomio de Taylor.

Evaluando esta última ecuación sobre los n puntos, se puede escribir ahora:

$$\frac{\partial^2 u}{\partial x^2}(x_i, t) \simeq \frac{u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t))}{\Delta x^2}$$

Esta aproximación vale para todos los puntos intermedios, para los casos $x = x_0$ y $x = x_{n-1}$, se sabe en este caso que $u(x, t) = 0$. De esto resulta el siguiente de $n - 2$ ODEs:

$$u(x_0, t) = 0 \quad (9)$$

$$\frac{\partial u}{\partial t}(x_i, t) \simeq \frac{u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t))}{\Delta x^2}, 0 < i < n - 1 \quad (10)$$

$$u(x_{n-1}, t) = 0 \quad (11)$$

, ya que en cada ecuación la derivada es respecto de t , la única variable independiente. Nótese que habiendo reescrito la PDE con aproximaciones a la derivada parcial, la incógnita no es exactamente u , sino una aproximación a ésta.

5.2. Modelo CellDEVS

Para el caso en particular que se está resolviendo, el espacio de celdas del modelo será bidimensional. Tomando n y m como la cantidad de valores discretos para x y t respectivamente, el espacio de celdas será un rectángulo de $n \times m$ (filas \times columnas). Se debe recordar que la variable t no corresponde al tiempo de simulación T en esta definición.

Tomando, como antes, $u(x_i, t) \triangleq u_i(t)$ y $\Delta x \triangleq h$, las ecuaciones (9)-(11) se reescriben como:

$$u_0(t) = 0 \quad (12)$$

$$u'_i(t) = \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t))}{h^2}, 0 < i < n - 1 \quad (13)$$

$$u_{n-1}(t) = 0 \quad (14)$$

Teniendo en cuenta entonces que, $\forall i \in (0, n - 1)$:

$$u'_i(t) = g(t, u_{i-1}(t), u_i(t), u_{i+1}(t))$$

los factores $k_{1,i}$ a $k_{4,i}$ del método de Runge-Kutta resultan en:

$$\begin{aligned} k_{1,i} &= \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t))}{h^2} \\ k_{2,i} &= \frac{(u_{i+1}(t) + \frac{1}{2}\Delta t k_{1,i}) - 2(u_i(t) + \frac{1}{2}\Delta t k_{1,i}) + (u_{i-1}(t) + \frac{1}{2}\Delta t k_{1,i}))}{h^2} \\ k_{3,i} &= \frac{(u_{i+1}(t) + \frac{1}{2}\Delta t k_{2,i}) - 2(u_i(t) + \frac{1}{2}\Delta t k_{2,i}) + (u_{i-1}(t) + \frac{1}{2}\Delta t k_{2,i}))}{h^2} \\ k_{4,i} &= \frac{(u_{i+1}(t) + \Delta t k_{3,i}) - 2(u_i(t) + \Delta t k_{3,i}) + (u_{i-1}(t) + \Delta t k_{3,i}))}{h^2} \end{aligned}$$

lo que se reduce a:

$$\begin{aligned} k_{1,i} &= \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2} \\ k_{2,i} &= \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2} \\ k_{3,i} &= \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2} \\ k_{4,i} &= \frac{u_{i+1}(t) - 2u_i(t) + u_{i-1}(t)}{h^2} \end{aligned}$$

De este modo queda definido:

$$u_i(t_{j+1}) = u_i(t_j) + \frac{1}{6}\Delta t(k_{1,i} + 2k_{2,i} + 2k_{3,i} + k_{4,i})$$

que, por las ecuaciones obtenidas, en todos los casos se reduce a:

$$u_i(t_{j+1}) = u_i(t_j) + \Delta t k_{1,i}$$

A esta altura es evidente que el método de Runge-Kutta de cuarto orden, en este caso, es equivalente al llamado *Método de Euler*. En primer lugar, esto se debe a que f y g no dependen de t , lo que implica que no hay diferencia entre k_2 y k_3 . Y en segundo lugar, al usar diferencias finitas, los coeficientes $\frac{1}{2}\Delta t k_{2,i}$ y $\Delta t k_{3,i}$ se terminan cancelando.

Finalmente, las celdas del modelo CellDEVS se definen como:

$$C(i, j) = u_j(t_i) \quad (15)$$

(notar que esto implica que en la primera columna se encontrarán los valores iniciales).

5.2.1. Reglas

El conjunto de reglas correspondientes pueden verse en la siguiente definición del modelo CellDEVS, en el lenguaje interpretado por el simulador CD++ (donde los *macros* N, M, DX y DT corresponden a los valores de $n, m, \Delta x$ y Δy , respectivamente):

Algorithm 1 Reglas del modelo CellDEVS

```

1 rule : { 0 } 1 { cellpos(0) = (#Macro(N) - 1) and cellpos(1) = 0 }
2 rule : { 100 * sin( PI * cellpos(0) * #Macro(DX) ) } 1 { cellpos(1) = 0 }
3 rule : { 0 } 1 { cellpos(0) = 0 or cellpos(0) = (#Macro(N) - 1) }
4 rule : { (0, -1) + ( #Macro(DT) / ( #Macro(DX) * #Macro(DX) ) ) *
5         ( (-1, -1) - 2 * (0, -1) + (1, -1) ) } 1 { t }
```

En las primeras dos líneas se está expresando las condiciones iniciales del sistema. La condición de la segunda línea evaluará a *true* únicamente en la primera columna, dando a dichas celdas el valor correspondiente a la ecuación (5). El propósito de la regla de la primera línea es el de evitar la inconsistencia introducida por el error numérico asociado a la evaluación de $100\sin(\pi x_f)$ (que debería ser 0). Para mantener la simetría con el extremo x_0 , se fuerza a 0 el valor erróneo.

La tercera regla corresponde a la aplicación de las condiciones de borde dadas por las ecuaciones (6) y (7). Por la condición utilizada, esta regla se aplicará únicamente en la primera y última fila del espacio de celdas.

Por último, la cuarta regla aplicará sobre el resto de las celdas, aplicando la dinámica dada por la ecuación (13).

5.2.2. Vecindad

La vecindad definida para este modelo (y para cualquier otro que resulte de aplicar MOL), estará dada por el tipo de diferencia finita que se utilizó en las aproximaciones. En el ejemplo dado, al usar la *central difference* de segundo orden, para evaluar la regla principal (correspondiente a dicha diferencia) se requieren tres valores de: $u_i(t_{j-1}), u_{i-1}(t_{j-1}), u_{i+1}(t_{j+1})$. Teniendo en cuenta la asignación de valores de u_i a las celdas del modelo (ecuación (15)), la vecindad para este caso resulta como se ve en la figura 2.

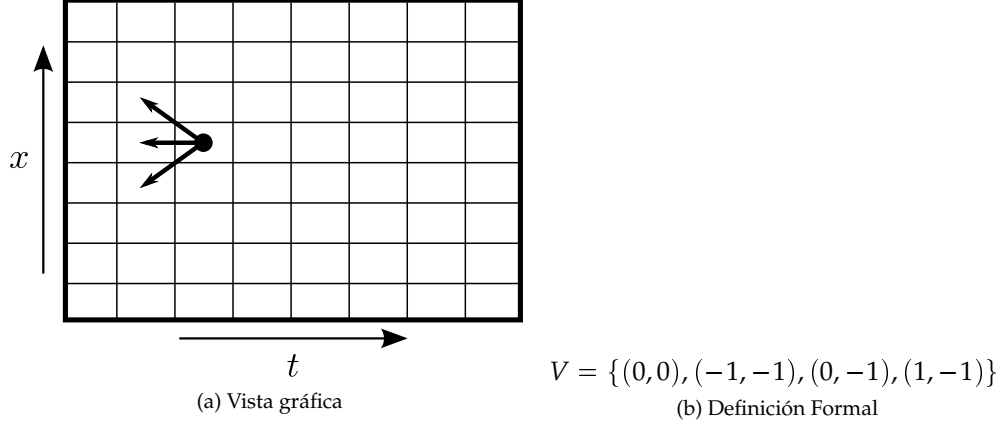


Figura 2: Definición de la vecindad para el caso de la Ecuación de Calor en una dimensión espacial

Hay que aclarar que, en realidad, la inclusión de la celda actual (es decir, $(0,0)$) en V se debe más que nada a una cuestión de costumbre. En este caso es evidente que no es necesario dicho valor, sin embargo en el simulador CD++, la inclusión de dicha celda en V es necesaria para su correcto funcionamiento.

5.3. Estabilidad Numérica

La estabilidad numérica de las aproximaciones usadas es un factor importante a tener en cuenta. En general, habrá métodos cuya estabilidad estará garantizada para ciertos sistemas de ecuaciones. En otros casos, la solución solo será estable solo cuando exista cierta relación entre los parámetros.

Para el caso especial del Método de Euler, aplicado a este sistema, se sabe que la solución sólo será estable si:

$$r = \frac{\Delta t}{\Delta x^2} < 0.5 \quad (16)$$

Esto implica que habrá que elegir n y m apropiadamente para cumplir con esta condición.

6. Resultados

Se definió un modelo CellDEVS, con las características mencionadas, utilizando el lenguaje del simulador CD++. Además, para verificar los resultados obtenidos, se implementaron las mismas ecuaciones en un script de MATLAB. Para emular la dinámica que rige el modelo CellDEVS utilizado, se utilizó un doble ciclo que itera la totalidad de las celdas, en el orden apropiado (que viene dado por la vecindad definida). Los resultados obtenidos fueron graficados mediante el MATLAB.

Los parámetros utilizados fueron los siguientes:

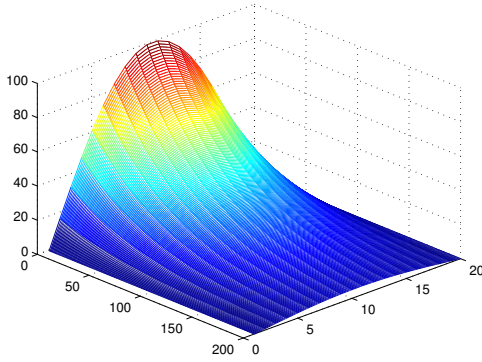
$$\begin{aligned}(n, m) &= (20, 200) \\ (t_0, t_f) &= (0, 0.3) \\ (x_0, x_f) &= (0, 1)\end{aligned}$$

De este modo, resulta que:

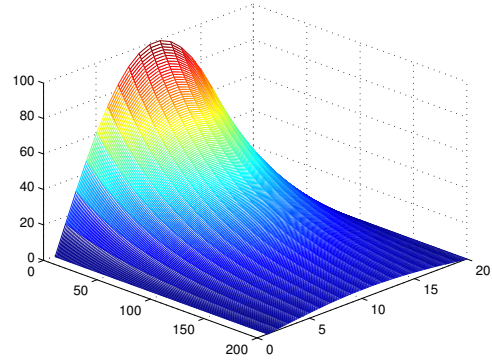
$$\begin{aligned}\Delta t &\simeq 0.00125 \\ \Delta x &\simeq 0.0526 \\ r &\simeq 0.4535\end{aligned}$$

Estando restringido por la condición de estabilidad de la ecuación (16), no es posible tomar un intervalo mayor para t sin incrementar también m . Los valores se eligieron de modo de cubrir la parte significativa de u , en la dimensión asociada a t y, a su vez, manteniendo un espacio de celdas no demasiado grande.

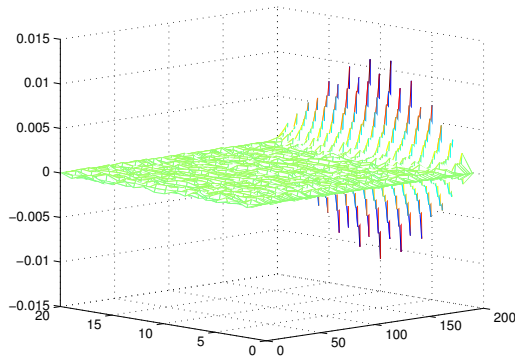
Con estos valores, los resultados obtenidos por el script de MATLAB y por el simulador CD++, junto al error absoluto y cuadrático entre ambos, se pueden ver en las siguientes figuras:



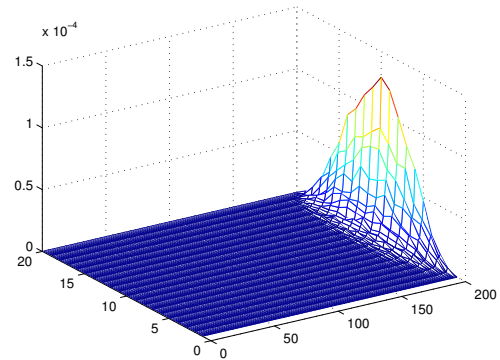
(a) Resultado obtenido con MATLAB



(b) Resultado obtenido con CD++



(c) Error absoluto



(d) Error cuadrático

Figura 3: Resultados de la aproximación de $u(x, t)$

7. Discusión

Con los resultados obtenidos se puede ver que el Método de Líneas parece muy apropiado para resolver problemas que involucren sistemas de ecuaciones diferenciales no-triviales utilizando los conceptos ya estudiados de Sistemas de Eventos Discretos. Más allá de que el método en sí sea relativamente simple, hay que tener en cuenta posibles inestabilidades numéricas a la hora de llegar a la simulación.

Los errores numéricos más importantes (comparando con la implementación de MATLAB), se dieron principalmente en los extremos de los intervalos de integración, lo que apunta a un arrastre de error de acumulación. Sin embargo, los resultados en general muestran que el simulador utilizado es más que capaz de resolver sistemas de estos tipos.

A. Método de Elementos Finitos

Existe otro método, muy usado en Ingeniería, que también permite encarar problemas de sistemas de PDEs, llamado FEM o Finite Element Method. Al igual que MOL, la idea base consiste en particionar las regiones de integración en unidades que pueden ser resueltas numéricamente. Sin embargo, FEM permite considerar geometrías de discretización mucho más complejas que MOL. Es decir, mientras MOL genera una discretización en donde cada aproximación se hace sobre un punto de la grilla, en FEM estas se hacen por *elementos*, que no son más que geometrías poligonales (triángulos, rectángulos, etc). En los vértices de estas geometrías se encuentran las soluciones aproximadas a la ecuación (análogas a las obtenidas por MOL en su grilla correspondiente), mientras que los elementos en sí tienen asociadas una función de interpolación a partir de los valores en sus vértices. Ésta función es la que rige la dinámica del sistema.

Si bien FEM puede ser aplicado también a CellDEVS[4], MOL tiene algunas características que resultan atractivas para obtener un *mapeo* a un modelo CellDEVS. En primer lugar, MOL es un método relativamente simple en comparación con FEM. Esta simplicidad tiene aparejada la limitación de MOL de no poder utilizar geometrías complejas como FEM, pero permite ser implementado fácilmente y con un relativamente bajo costo computacional. Por otro lado (y esta quizás es la diferencia más visible en la práctica), por la manera en que se define MOL, la solución aproximada se puede ir obteniendo a medida que se resuelve el sistema. En el caso típico donde existe una variable temporal y otras espaciales, MOL irá devolviendo en forma ordenada los resultados para t_i, t_{i+1} , etc. Por su parte, MOL consiste finalmente en la resolución de un sistema de ecuaciones que en general implica la utilización de un método iterativo.

B. Código de conversión DRW→Matlab

En este trabajo se utilizó un *script* escrito en el lenguaje de programación Ruby que permite generar una matriz en formato MATLAB a partir de un timeframe de una simulación CellDEVS con el software cdpp. Es decir, el input a este script consiste en la salida para un t específico del comando `drawlog`. A continuación se presenta el código correspondiente:

Algorithm 2 Script de conversión DRW→MATLAB

```
#!/usr/bin/ruby -w
puts 'A = ['
while (s = gets)
  o = s[/\|(.+)\|/,1]
  puts o + ';' unless o.nil?
end
puts '']'
```

Referencias

- [1] William F. Ames. *Numerical methods for partial differential equations*. Computer science and applied mathematics. Academic Press, 1977.
- [2] Lawrence C. Evans. *Partial Differential Equations*. Graduate Students in Mathematics. American Mathematical Society, 1998.
- [3] S. Hamdi, W. E. Schiesser, and G. W Griffiths. Method of lines. *Scholarpedia*, 2(7):2859, 2007.
- [4] H. Saadawi and G. Wainer. Modeling physical systems using finite element Cell-DEVS. *Simulation Modelling Practice and Theory*, 2007.