Ecuación de onda

caso unidimensional

Introducción

- La *ecuación de onda* es una ecuación diferencial en derivadas parciales de orden 2
- Permite modelar diversos fenómenos físicos:
  - Caso unidimensional: la vibración de una cuerda de guitarra
  - Caso bidimensional: el movimiento concéntrico de las olas en la superficie de un lago al arrojar una piedra
  - Caso tridimensional: la propagación de las ondas de choque en un terremoto
- La teoría se desarrolló en el siglo XVIII: d'Alembert descubrió la ecuación unidimensional en 1746 y 10 años después Euler descubrió la ecuación tridimensional

Modelo unidimensional

- El modelo unidimensional representa la vibración de una cuerda de guitarra
- El contexto es el siguiente
  - Consideremos coordenadas (x, y) en el plano cartesiano
  - Modelamos la cuerda en reposo paralela al eje x en $y = 0$ en el intervalo $x \in [0, L]$
  - Los extremos de la cuerda ($x = 0$, $x = L$) se mantienen fijos
  - Si tensamos la cuerda y la soltamos podemos modelar la vibración mediante una función $y = u(x, t)$ que describe la posición en y como función de x y del tiempo t

Modelo unidimensional

Ecuación

- $\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2}$ con $x \in (0, L)$ y $t \in (0, T]$

Condiciones iniciales

- $u(x, 0) = I(x)$ con $x \in [0, L]$
- $\frac{\partial}{\partial t} u(x, 0) = 0$ con $x \in [0, L]$

Condiciones de borde

- $u(0, t) = 0$ con $t \in (0, T]$
- $u(L, t) = 0$ con $t \in (0, T]$

La función $I(x)$ y la constante c son datos del problema.

Diferencias finitas

- Discretizar el dominio
- Reemplazar las derivadas por versiones discretas
- Definir la recurrencia para calcular nuevos puntos

Diferencias finitas: discretizar el dominio

- La ecuación que queremos resolver involucra a una función bivariada $u(x, t)$, su dominio es $[0, L] \times [0, T]$
- Discretizar en este caso significa muestrear el dominio bidimensional
  - Muestrear x: $0 = x_0 < x_1 < \cdots < x_{N_x - 1} < x_{N_x} = L$
  - Muestrear t: $0 = t_0 < t_1 < \cdots < t_{N_t - 1} < t_{N_t} = T$
  - Considerar la malla que surge del producto cartesiano: (x_i, t_n) con $i = 0, \ldots, N_x$ y $n = 0, \ldots, N_t$
  - Vamos a considerar que la malla de puntos del dominio es equidistante en las dos dimensiones; o sea: $x_{i+1} - x_i = \Delta x$, $t_{n+1} - t_n = \Delta t$

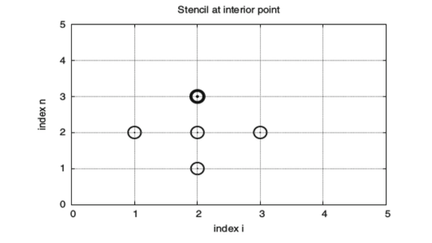

Fig. 2.1 Mesh in space and time. The circles show points connected in a finite difference equation

Diferencias finitas: reemplazar las derivadas

- Derivadas de orden 2:
  - $\frac{\partial^2}{\partial t^2} u(x_i, t_n) \sim \frac{u_i^{n+1} - 2\ u_i^n + u_i^{n-1}}{\Delta t^2}$
  - $\frac{\partial^2}{\partial x^2} u(x_i, t_n) \sim \frac{u_{i+1}^n - 2\ u_i^n + u_{i-1}^n}{\Delta x^2}$
- La versión discreta de la ecuación diferencial queda definida de este modo:

$$\frac{u_i^{n+1} - 2\ u_i^n + u_i^{n-1}}{\Delta t^2} = c^2 \frac{u_{i+1}^n - 2\ u_i^n + u_{i-1}^n}{\Delta x^2}$$

Diferencias finitas: definir la recurrencia

- Resolver la ecuación implica calcular la función u en el tiempo siguiente
- Introducimos una constante (para simplificar las cuentas): $C = c\frac{\Delta t}{\Delta x}$
- La recurrencia se define del siguiente modo (Verificar!):

$$u_i^{n+1} = -u_i^{n-1} + 2\ u_i^n + C^2(u_{i+1}^n - 2\ u_i^n + u_{i-1}^n)$$

Diferencias finitas: definir la recurrencia (cont.)

$$u_i^{n+1} = -u_i^{n-1} + 2\ u_i^n + C^2(u_{i+1}^n - 2\ u_i^n + u_{i-1}^n)$$

- Warning!:
  - Analizar qué pasa en t_1 (analizar la recurrencia en $n = 1$)
  - Analizar qué pasa en t_2 (analizar la recurrencia en $n = 2$)

Diferencias finitas: definir la recurrencia (cont.)

$$u_i^{n+1} = -u_i^{n-1} + 2\ u_i^n + C^2(u_{i+1}^n - 2\ u_i^n + u_{i-1}^n)$$

- Warning!:
  - Analizar qué pasa en t_1 (analizar la recurrencia en $n = 1$)
  - Analizar qué pasa en t_2 (analizar la recurrencia en $n = 2$)
- Solución!:
  - Usamos la condición inicial: $\frac{\partial}{\partial t} u(x, 0) = 0$
  - Su versión discreta es: $u_i^0 = \frac{u_i^1 - u_i^{-1}}{2\ \Delta t} = 0 \Rightarrow u_i^1 = u_i^{-1}$
  - Ejercicio: con esta nueva información despejar u_i^1

El algoritmo

1. Computar el vector de las posiciones iniciales: $u_i^0 = I(x_i)$ con $i = 0, \ldots, N_x$
2. Computar el vector u_i^1
   - Fijamos los extremos: $u_0^1 = u_{N_x}^1 = 0$
   - $u_i^1 = u_i^0 + \frac{C^2}{2}(u_{i+1}^0 - 2\ u_i^0 + u_{i-1}^0)$ con $i = 1, \ldots, N_x - 1$
3. Para cada tiempo $n = 1, \ldots, N_t - 1$ computar el vector u_i^n
   - Fijamos los extremos: $u_0^n = u_{N_x}^n = 0$
   - Aplicar la ecuación de recurrencia para calcular u_i^n con $i = 1, \ldots, N_x - 1$

Algunas curiosidades

- El algoritmo requiere sólo 3 vectores de tamaño N_x
- Notar que las 4 condiciones (2 iniciales y 2 de borde) fueron necesarias
  - Hay una relación entre el orden de las derivadas de la ecuación y la cantidad de condiciones que necesitamos para resolverla
- La generalización de la ecuación a más dimensiones se define del siguiente modo:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \nabla^2 u$$

  - u es una función multivariada: $u(x_1, \ldots, x_n, t) = y$
  - ∇^2 es el *operador laplaciano* y se define como: $\nabla^2 = \sum_{i=1}^{n} \frac{\partial^2 u}{\partial x_i^2}$

Bibliografía

- Finite Difference Computing with PDEs (Hans Peter Langtangen-Svein Linge, Springer, 2010)