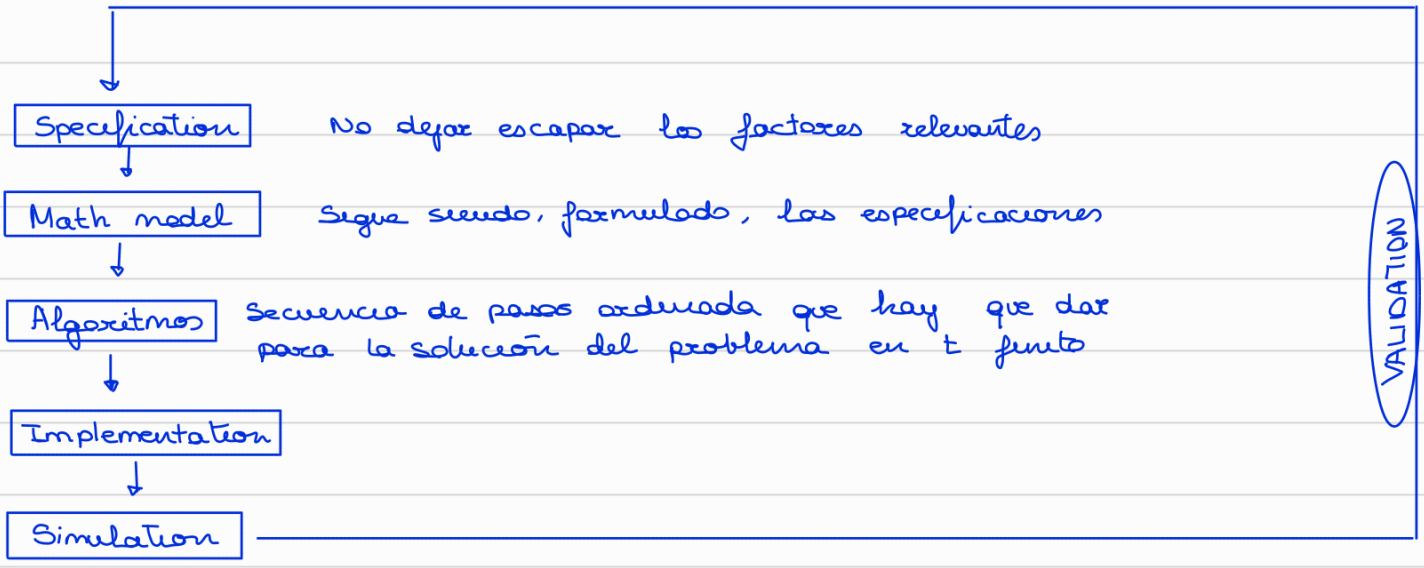


TODOS LOS PROBLEMAS DE CAUCHY SE EXPRESAN COMO

$$\begin{cases} \frac{dU}{dt} = f(U,t) \\ U(0) \end{cases} \qquad f: \mathbb{R}^N \times \mathbb{R} \longrightarrow \mathbb{R}^{N+1}$$

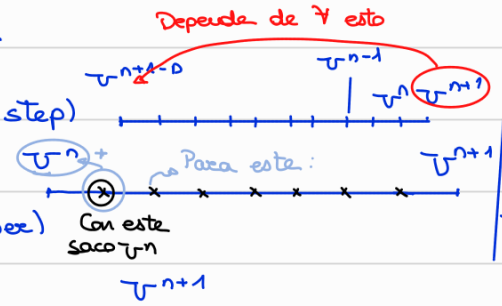
$$\begin{cases} U^{n+1} = U^n + \Delta t \cdot G(U^n, t_n) \\ U^0 \quad \text{dato} \end{cases}$$ Objetivo que queremos

TIPOS DE ESQUEMAS TEMPORALES

- Explícitos: U^{n+1} se estudian funciones ant. pasos
- Implícitos $H(U^{n+1}, \underbrace{U^n, U^{n-1}, \dots}_{\text{NO CONOCIDO}}) = 0$

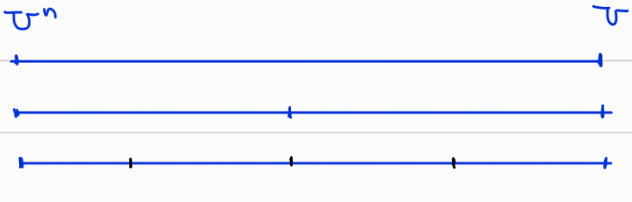
En explícitos nos encontramos:

- Adam - Bashforth (multistep)
- single step multietapa
  - Runge - Kutta
  - GBS (Gragg - Burlish - Sbez)



$$U^n = \begin{pmatrix} U^n \\ U^{n-1} \\ \vdots \\ U^{n+1-p} \end{pmatrix}$$

$$U^{n+1} = H(U^n)$$



Se va haciendo un mallado cada vez refinando más.
Se va así minimizando el error.

Ej Explícito de 2 pasos AB2

$$\underbrace{U^{n+1}}_{?} = U^n + \frac{\Delta t}{2}(3F^n - F^{n-1})$$

Ej Implícito Crank Nicolson

$$U^{n+1} = U^n + \frac{\Delta t}{2}\left(F^{n+1} + 1\right)$$

(?) (?)

Ahí tenemos la incógnita

U^{n+1} depende de f^{n+1}

¿QUÉ ERROR Y CUANTO COMETEN ESTOS MODELOS ?

Hay 2 tipos de errores

- Error local : $E^n = U(t_n) - U^n$

(1) $U^{n+1} = U^n + \Delta t\ G(U^n, t_n)$ (solu. numérica en t_n

(2) $U(t_{n+1}) = U^n + \Delta t\,(G(U(t_n), t_n) + \underbrace{T^{n+1}}_{\text{Error}}$ (solución exacta en t_n)

$\varepsilon \rightarrow$ error local de truncación

Residuo que deja la solución exacta cuando se introduce en el esq. numérico

- Error global :

(2) − (1) $\underbrace{U(t_{n+1}) - U^{n+1}}_{\text{Error global } T^{n+1}} = \underbrace{U(t_n) - U^n}_{\text{Error global en } T^n} + \Delta t\left[G(U(t_n), t_n) - G(U^n, t_n)\right] + \underbrace{T^{n+1}}_{\text{Error local}}$

$$\underbrace{E^{n+1}}_{\text{VECTORIAL}} = E^n + \Delta t \cdot \underbrace{\frac{\partial G}{\partial t}(\xi)}_{\text{MATRIZ}} \cdot \underbrace{E^n}_{\text{VECTOR COLUMNA}} + T^{n+1}$$

$$E^{n+1} = \underbrace{\left(I + \Delta t \frac{\partial G}{\partial U}(\xi)\right)}_{\text{MATRIZ } B} E^n + T^n$$

$$\boxed{E^{n+1} = B \cdot E^n + T^{n+1}}$$

Que el error global es : error del paso anterior (B atenua o ↑) + el error que se comenta al pasar al siguiente paso.

Ej $E^1 = B \cdot E^0 + T^1$

- E^0 → Podría ser $\neq 0$ y si no es 0 es que hay una falta de precisión.
- T^1 → Depende del esquema que empleemos (propiedad LOCAL DEL ESQUEMA)

OJO ⚠ Este no es igual al anterior

$E^2 = B E^1 + T^2$

$$E^2 = B(BE^0 + T^1) + T^2$$

$$E^3 = BE^2 + T^3 = B\left[B^2E^0 + BT^1 + T^2\right] + T^3$$

$$E^4 = BE^4 + T^4 = B\left(B^3E^0 + B^2T^1 + BT^2 + T^3\right] + T^4$$

$$= B^4E^0 + B^3T^1 + B^2T^2 + BT^3 + T^4$$

$$\boxed{E^n = B^nE^0 + \sum_{k=1}^{n} B^{n-k}\cdot T^k}$$

B^nE^0: Acumulación de los E locales de acumulación

$\sum_{k=1}^{n} B^{n-k}\cdot T^k$: El objetivo es que ↓

La norma de una matriz es la que induce la norma vectorial.

$$\|E^n\| = \|B^nE^0 + \sum_{k=1}^{n} B^{n-k}\cdot T^k\| \le \|B^nE^0\| + \sum_{k=1}^{n} \|B^{n-k}\cdot T^k\| \le$$

$$\le \|B^n\|\cdot\|E^0\| + \sum_{k=1}^{n} \|B^{n-k}\|\cdot\|T^k\| \le \|B^n\|\cdot\|E^0\| + \sup_{\forall k\in[1,n]} \|T^k\| \cdot \sum_{k=1}^{n} \|B^{n-k}\|$$

$$\|A\| = \sup \frac{\|Ax\|}{\|x\|} \quad \forall x$$

Si B es normal $\rightarrow B^T\cdot B = B\cdot B^T$

$\|B^n\| = \rho^n$ donde ρ es el radio espectral de la matriz B

→ módulo max de los autovalores

$\rho = \sup_{z\in\lambda(B)} |z|$

Entonces B depende de ρ

si $\rho\uparrow$ → amplificación

$\rho\downarrow$ → atenua

Por tanto, la norma del error sería

$$\|E^n\| \le \rho^n\varepsilon + \sup_{\forall k}\|T^k\| \sum_{k=1}^{n} \rho^{n-k} = \rho^n\varepsilon + \sup_{\forall k}\|T^k\| \cdot \left(\frac{1-\rho^n}{1-\rho}\right)$$

$\sum \rho^{n-k}$

$S_n = \rho^{n+1} + \rho^n + \dots + \rho + 1 \quad (1)$

$\rho S_n = \rho^n + \dots + \rho^2 + \rho \quad (2)$

(1 - 2)

$$(1-\rho)S_n = 1-\rho^n \rightarrow S_n = \frac{1-\rho^n}{1-\rho}$$

- Si $\rho > 1 \longrightarrow \|E^n\| \to \infty$ con $n\to\infty$ tiende a explotar
- Si $\rho < 0 \xrightarrow[n\to\infty]{} \|E^n\| \le \frac{\sup\|T^k\|}{1-\rho}$
- Si $\rho = 1 \longrightarrow \|E^n\| \le \varepsilon + \sup_{\forall k}\|T^k\|\cdot n$ (orbitas)

Discutir si es pequeño o no o si se puede acotar

PYTHON

Building blocks (ladrillos para hacer un programa).

- Asignación : Variable = expresión matemática
  Conj. de variables unidas entre sí mediante operadores

Las ecuaciones en ordenador NO EXISTEN x = 3 NO x ← 3
La flecha mete la cosa y dentro hay un 3.

y = x no lo lee
y = = x

x ← 3
y ← 8 print (y = = x) # TRUE
↳ Carga el nº 8 en la caja y

- Llamadas a funciones : que sean matemáticas
- Instrucciones o sentencias de control de flujo
  - if elif else
  - loops (bucles) } for
    while (cuando hay varios condicionantes)
- Input / Output print ("Hola world")
  input (a través de la consola)

Tipos de variables

- Numéricos
  - Real (float)
  - Integer
  - Complex
  - Vectores
  - Matrices
- Booleanos
- String : Cadena de caracteres name = "Juan"

En Python hay lenguaje no tipado (no da tipo de variable)
(OUCH Typing)

En x = 3.5 para saber que tipo es

print (type(x)) → Real

x = 3 print (type(x)) → entero

x = "Juan print (type(x)) → string

Una variable en python es un tipado que tiene una memoria del ensamblaje

Operadores

- Numéricos
  - aritméticos: +, −, ×, ÷, ** exponencial
    //, % → parte entera de una división; → resto de una división
  - Relacionales <, >, <=, =>, ==, != distinto
- Logical ==, !=, and or not
- String (+), *, (en) not en
  - (+) → concatenación
  - (en) → pertenece un String a otro?

"Juan" "Juan" "Juan"
"Rubio"

Ej algoritmo de $n! = n \cdot (n-1) \cdot (n-2) \cdots 1$

1957 — Función

```
def factorial (n)
    f = 1
    for i in range (1, n+1):
        f = f * i
    return f
```

cerrado → abierto: para que incluya el 5

Llamada factorial (4)

Cajones:
n ← 5
f ← 1
i ← 1
f ← 2
i ← 3
f ← 6
i ← 4
f ← 24

2023 → $n! = n \cdot (n-1)!$

```
def factorial (n) :
    if  n == 1 :
        return 1
    else
        return  factorial (n-1)*n
```