Problema oscilatorio

$$\left.\begin{array}{l} \ddot{x} + x = 0 \\ x(0) = 1 \quad \dot{x}(0) = 0 \end{array}\right\} \qquad \vec{U} = \begin{pmatrix} x \\ \dot{x} \end{pmatrix} \longrightarrow \underbrace{\frac{d}{dt}\begin{pmatrix} x \\ x' \end{pmatrix}}_{d\vec{U}/dt} = \underbrace{\begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}}_{A} \underbrace{\begin{pmatrix} x \\ \dot{x} \end{pmatrix}}_{\vec{U}}$$

vector estado

$$\begin{pmatrix} -\lambda & 1 \\ -1 & -\lambda \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \boxed{A\vec{W} = \lambda\vec{W}}$$

$$|A - \lambda I| = 0 \longrightarrow \lambda^2 + 1 = 0 \quad \begin{array}{l} \rightarrow \lambda_1 = +i \\ \rightarrow \lambda_2 = -1 \end{array} \qquad \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 1 \\ i \end{pmatrix} \qquad \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} -1 \\ i \end{pmatrix}$$

Los autovectores asociados

$$\begin{pmatrix} -i & 1 \\ -1 & -1 \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \qquad \begin{pmatrix} i & 1 \\ -1 & 1 \end{pmatrix}\begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

$$-i\,w_1 + w_2 = 0$$

Solución del sistema

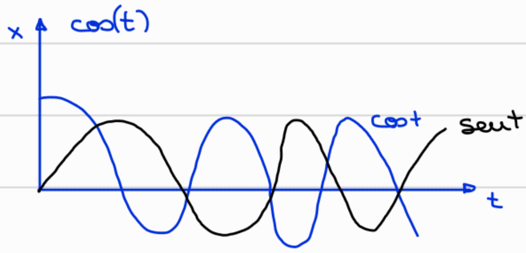
$$\boxed{\vec{U} = C_1\begin{pmatrix} 1 \\ i \end{pmatrix} e^{it} + C_2\begin{pmatrix} -1 \\ i \end{pmatrix} e^{-it}}$$

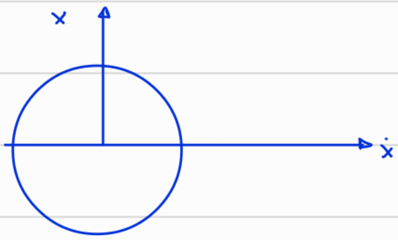
Condiciones iniciales

$$\begin{pmatrix} 1 \\ 0 \end{pmatrix} = C_1\begin{pmatrix} 1 \\ i \end{pmatrix} + C_2\begin{pmatrix} -1 \\ i \end{pmatrix} \rightarrow \begin{array}{l} 1 = C_1 - C_2 \rightarrow 1 = -2C_2 \rightarrow \\ 0 = C_1 i + C_2 i \rightarrow C_1 = -C_2 \end{array} \quad \begin{array}{l} C_2 = -1/2 \\ C_1 = 1/2 \end{array}$$

$$\vec{U} = \frac{1}{2}\begin{pmatrix} e^{it} + e^{-it} \\ ie^{it} - e^{-it} \end{pmatrix} = \begin{pmatrix} \overset{x}{\cos(t)} \\ \underset{\dot{x}}{-\operatorname{sen}(t)} \end{pmatrix}$$

Si lo representamos: $x = \cos t \qquad \dot{x} = -\operatorname{sen}(t) \rightarrow \boxed{x^2 + (\dot{x})^2 = 1}$





Si representamos x y $\dot{x}$, lo que tenemos son gráficas.

Lo que se observa es que la solución es siempre una gráfica donde la amplitud no cambia.

¿Qué ocurre si se aplica un esquema numérico como Euler?

<u>EULER</u> $\quad \vec{U}^{n+1} = \vec{U}^n + \Delta t \cdot A \cdot \vec{U}^n = \vec{U}^n \underbrace{(I + \Delta t \cdot A)}_{B}$

$$\overbrace{(I + \Delta t \cdot A)}^{B}\, W = r \cdot \vec{W}$$

$$\Delta t \cdot A \cdot \vec{W} = (r - 1) \cdot \vec{W} \quad \text{Es la identidad}$$

$$A \cdot \vec{W} = \underbrace{\left(\frac{r-1}{\Delta t}\right)}_{\lambda} \cdot \vec{W} \longrightarrow A \cdot \vec{W} = \lambda \cdot \vec{W}$$

└→ De aquí no hemos calculado los autovalores de B.

EDOS $\quad e^{\lambda_i t} \quad \lambda_i \in \Lambda(A)$

EED $\quad r_i^n \quad r_i \in \Lambda(B)$

└→ Ecuaciones en diferencias

$r \rightarrow$ complejo

el módulo < 0 para estable.

$r_1 = 1 + \Delta t\, i \longrightarrow w_1 = \begin{pmatrix} 1 \\ i \end{pmatrix}$

$r_2 = 1 - \Delta t\, i \longrightarrow w_2 = \begin{pmatrix} -1 \\ i \end{pmatrix}$

$v^n = k_1 \begin{pmatrix} 1 \\ i \end{pmatrix} x_1^n + k_2 \begin{pmatrix} -1 \\ i \end{pmatrix} x_2^n$

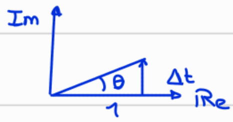
Las c.i $v^0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = k_1 \begin{pmatrix} 1 \\ i \end{pmatrix} r_1^0 + k_2 \begin{pmatrix} -1 \\ i \end{pmatrix} r_2^0$

$\begin{cases} k_1 - k_2 = 1 \\ k_1 + k_2 = 0 \end{cases}$ $\quad k_1 = 1/2$, $\quad k_2 = -1/2$

Vamos a escribir r_1 y r_2 en módulo y argumento:

$x_1 = \rho e^{i\theta}$ $\qquad \rho = \sqrt{1^2 + \Delta t^2}$

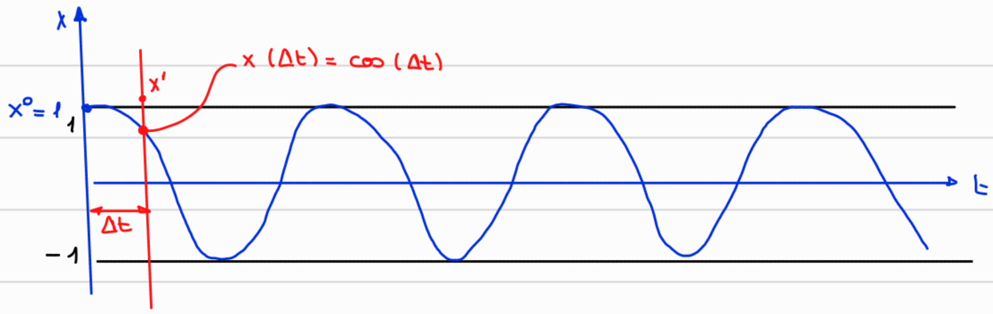
$x_2 = \rho e^{-i\theta}$



$\operatorname{tg} \theta = \Delta t \longrightarrow \theta = \operatorname{arctg}(\Delta t)$

$$v^n = \frac{1}{2} \begin{pmatrix} (\rho e^{i\theta})^n + (\rho e^{-i\theta})^n \\ i \cdot [(\rho e^{i\theta})^n - i(\rho e^{-i\theta})^n] \end{pmatrix} = \frac{1}{\cancel{2}} \cdot \rho^n \begin{pmatrix} \cancel{2} \cos(n\theta) \\ -\cancel{2} \operatorname{sen}(n\theta) \end{pmatrix} = \begin{pmatrix} \rho^n \cdot \cos(n\theta) \\ -\rho^n \cdot \operatorname{sen}(n\theta) \end{pmatrix}$$

$e^{i\theta} = \cos\theta + i \operatorname{sen}\theta$



• $x = \cos t$

$x^n = \rho^n \cos(n\theta) \longrightarrow$ cuando puntamos en python, en realidad habría puntos, no rectas (por defecto une esos puntos)

$x^0 = 1$

$x^1 = \rho \cdot \cos\theta \longrightarrow$ mayor que uno

$x^2 = \rho^2 \cdot \cos(2\theta) \longrightarrow$ más grande que antes

Hay un error de fase porque θ no es exactamente Δt

Por tanto el error en un punto n:

$$E^n = \cos(t_n) - \rho^n \cos(n\theta) = \cos(n\Delta t) - \underline{(1+\Delta t^2)^{n/2}} \cos(n\theta)$$

$t_n = cte$
$t_n = n\Delta t$

$n \to \infty$ tiende a ∞. Este es el error de AMPLITUD.

ERROR de FASE ya que $\operatorname{tg}\theta = \Delta t$

Esto significa que en el caso Euler, al aplicar a una

La amplitud aumenta con el Δt.
Para ello, en el caso Euler lo que se hace es hacer Δt pequeño para retrasar el crecimiento de la amplitud

(Pintar la gráfica desfasando el $\cos(t_n)$ y $\cos(n\theta)$)

PARA ORDENAR EL CÓDIGO

Argumento → ( ):
Imagen → return

Cuando se hable de módulos.

Para identificar modulos → lista de funcionalidad → especificación

FÍSICO

MATEMÁTICO

Temporal_schemes.py
Non_lineal_systems.py (Newton, bisección)
Cauchy_Problem.py

Ej: $\xrightarrow{a,b}$ $\boxed{\text{Integral}}$ → I imagen

f → función a integrar

$$I = \int_a^b f \, dx$$

" $\boxed{\text{Definir que es cada cosa}}$

```
def Integral(a, b, f):
    ⋮
    return I
```

Sino hay una interface es un lío pq no sabes que es cada cosa.

```
def ej(x):
    return x ** 2
```

```
I = Integral(0, 1, g)
I = Integral(a = 0, b = 1, f = ej 1)
print("Integral de x² entre [0,1]", I)
```

① * Puede ocurrir que como parámetros de entrada tengamos funciones

Nos piden implementar $\underbrace{U^{n+1}}_{\text{un solo paso}} = U^n + \overbrace{\Delta t}^{t_{n+1} - t_n} F(U^n, t_n)$ return. Sería

Falta definir U

```
def Euler(U, t1, t2, F):
    return U + [t2 - t1] * F(U, t1)
```

Hay que indicar qe es U → U vector (de numpy) → podría ser de complejos

Crear un módulo de esquemas temporales

② def Crank_Nicolson(U, t1, t2, F): $U^{n+1} = U^n + \frac{\Delta t}{2}(F^n + F^{n+1})$

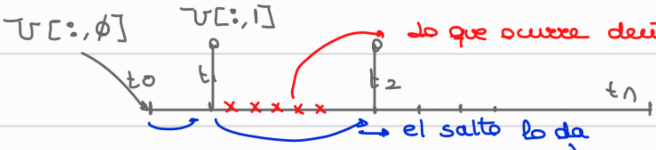
Construir residuo en funcion de F $G(U^{n+1}) = U^{n+1} - U^n - \frac{\Delta t}{2}(F^n + F^{n+1})$

```
def    Crank_Nicolson (U, t1, t2, F):

    def residual (x):

        return x - U - (t2-t1)/2 * (F(U,t1) + F(X,t2))

    return Newton ( fun = Residual, x0 = U).
```



lo que ocurre dentro se encarga los esquemas temporales Paso variable.

el salto lo da

⑤ Cauchy_Problem.py

De t1 a t2 ...

```
def Cauchy_Problem.py ( time_domain, temporal-schemes, F, U0):

    U[:, 0] = U_0.

    for n in range (N+1):

        U[:, n+1] = Temporal_scheme ( U = U[:, n), t2= , t1 = , F)
```

↳ tiene la dimension del U

Time_domain

t_{n+1} [time=n]

Time_domain vector con n+1 columnas