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

$$\underline{\overset{x}{u^{n+1}}} - \frac{\Delta t}{2} f^{n+1} = u^n + \frac{\Delta t}{2} f^n$$

$$f(x) = f(n+1)$$

$$\underline{\overset{x}{u^{n+1}}} \underbrace{- u^n - \frac{\Delta t}{2} f^n}_{} - \frac{\Delta t}{2} f^{n+1} = 0$$

U_{temp}

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

Si F es lineal $\rightarrow F = AU$ (matriz por vector)

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

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

$$U^{n+1} = \left(I - \frac{\Delta t}{2}A\right)^{-1} \cdot \left(I + \frac{\Delta t}{2}A\right)U^n$$

Llegaríamos a un Euler.

Para este caso que F es no lineal:

```
U = array ( zeros[4, N] )
U[:, 0] = array( [ ... ] )
a = U[:, n]
def G(x):

    return x - a - dt/2 . ( F(a) + F(x) )

for n in range (1, N-1):
U[:, n+1] = Newton ( func = G, x_0 = U[:, n])
```

Sería U^{n+1}

Vector columna

Sería la F Kepler evaluada en n+1.

scipy.optimize.newton

U^0 →

Temporal scheme →

F →

Time - domain →

Cauchy-Problem → $U[0:4,\ 0:N-1]$

Lo que devuelve.

Time - domain

↳ vector (partición del segmento que quiero integrar desde 0 hasta T)