Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingeniería Aeronáutica y del Espacio

Máster Universitario en Sistemas Espaciales

Milestone 3

Ampliación de Matemáticas 1

20 de octubre de 2022

Autores:

- Jaime Jiménez-Alfaro Piédrola

Índice

1. Introducción **1**

2. Código **1**

2.1. Módulo: Kepler 1

2.2. Módulo: EDO 1

2.3. Módulo: Matemáticas 3

2.4. Módulo: Esquemas numéricos 3

2.5. Módulo: Principal Hito 3 3

3. Resultados **6**

3.1. Euler 6

3.2. Runge-Kutta orden 4 8

3.3. Crank-Nicolson 10

3.4. Euler inverso 12

1. Introducción

En este informe se van a evaluar los errores de integración de una órbita de Kepler con diferentes esquemas numéricos. Para evaluar estos errores se va a utilizar la extrapolación de Richardson. Se va a explicar el código que se utiliza para ello y se van a comentar los resultados que se obtienen.

2. Código

El código que se ha utilizado para resolver este problema se ha separado en diferentes módulos.

2.1. Módulo: Kepler

En este módulo se encuentra el problema que se quiere resolver. Se quiere integrar una órbita de Kepler conociendo las condiciones iniciales.

Este módulo se muestra en el informe del Hito 2.

2.2. Módulo: EDO

En este módulo se encuentra el problema de Cauchy, el cálculo del error mediante la extrapolación de Richardson y el ratio de convergencia del esquema temporal.

El error de una solución numérica se define como la diferencia entre la solución exacta $U(t_n)$ menos la solución aproximada U^n en un instante t_n:

$$E^n = U(t_n) - U^n \tag{1}$$

La estimación del error mediante la extrapolación de Richardson es:

$$E^n = \frac{U_1^n - U_2^n}{1 - \dfrac{1}{2^q}} \tag{2}$$

Siendo q el orden del esquema numérico utilizado y U_1 y U_2 las soluciones integradas.

El ratio de convergencia del esquema numérico es:

$$log(|E^n|) = C - q \cdot log(N) \tag{3}$$

El código es:

```python
# Convergencia
def Convergence_rate(F, t, U0, E_Temporal):

    n = size(t)
    T = t[n-1]
    t1 = t

    U1 = Cauchy(F, t1, U0, E_Temporal)

    m = 7
    log_E = zeros(m)
    log_N = zeros(m)
    Error = zeros(m)
    n = 2*n

    for i in range (0,m):

        t2 = linspace(0, T, (2**i)*n)
        U2 = Cauchy(F, t2, U0, E_Temporal)

        Error[i] = norm(U2[int((2**i)*n-1),:] - U1[int((2**i)*n/2-1),:])
        log_E[i] = log10(Error[i])
        log_N[i] = log10((2**i)*n)

        U1 = U2
        print(i)

        for j in range(0,m):

         if (abs(log_E[j]) > 12):

             break

    j = min(j, m)

    # Regresión lineal y cálculo de la pendiente

    reg = LinearRegression().fit(log_N[0:j+1].reshape((-1, 1)),log_E[0:j+1])
    order = round_(abs(reg.coef_),1)

    log_N_lineal = log_N[0:j+1]
    log_E_lineal = reg.predict(log_N[0:j+1].reshape((-1, 1)))

    return [log_E, log_N, log_E_lineal, log_N_lineal, order]
```

Figura 1: Código del módulo EDO: convergencia

```python
# Error de Richardson
def Error_Richardson(F, t, U0, E_Temporal, q):

    N  = size(t)
    Error = zeros((N,size(U0)))

    T  = t[N-1]
    t1 = t
    t2 = linspace(0,T,2*N)

    U1 = Cauchy(F, t1, U0, E_Temporal)
    U2 = Cauchy(F, t2, U0, E_Temporal)

    for i in range(N):
        Error[i,:] = ( U2[2*i,:] - U1[i,:] )/( 1 - (1/(2**q)) )

    return Error

def Error_Module(Error_E_Temporal):

    Module_Error = zeros( size(Error_E_Temporal[:,0]) )

    for i in range(0,size(Error_E_Temporal[:,0])):
     Module_Error[i] = (Error_E_Temporal[i,0]**2 + Error_E_Temporal[i,1]**2)**(1/2)

    return Module_Error
# ------------
```

Figura 2: Código del módulo EDO: error de Richardson

2.3. Módulo: Matemáticas

En este módulo se encuentran algunas funciones matemáticas.

Este módulo se muestra en el informe del Hito 2.

2.4. Módulo: Esquemas numéricos

En este módulo se encuentran los cuatro esquemas numéricos que se han utilizado para la integración del problema de Kepler.

Este módulo se muestra en el informe del Hito 2.

2.5. Módulo: Principal Hito 3

Este módulo será el módulo principal, y desde aquí se llamará a los diferentes módulos y funciones y se harán las gráficas. También se podrá variar las variables temporales.

El código es:

```python
from numpy import array, zeros, linspace, size
import matplotlib.pyplot as plt

from Esquemas_Numericos import Euler, RK4, Crank_Nicolson, Inverse_Euler
from EDO import Cauchy, Error_Richardson, Error_Module, Convergence_rate
from Kepler import Kepler

#----- MODULO PRINCIPAL HITO 3 -----#


# Variables
T = 15          # Duracion
dt = 0.001      # Paso integracion

N = int(T/dt)
t  = linspace(0,T,N)
U0 = array( [1,0,0,1] )


# Errores
E_Euler          = Error_Richardson(Kepler, t, U0, Euler, 1)
E_RK4            = Error_Richardson(Kepler, t, U0, RK4, 4)
E_Crank_Nicolson = Error_Richardson(Kepler, t, U0, Crank_Nicolson, 2)
E_Inverse_Euler  = Error_Richardson(Kepler, t, U0, Inverse_Euler, 1)
```

Figura 3: Código del módulo Hito3

```python
# Graficas
E_Temporal       = [ Euler, RK4, Crank_Nicolson, Inverse_Euler ]
E_Temporal_Error = [ E_Euler, E_RK4, E_Crank_Nicolson, E_Inverse_Euler ]
E_Temporal_Plot  = ['EULER','RUNGE-KUTTA 4','CRANK-NICOLSON','EULER INVERSO']

for i in range(4):

    E_Temporal_Error_Module = Error_Module( E_Temporal_Error[i] )

    plt.title(f'ERROR {E_Temporal_Plot[i]}, dt = {dt} s')
    plt.plot(t, E_Temporal_Error[i][:,0],"r", label = "x")
    plt.plot(t, E_Temporal_Error[i][:,1],"b", label = "y")
    plt.plot(t, E_Temporal_Error_Module, 'c', label = 'Module')
    plt.xlabel("t (s)")
    plt.ylabel("Error")
    plt.legend(loc = "upper left")
    plt.grid()

    plt.savefig('Plots/' + E_Temporal_Plot[i]+ ' ' + str(dt)+'.png')
    plt.show()

    [log_E, log_N, log_E_lineal, log_N_lineal, order] = Convergence_rate(Kepler, t, U0, E_Temporal[i])

    plt.plot(log_N, log_E, "b", label = E_Temporal_Plot[i])
    plt.plot(log_N_lineal, log_E_lineal, "r", label = 'Linear regression')
    plt.legend(loc ='lower left')
    plt.xlabel("log(N)")
    plt.ylabel("log(U2-U1)")
    plt.title(f'{E_Temporal_Plot[i]}, order = {order}')
    plt.plot()
    plt.grid()
    plt.savefig('Plots/'+ E_Temporal_Plot[i]+ ' ' + str(dt)+'.png')
    plt.show()
```

Figura 4: Código del módulo Hito3

3. Resultados

Utilizando los cuatro métodos numéricos, se han realizado varias simulaciones con un tiempo de simulación de 15 s y variando el paso de integración ($\Delta t = 0{,}1$, $\Delta t = 0{,}01$ y $\Delta t = 0{,}001$).

3.1. Euler

Los resultados obtenidos con el esquema numérico de Euler son:

3.1.1. Error

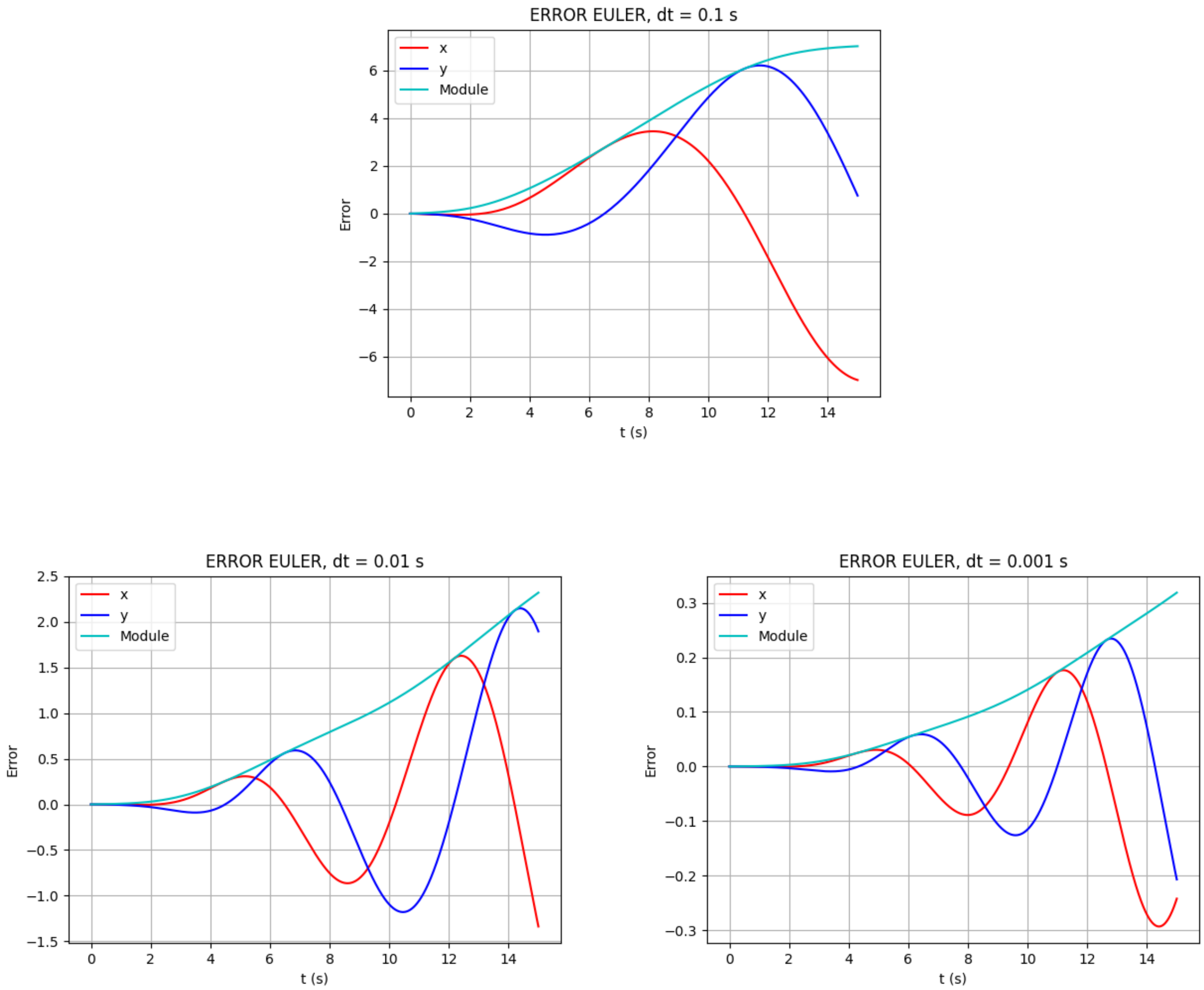


Figura 5: Resultados con Euler

En los resultados obtenidos se observa que cuanto más pequeño se hace el paso de integración Δt, la solución es más precisa.

3.1.2. Convergencia

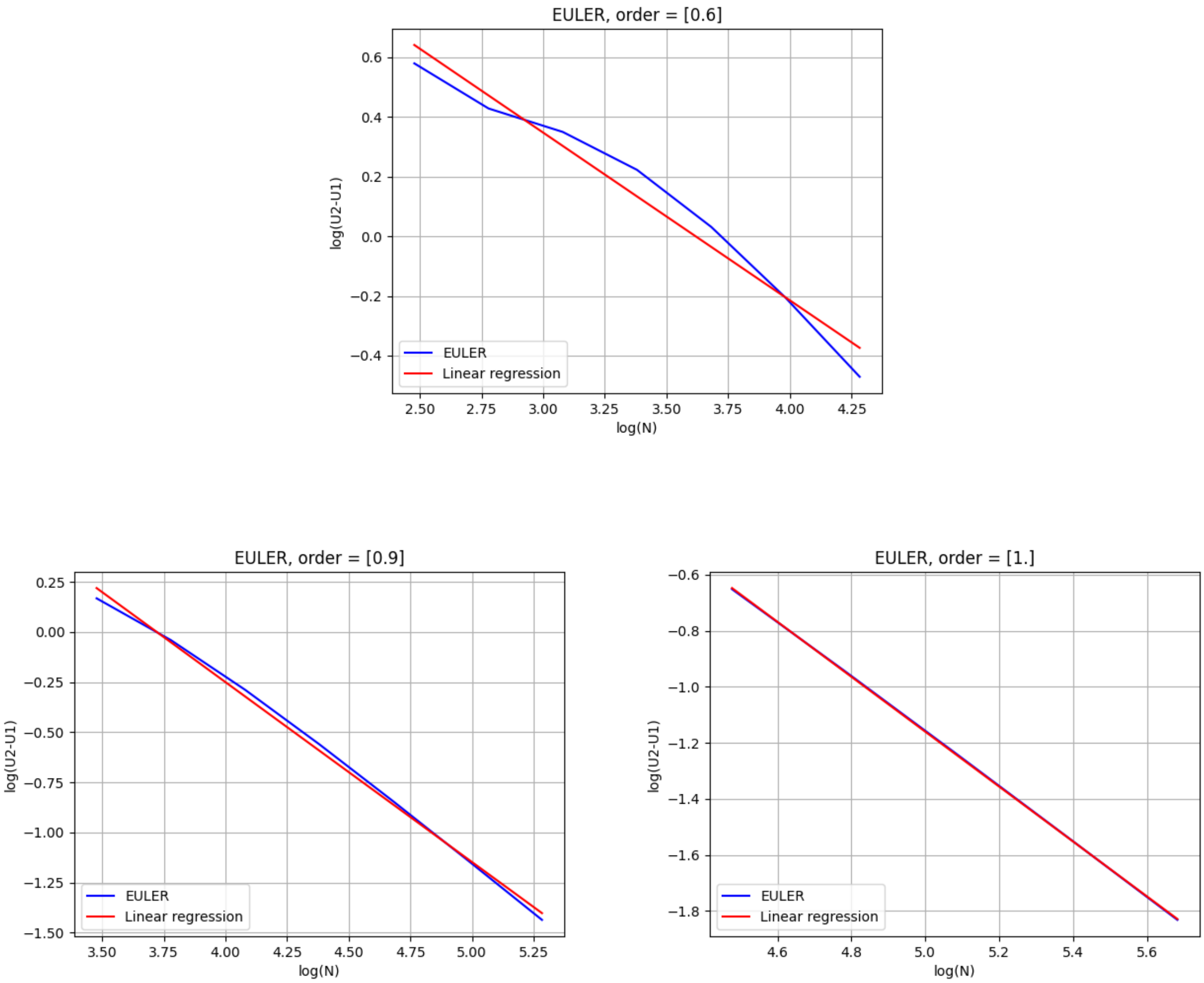


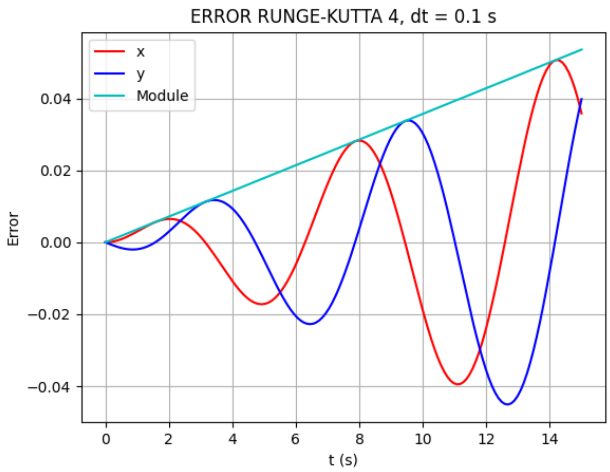
Figura 6: Resultados con Euler

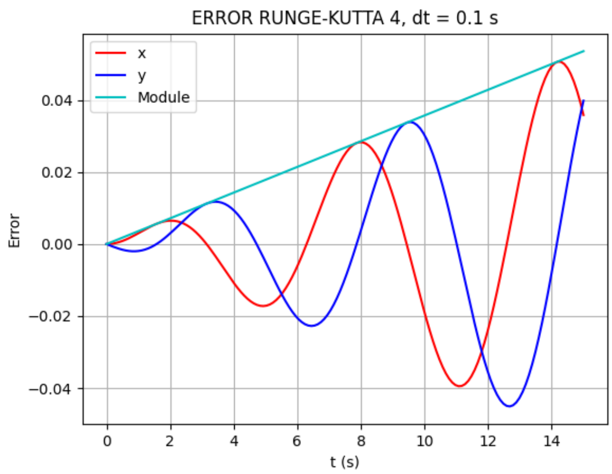
En los resultados obtenidos se observa que cuanto más pequeño se hace el paso de integración Δt, se forma una recta.

3.2. Runge-Kutta orden 4

Los resultados obtenidos con el esquema numérico de Runge-Kutta de orden 4 son:

3.2.1. Error





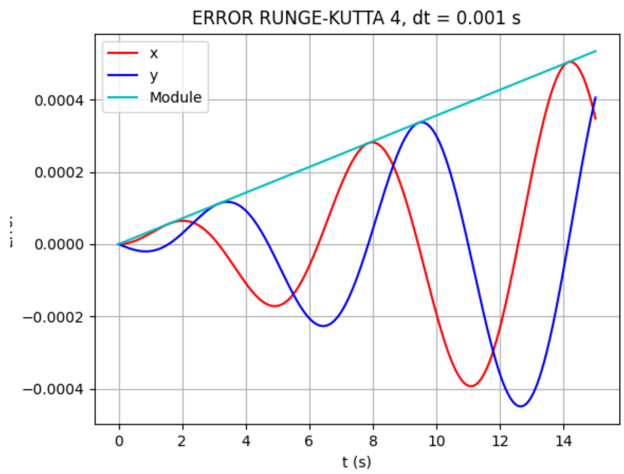
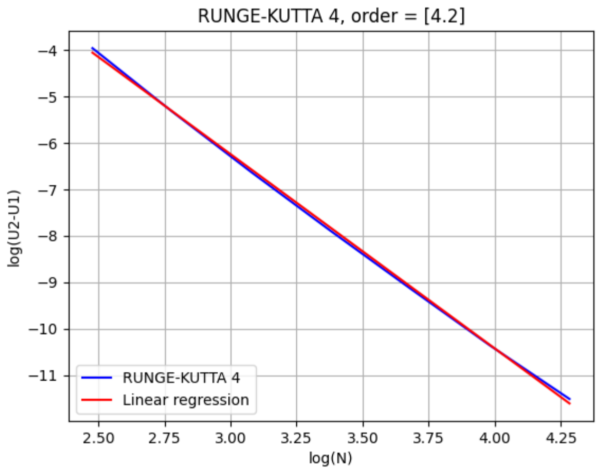


Figura 7: Resultados con Runge-Kutta orden 4

En los resultados obtenidos se observa que este esquema numérico tiene mucha precisión.

3.2.2. Convergencia



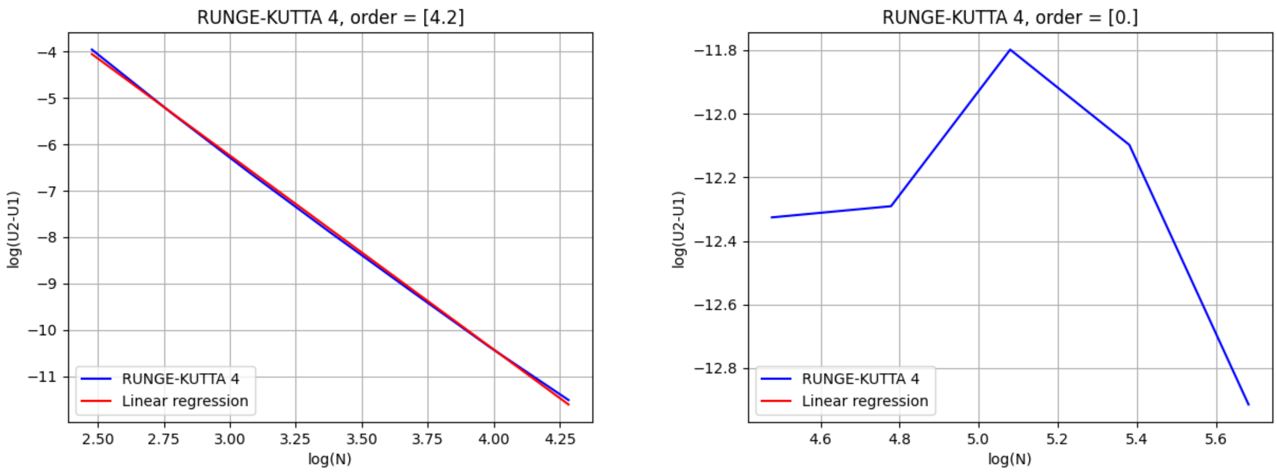


Figura 8: Resultados con Runge-Kutta orden 4

En los resultados obtenidos se observa que con este esquema numérico se obtiene el orden del esquema. Cuando se utiliza un valor de paso de integración Δt muy pequeño, debido al error de redondeo la gráfica no es correcta.

3.3. Crank-Nicolson

Los resultados obtenidos con el esquema numérico de Crank-Nicolson son:

3.3.1. Error

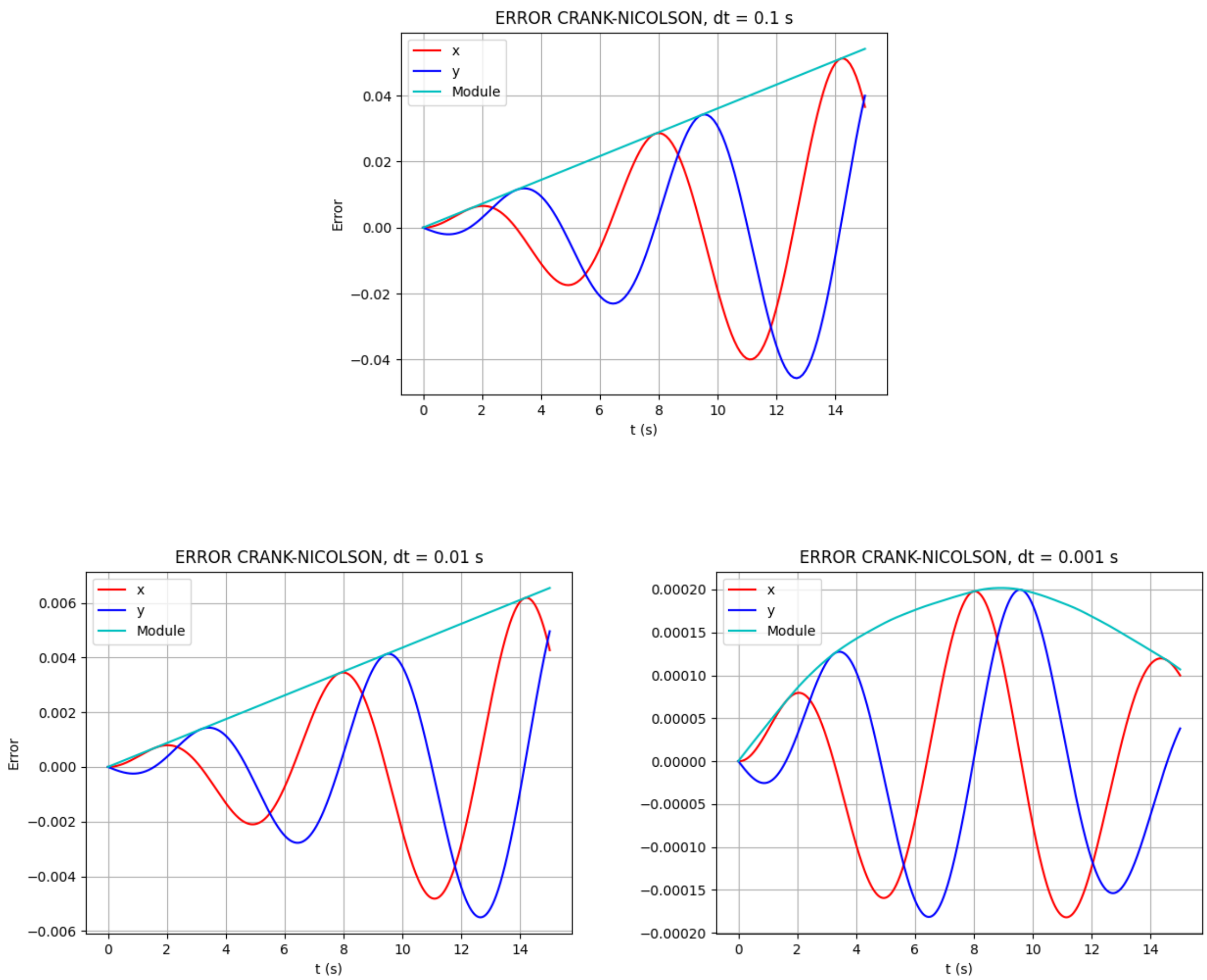


Figura 9: Resultados con Crank-Nicolson

En los resultados obtenidos se observa que este esquema numérico, al igual que el Runge-Kutta de orden 4, tiene mucha precisión.

3.3.2. Convergencia

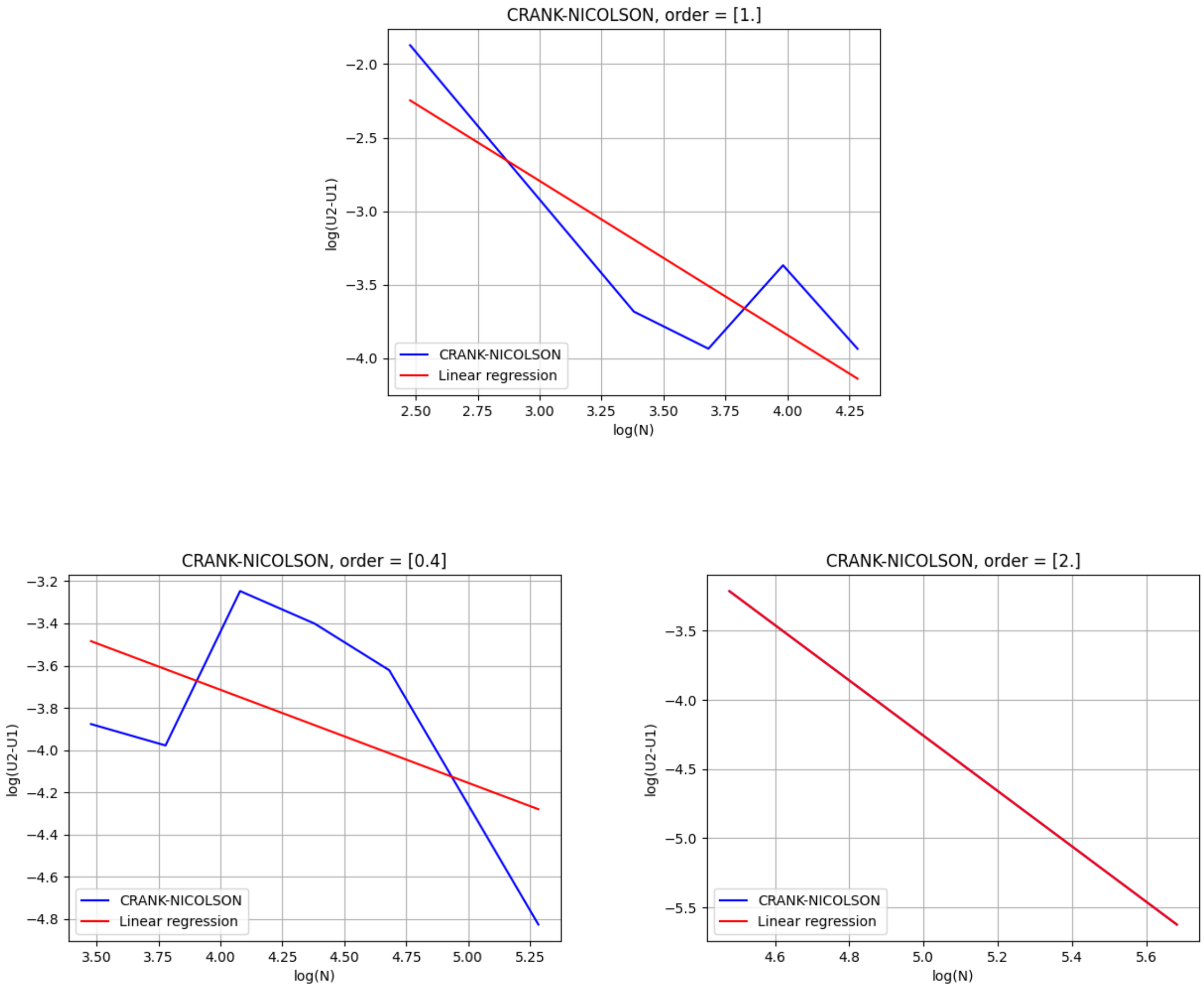


Figura 10: Resultados con Crank-Nicolson

En los resultados obtenidos se observa que este esquema numérico, para todos los valores del paso de integración Δt se obtiene el valor del orden del esquema con exactitud.

3.4. Euler inverso

Los resultados obtenidos con el esquema numérico de Euler inverso son:

3.4.1. Error

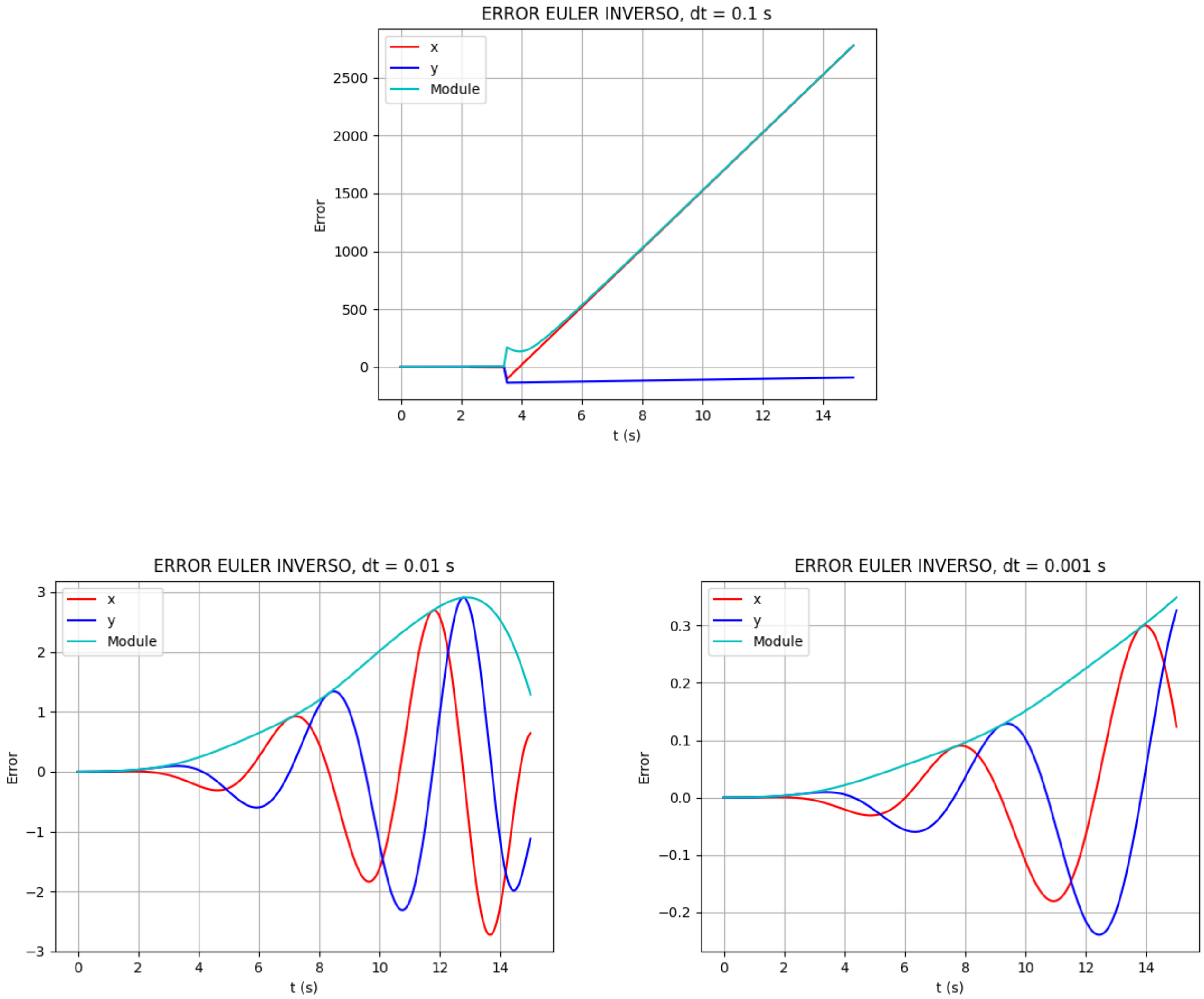


Figura 11: Resultados con Euler inverso

En los resultados obtenidos se observa que en este esquema numérico, hay que utilizar un paso de integración muy pequeño para que la solución sea precisa.

3.4.2. Convergencia

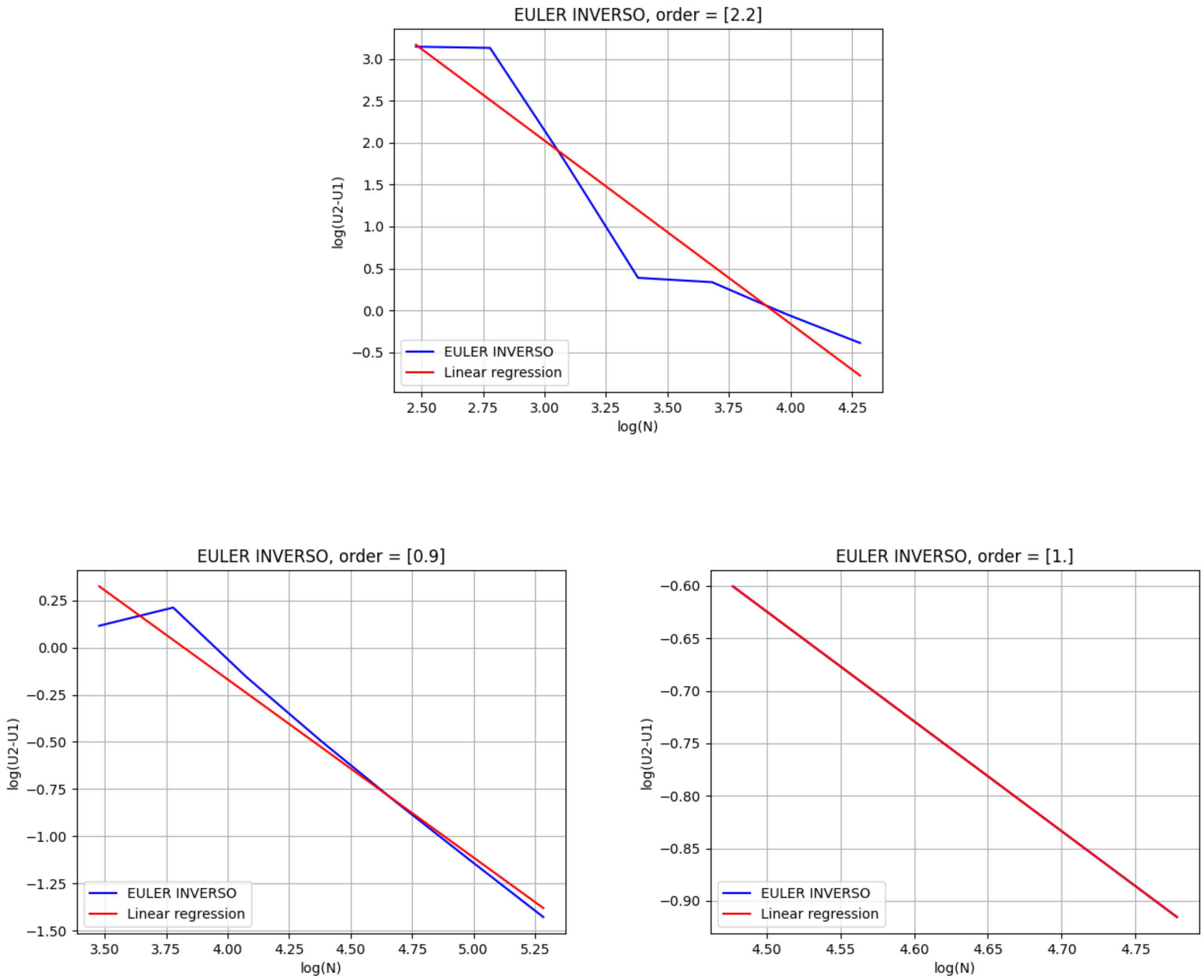


Figura 12: Resultados con Euler inverso

En los resultados obtenidos se observa que en este esquema numérico, hay que utilizar un paso de integración muy pequeño para que la convergencia sea precisa.