# Modelación Numérica II
# Tarea I

Iván Vega Gutiérrez

Centro de Investigación en Matemáticas A.C.
Unidad Aguascalientes
E-mail: ivan.vega@cimat.mx

## I. Ejercicio 1

Considere los siguientes problemas de valor inicial

A) $y' = -y, \quad y(0) = 1; \quad 0 \leq t \leq 2$

B) $y' = ty, \quad y(0) = 1; \quad 0 \leq t \leq 2$

Realice lo siguiente

1. Del teorema de existencia y unicidad da el intervalo donde existe la solución.
2. Calcular la solución analítica.
3. Utilizar los método de Euler, Taylor de orden 2 y 4, Runge-Kutta de orden 2 y 4, para la solución al problema de valor inicial.
4. En una tabla comparar la solución analítica con las soluciones de los métodos numéricos junto con sus respectivos errores para un paso de 0.25.
5. Graficar las soluciones obtenidas con los distintos métodos y la solución exacta.

### I.1. Solución de A)

Primero, verifiquemos que el problema de valor inicial tiene solución única y encontremos el rectángulo donde existe la solución.

Notemos que $f(t, y) = -y$ es continua en $\mathbb{R}^2$, por otro lado $\frac{\partial f}{\partial y} = -1$ también es continua en $\mathbb{R}^2$. Por lo tanto, por el teorema de existencia y unicidad se garantiza que existe una única solución en cualquier rectángulo $R \subset \mathbb{R}^2$,tal que $(0, 1) \in R$.

Ahora, hallemos la solución analítica del problema con valor inicial

$$\frac{dy}{dt} = -y, \quad y(0) = 1. \tag{1}$$

Para resolver el problema con valor inicial, Integramos (1) de 0 a $t$,

$$\int_0^t \frac{1}{y} dy = -\int_0^t ds.$$

Luego,

$$\ln y(t) - \ln y(0) = -t.$$

Por lo tanto la solución analítica de (1) es

$$y(t) = e^{-t}.$$

Notemos que $t \in \mathbb{R}$ y $y \in \mathbb{R}^+$. A continuación se presenta la solución numérica al problema asociado.

En la tabla (1) se compara el desempeño de los métodos numéricos: Euler, Taylor de orden 2 y 4 y Runge-Kutta de orden 2 y 4. Mientras que en la tabla (2) se muestran los respectivos errores cometidos por cada método.

| t | Analítica | Euler | Taylor orden 2 | Taylor orden 4 | Runge-Kutta orden 2 | Runge-Kutta orden 4 |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0.25 | 0.778801 | 0.75 | 0.78125 | 0.778808 | 0.78125 | 0.778808 |
| 0.5 | 0.606531 | 0.5625 | 0.610351 | 0.606543 | 0.610351 | 0.606543 |
| 0.75 | 0.472367 | 0.421875 | 0.476837 | 0.472381 | 0.476837 | 0.472382 |
| 1 | 0.367879 | 0.316406 | 0.372529 | 0.367894 | 0.372529 | 0.367894 |
| 1.25 | 0.286505 | 0.237305 | 0.291038 | 0.286519 | 0.291038 | 0.286519 |
| 1.5 | 0.22313 | 0.177978 | 0.227374 | 0.223143 | 0.227374 | 0.223143 |
| 1.75 | 0.173774 | 0.133484 | 0.177636 | 0.173786 | 0.177636 | 0.173786 |
| 2 | 0.135335 | 0.100113 | 0.138778 | 0.135346 | 0.138778 | 0.135346 |

Tabla 1: Comparación entre métodos numéricos para el problema A)

| t | Euler | Taylor orden 2 | Taylor orden 4 | Runge-Kutta orden 2 | Runge-Kutta orden 4 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0.25 | 0.028801 | 0.002449 | 0.000008 | 0.002449 | 0.000008 |
| 0.5 | 0.044031 | 0.003821 | 0.000012 | 0.003821 | 0.000012 |
| 0.75 | 0.050492 | 0.004471 | 0.000014 | 0.004471 | 0.000014 |
| 1 | 0.051473, | 0.004650 | 0.000015 | 0.004650 | 0.000015 |
| 1.25 | 0.049200 | 0.004534 | 0.000014 | 0.004534 | 0.000014 |
| 1.5 | 0.045152 | 0.004244 | 0.000013 | 0.004244 | 0.000013 |
| 1.75 | 0.040290 | 0.003862 | 0.000012 | 0.003862 | 0.000012 |
| 2 | 0.035222 | 0.003443 | 0.000011 | 0.003443 | 0.000011 |

Tabla 2: Errores de los métodos numéricos para el problema A)

De las tablas(1) y (2) se observa que el mejor método es Runge-Kutta de orden 4 y el que da una peor aproximación es el método de Euler. Para poder visualizar mejor los resultados se muestra la siguiente gráfica. A simple vista de la figura (1), solo se puede distinguir que el método de Euler es el que está alejado a los demás métodos. Sin embargo en las figura (2) se puede observar que los métodos de Taylor de orden 2 y Runge-Kutta de orden 2 prácticamente tienen el mismo desempeño, mientras que en la figura (3) los métodos de Taylor de orden 4 y Runge-Kutta de orden 4 son los que mejor aproximan a la solución exacta, incluso haciendo el mayor zoom posible no se nota diferencia entre ambos métodos.

### I.2. Solución de B)

Observemos que $f(t, y) = ty$ y $\frac{\partial f}{\partial y} = t$ son continuas en $\mathbb{R}^2$, por el teorema de existencia y unicidad, la solución al problema con valor inicial existe y es unica en cualquier rectanculo de $R \subset \mathbb{R}^2$

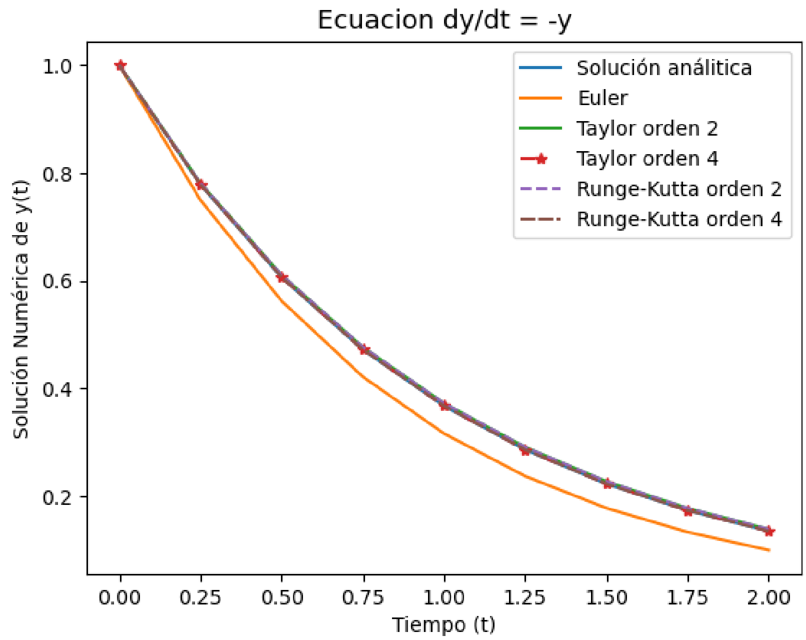


Figura 1: Solución del problema A)

que contenga al punto $(0, 1)$.

Ahora, resolvamos problema con valor inicial.

$$\frac{dy}{dt} = ty \qquad y(0) = 1 \tag{2}$$

A partir de (2) obtenemos que,

$$\int_0^t \frac{1}{y} = \int_0^t sds.$$

En consecuencia,

$$\ln y(t) - \ln y(0) = \frac{t^2}{2}.$$

Por lo tanto, la solución analítica del problema de valor inicial (2) es

$$y(t) = e^{\frac{t^2}{2}}$$

Notemos que $t \in \mathbb{R}$, $y \in [1, \infty)$.

De manera similar al problema anterior, las tablas (3) y (4) muestran que el mejor rendimiento lo obtiene el método de Runge-Kutta de orden 4 y la peor aproximación es con el método de Euler, sin embargo, a diferencia del problema previo donde los métodos de Taylor y Runge-Kutta daban resultados prácticamente iguales, estos métodos muestan diferencias signficativas. En la figura (4) se muestran las gráfica de los rendimientos de los métodos numéricos.

En la figura (4), se puede observar que el peor método es el de Euler, seguido de Taylor de orden 2 y Runge-Kutta de orden 2. En la figura(5), se observa que la mejor aproximación se obtiene mediante el método de Runge-Kutta de orden 4.

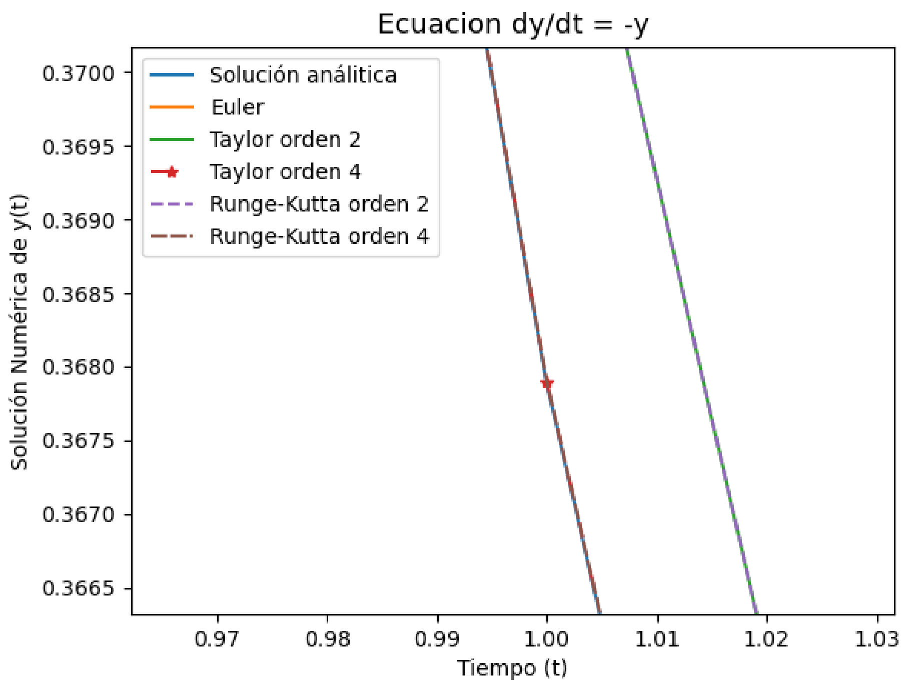


Figura 2: Taylor y Runge-Kutta de orden 2

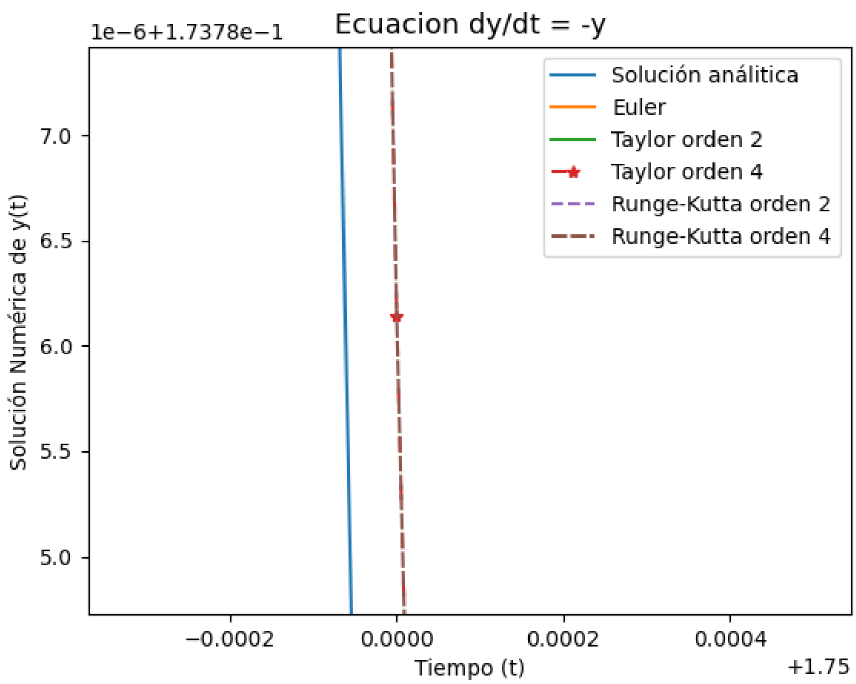


Figura 3: Taylor y Runge-Kutta de orden 4

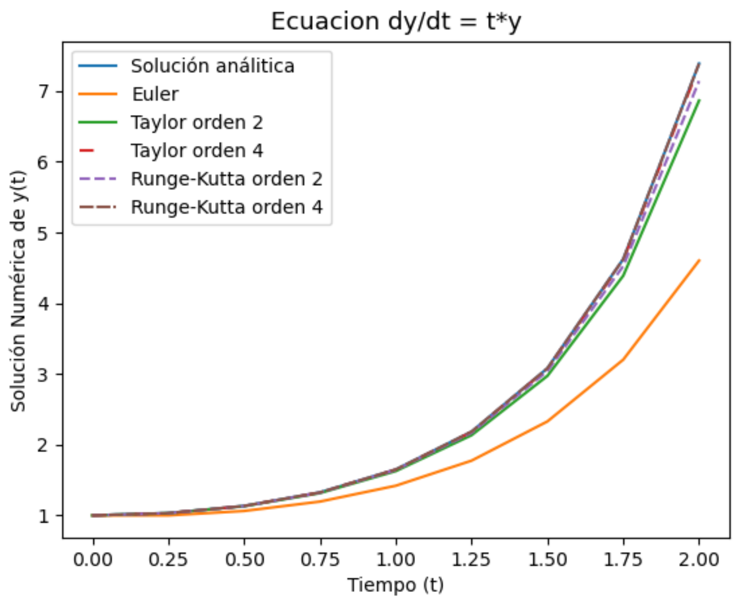


Figura 4: Solución del problema B)

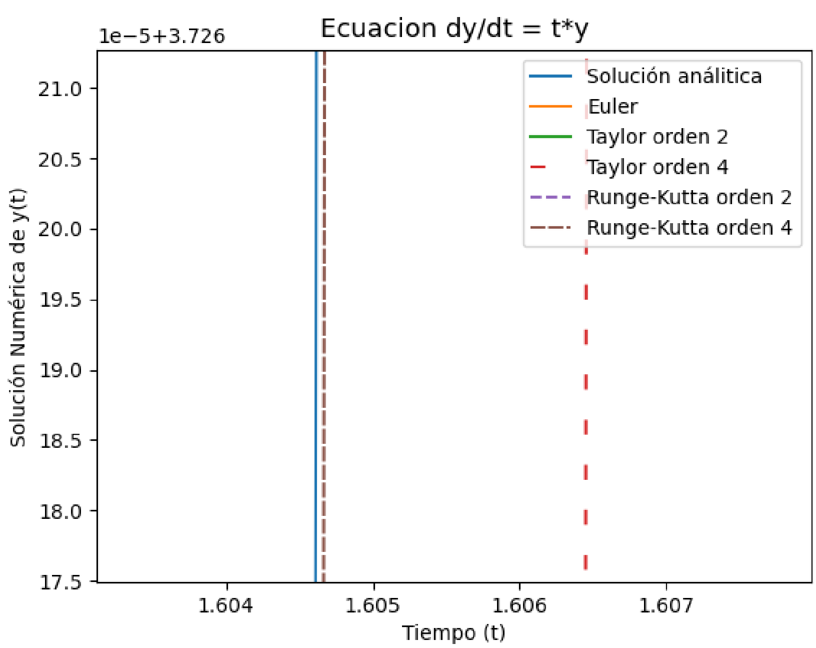


Figura 5: Taylor y Runge Kutta de orden 4

| t | Analítica | Euler | Taylor orden 2 | Taylor orden 4 | Runge-Kutta orden 2 | Runge-Kutta orden 4 |
|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0.25 | 1.031743 | 1 | 1.03125 | 1.031413 | 1.031250 | 1.031743 |
| 0.5 | 1.133148 | 1.0625 | 1.129944 | 1.132471 | 1.131958 | 1.133148 |
| 0.75 | 1.324785 | 1.195313 | 1.315325 | 1.323602 | 1.322091 | 1.324782 |
| 1 | 1.648721 | 1.419434 | 1.626174 | 1.646628 | 1.642286 | 1.648709 |
| 1.25 | 2.184201 | 1.774291 | 2.134353 | 2.180285 | 2.168330 | 2.184155 |
| 1.5 | 3.080217 | 2.328758 | 2.972253 | 3.072445 | 3.040744 | 3.080049 |
| 1.75 | 4.623953 | 3.202042 | 4.388717 | 4.607687 | 4.525483 | 4.6233759 |
| 2 | 7.389056 | 4.602936 | 6.865943 | 7.353354 | 7.141778 | 7.387147 |

Tabla 3: Comparación entre métodos numéricos para el problema B)

| t | Euler | Taylor orden 2 | Taylor orden 4 | Runge-Kutta orden 2 | Runge-Kutta orden 4 |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |
| 0.25 | 0.031743 | 0.000493 | 0.000331 | 0.000493 | 0.0 |
| 0.5 | 0.070648 | 0.003205 | 0.000678 | 0.001190 | 0.0 |
| 0.75 | 0.129472 | 0.009459 | 0.001183 | 0.002693 | 0.000003 |
| 1 | 0.229288 | 0.022548 | 0.002093 | 0.006436 | 0.000012 |
| 1.25 | 0.409909 | 0.049848 | 0.003915 | 0.015871 | 0.000046 |
| 1.5 | 0.751459 | 0.107964 | 0.007772 | 0.039472 | 0.000168 |
| 1.75 | 1.421911 | 0.235236 | 0.016266 | 0.098470 | 0.000577 |
| 2 | 2.786120 | 0.523113 | 0.035702 | 0.247278 | 0.001909 |

Tabla 4: Errores de los métodos numéricos para el problema B)

## II. Ejercicio 2

Considere el problema de valor inicial

$$y'' - 0.05y' + 0.15y = 0, \quad y(0) = 0, \;\; y'(0) = 1 \tag{3}$$

Calcular $y(1)$, $y'(1)$ utilizando el método de Euler.

### II.1. Solución

Para poder resolver el problema (3) pasemos de una ecuación de segundo orden a dos ecuaciones de primer orden, de la siguiente manera.

Sea $u(t) = y'(t)$, entonces $u' = y''$. Por lo tanto, tenemos

$$\frac{dy}{dt} = u(t), \quad y(0) = 0 \tag{4}$$

$$\frac{du}{dt} = 0.05u - 0.15y, \quad u(0) = 1 \tag{5}$$

Notemos que al resolver el problema (5) hallamos $u(t)$ y por lo tanto podemos resolver (4). Por otro lado, notemos que al resolver (4) hallamos $y(t)$, en consecuencia se puede resolver (5).

Por lo tanto, el problema inicial (3) se reduce a resolver (4) y (5) simultáneamente. Al aplicar el método de Euler con un tamaño de paso de 0.1 obtenemos

$$y(1) = 1.00454216202259, \quad y'(1) = 0.982296676607503$$

A continuación se muestran los valores obtenidos al aplicar el método de Euler.

Otra forma de resolver el problema de valor inicial es derivar (3) con respecto a $t$.

$$y''' - 0.05y'' + 0.15 = 0 \tag{6}$$

Notemos que $y''(0) = 0.05y'(0) - 0.15y(0) = 0.05$.

Sea $u(t) = y''(t)$, luego $u'(t) = y'''(t)$. Por lo tanto, obtenemos el siguiente problema de valor inicial.

$$u' = 0.05u - 0.15, \quad u(0) = 0.05 \tag{7}$$

Sustituyendo $u = y''$ en (3)

$$y' = \frac{1}{0.05}(u + 0.15y), \quad y(0) = 0 \tag{8}$$

Resolviendo (7) hallamos $u(t)$, sustituimos en (8) y obtenemos la solución $y(t)$. Aplicando el método de Euler se tiene que

$$y(1) = 1.02280264081581, \quad y'(1) = 1.05114013204079$$

En el código (1) se muestra el algoritmo para obtener la solución mediante la primera alternativa y al final del código (2) aparece el algoritmo para hallar la solución mediante la segunda alternativa.

## III. Anexos

Código 1: Problema 2

```
import numpy as np
from sympy import *

def Euler_2(a,b,y_0, y_prima_0, h, fun):
    '''
    Calcula la solucion de una ecuacion diferencial de segundo orden
    mediante el metodo de Euler
    Arugmentos:
    a : limite izquierdo del intervalo de la funcion
    b : limite derecho del intervalo de la funcion
    y_0 : valor inicial de la funcion y en el tiempo t_0
    y_prima_0 : valor inicial de la derivda de la funcion en t_o
    h : tamano de paso
    fun : ecuacion diferencial que queremos resolver, debe estar de la
        forma  y'' = by'(t) +cy(t) +d = f(t,y,y')
    Salida
    T : vector que contiene la particion de la variable independiente t
    Y : vector que contiene la solucion y en cada punto t_i en (a,b)
    U : vector que coneiten la solucion y' en cada punto t_i en (a,b)
    '''
    Y = [y_0]
    U = [y_prima_0]
    T = [a]
    i = 0
    while T[i] < b:
        Y.append(Y[i] + h*U[i])
        U.append(U[i] + h*funcion.subs({t:T[i], y_prima:U[i], y:Y[i]}))
```

```
        T.append(T[i] + h)
        i += 1
    return T, Y, U

# PARAMETROS
t = Symbol('t')
y = Symbol('y')
y_prima = Symbol('y_prima')
a = 0
b = 1
y_0 = 0
y_prima_0 = 1
h = 0.1
funcion = 0.05*y_prima - 0.15*y
# IMPRESION DE RESULTADOS
T, Y, U = Euler_2(a,b,y_0, y_prima_0, h, funcion)
N = len(T)
for i in range(N):
    print("t = {},  y = {}, y' = {}  ".format(T[i], Y[i], U[i]))
```

Código 2: Problema 1

```
import numpy as np
from sympy import *
import matplotlib.pyplot as plt
import math
from collections import OrderedDict

linestyles_dict = OrderedDict(
    [('solid',               (0, ())),
     ('loosely dotted',      (0, (1, 10))),
     ('dotted',              (0, (1, 5))),
     ('densely dotted',      (0, (1, 1))),

     ('loosely dashed',      (0, (5, 10))),
     ('dashed',              (0, (5, 5))),
     ('densely dashed',      (0, (5, 1))),

     ('loosely dashdotted',  (0, (3, 10, 1, 10))),
     ('dashdotted',          (0, (3, 5, 1, 5))),
     ('densely dashdotted',  (0, (3, 1, 1, 1))),

     ('loosely dashdotdotted', (0, (3, 10, 1, 10, 1, 10))),
     ('dashdotdotted',         (0, (3, 5, 1, 5, 1, 5))),
     ('densely dashdotdotted', (0, (3, 1, 1, 1, 1, 1)))])

def Euler(a, b, y0, h, fun):
    print("Metodo de Euler")
    Y = [y0]
    T = [a]
    i = 0
    print("t = {}, y = {}".format(T[i], Y[i]))
    while T[i] < b:
        Y.append(Y[i] + h*fun.subs({t:T[i], y:Y[i]}))
```

```
        T.append(T[i] + h)
        i += 1
        print("t = {}, y = {}".format(T[i], Y[i]))
    return  Y

def Taylor_orden_2(a, b, y0, h, fun):
    print("Metodo de Taylor orden 2")
    # Hallamos las derivadas parciales
    parcial_f_t = fun.diff(t)
    parcial_f_y = fun.diff(y)
    k11 = parcial_f_t + fun * parcial_f_y
    # Inicializamos los arreglos para t,y
    T = [a]
    Y = [y0]
    i = 0
    print("t = {}, y = {}".format(T[i], Y[i]))
    while T[i] < b:
        K1 = fun.subs({t:T[i], y:Y[i]})
        K2 = k11.subs({t:T[i], y:Y[i]})
        Y.append(Y[i] + h*K1 + ((h**2)/2)*K2)
        T.append(T[i] + h)
        i += 1
        print("t = {}, y = {}".format(T[i], Y[i]))
    return Y

def Taylor_orden_4(a, b, y0, h, fun):
    print("Metodo de Taylor orden 4")
    # Hallamos las derivadas parciales
    f_t = fun.diff(t)
    f_ty = f_t.diff(y)
    f_tt = f_t.diff(t)
    f_tty = f_tt.diff(y)
    f_ttt = f_tt.diff(t)
    f_y = fun.diff(y)
    f_yy = f_y.diff(y)
    f_yyt = f_yy.diff(t)
    f_yyy = f_yy.diff(y)
    K11 =fun
    K22 = f_t + f_y*fun
    K33 = f_tt + 2*f_ty*fun + f_yy*fun**2 + f_y*f_t + fun*f_y**2
    K44 = f_ttt + f_tt*(2*f_ty+f_y) + f_t*(2*f_yy*fun + f_y**2)
          + 3*fun*(f_tty + f_yyt*fun) + f_ty*(3*f_y*fun + f_t + 2*f_y)
          + fun*f_y**3 + f_yy*(2*f_y*fun**2 + f_t*fun + 2*f_y*fun)
          + f_yyy*fun**3
    # Inicializamos los arreglos para t,y
    T = [a]
    Y = [y0]
    i = 0
    print("t = {}, y = {}".format(T[i], Y[i]))
    while T[i] < b:
        K1 = K11.subs({t:T[i], y:Y[i]})
        K2 = K22.subs({t:T[i], y:Y[i]})
        K3 = K33.subs({t:T[i], y:Y[i]})
        K4 = K44.subs({t:T[i], y:Y[i]})
```

```
        #print(K1, K2, K3, K4)
        Y.append(Y[i] + h*K1 + ((h**2)/2)*K2 + K3*((h**3)/6)
                + K4*((h**4)/24))
        T.append(T[i] + h)
        i += 1
        print("t = {}, y = {}".format(T[i], Y[i]))
    return Y

def RK_2(a, b, y0, h, fun):
    print("Metodo Runge-Kutta de orden 2")
    T = [a]
    Y = [y0]
    print("t = {}, y = {}".format(T[0], Y[0]))
    i = 0
    while T[i] < b:
        k1 = h*fun.subs({t:T[i],y:Y[i]})
        k2 = h*fun.subs({t:T[i]+ h, y:Y[i] + k1})
        Y.append( Y[i] + 0.5*(k1+k2) )
        T.append(T[i] + h)
        i +=1
        print("t = {}, y = {}".format(T[i], Y[i]))
    return Y

def RK_4(a, b, y0, h, fun):
    print("Metodo Runge-Kutta de orden 4")
    T = [a]
    Y = [y0]
    print("t = {}, y = {}".format(T[0], Y[0]))
    i = 0
    while T[i] < b:
        k1 = h*fun.subs({t:T[i],y:Y[i]})
        k2 = h*fun.subs({t:T[i]+ 0.5*h, y:Y[i] + 0.5*k1})
        k3 = h*fun.subs({t:T[i]+ 0.5*h, y:Y[i] + 0.5*k2})
        k4 = h*fun.subs({t:T[i]+ h, y:Y[i] + k3})
        Y.append(Y[i] + (k1 + 2*k2 + 2*k3 + k4)/6)
        T.append(T[i] + h)
        i +=1
        print("t = {}, y = {}".format(T[i], Y[i]))
    return  Y


# ================Ejercicio 1 ========================
# Parametros
t = Symbol('t')
y = Symbol('y')
a = 0
b = 2
y0 = 1
h = 0.25
N = int((b-a)/h)
T = np.linspace(a, b, N+1)
funcion1 = -y
# Resultados para el inciso A)
print('Ecuacion dy/dt = {}'.format(funcion1))
```

```
sol1 = [math.exp(-t) for t in T]
print("Solucion analitica")
for i in range(len(T)):
    print("i ={}, y = {}".format(T[i], round(sol1[i],6)))
E1 = Euler(a, b, y0, h, funcion1)
T21 = Taylor_orden_2(a, b, y0, h, funcion1)
T41 = Taylor_orden_4(a, b, y0, h, funcion1)
RK21 = RK_2(a, b, y0, h, funcion1)
RK41 = RK_4(a, b, y0, h, funcion1)

print('Errores')
for i in range(len(T)):
    print("t = {}, Euler = {}, T2 = {}, T4 ={}, RK2={}, RK4={}".
    format(T[i],
    round(sol1[i]-E1[i],6),round(sol1[i]- T21[i],6),
    round(sol1[i]- T41[i],6),round(sol1[i]- RK21[i],6),
    round(sol1[i]- RK41[i],6)))
plot1 = plt.figure(1)
plt.plot(T, sol1, label = 'Solución análitica')
plt.plot(T, E1, label = 'Euler', )
plt.plot(T, T21, label = 'Taylor orden 2')
plt.plot(T, T41, label = 'Taylor orden 4',
        marker = '*', linestyle = linestyles_dict['loosely dashed'])
plt.plot(T, RK21, label = 'Runge-Kutta orden 2', linestyle = 'dashed')
plt.plot(T, RK41, label = 'Runge-Kutta orden 4',
        linestyle = linestyles_dict['densely dashed'])
plt.xlabel("Tiempo (t)")
plt.ylabel("Solución Numérica de y(t)")
plt.title('Ecuacion dy/dt = {}'.format(funcion1), fontsize=13)
plt.legend()
# Resultados para el inciso B)
funcion2 = t*y
print('\n Ecuacion dy/dt = {}'.format(funcion2))
sol2 = [math.exp(0.5*t**2) for t in T]
print("Solucion analitica")
for i in range(len(T)):
    print("i={}, y = {}".format(T[i], round(sol2[i], 6)))
E2 = Euler(a, b, y0, h, funcion2)
T22 = Taylor_orden_2(a, b, y0, h, funcion2)
T42 = Taylor_orden_4(a, b, y0, h, funcion2)
RK22 = RK_2(a, b, y0, h, funcion2)
RK42 = RK_4(a, b, y0, h, funcion2)
print('Errores')
for i in range(len(T)):
    print("t = {}, Euler = {}, T2 = {}, T4 ={}, RK2={}, RK4={}"
    .format(T[i],round(sol2[i]-E2[i],6),
    round(sol2[i]- T22[i],6),round(sol2[i]- T42[i],6),
    round(sol2[i]- RK22[i],6),round(sol2[i]- RK42[i],6)))
plot2 = plt.figure(2)
plt.plot(T, sol2, label = 'Solución análitica')
plt.plot(T, E2, label = 'Euler')
plt.plot(T, T22, label = 'Taylor orden 2')
plt.plot(T, T42, label = 'Taylor orden 4',
        linestyle = linestyles_dict['loosely dashed'])
```

```
plt.plot(T, RK22, label = 'Runge-Kutta orden 2', linestyle = 'dashed')
plt.plot(T, RK42, label = 'Runge-Kutta orden 4',
        linestyle = linestyles_dict['densely dashed'])
plt.xlabel("Tiempo (t)")
plt.ylabel("Solución Numérica de y(t)")
plt.title('Ecuacion dy/dt = {}'.format(funcion2), fontsize=13)
plt.legend()
plt.show()
#==================EJERCICIO 2 ALTERNATIVA=====================
'''
# Parametros
t = Symbol('t')
y = Symbol('y')
a = 0
b = 1
y_0 = 0
h = 0.1
fun1 = 0.05*y - 0.15
print("Solucion  y''  ")
U = Euler(a, b, 0.05, h, fun1)

def fun(u, y):
    f = (u + 0.15*y)/0.05
    return f

def Euler2(a, b, y0, U, h):
    Y = [y0]
    T = [a]
    i = 0
    while T[i] < b:
        Y.append(Y[i] + h*fun(U[i], Y[i]))
        T.append(T[i] + h)
        i += 1
    return  T,Y
#Resultados
T, Y = Euler2(a,b, y_0,U, h)
print('\n Resultados')
for i in range(len(T)):
    print("t = {}, y = {},  y' ={}".format(T[i], Y[i],
    (U[i] + 0.15*Y[i])/0.05))
'''
```