ESCUELA
POLITÉCNICA
NACIONAL

Métodos numéricos.

Nombre: Luis Enrique Pérez Señalin.

CONJUNTO DE EJERCICIOS

1. Use el método de Euler para aproximar las soluciones para cada uno de los siguientes problemas de valor inicial.

a. $y' = te^{3t} - 2y, 0 \leq t \leq 1, y(0) = 0,$ con $h = 0.5$
b. $y' = 1 + (t-y)^2, 2 \leq t \leq 3, y(2) = 1,$ con $h = 0.5$
c. $y' = 1 + {}^{y}/_{t}, 1 \leq t \leq 2, y(1) = 2,$ con $h = 0.25$
d. $y' = \cos 2t + \operatorname{sen} 3t, 0 \leq t \leq 1, y(0) = 1,$ con $h = 0.25$

Respuesta:
A: 1.12
B: 2.625
C: 5.269
D: 2.236

2. Las soluciones reales para los problemas de valor inicial en el ejercicio 1 se proporcionan aquí. Compare el error real en cada paso.

a. $y(t) = \frac{1}{5}te^{3t} - \frac{1}{25}e^{3t} + \frac{1}{25}e^{-2t}$
b. $y(t) = t + \frac{1}{1-t}$
c. $y(t) = t\ln t + 2t$
d. $y(t) = \frac{1}{2}\operatorname{sen} 2t - \frac{1}{3}\cos 3t + \frac{4}{3}$

Respuesta:
A: real: 3.219 pred: 1.12
B: real 2.5 pred: 2.625
C: real: 5.386 pred: 5.269
D: real: 2.117 pred: 2.236

Pasos:
A)

Real:	[0.0, 0.283, 3.219],
Aproximado	[0, 0.0 , 1.12],
Error	[0.0, 0.283, 2.0986]

B)

Real:	[1.0, 1.833, 2.5],
Aproximado	[1, 2.0, 2.625],
Error	[0.0, 0.166, 0.125]

C)

Real:	[2.0, 2.7789, 3.608, 4.4793, 5.386],
Aproximado	[2, 2.75, 3.55, 4.3916, 5.269],
Error	[0.0, 0.0289, 0.05819, 0.0876, 0.117]

D)

Real:	[1.0, 1.329, 1.730, 2.04147, 2.1179],
Aproximado	[1, 1.25, 1.6398, 2.024, 2.236],
Error	[0.0, 0.0791, 0.090, 0.017, 0.118]

3. Utilice el método de Euler para aproximar las soluciones para cada uno de los siguientes problemas de valor inicial.

a. $y' = {}^{y}/_{t} - ({}^{y}/_{t})^2, 1 \leq t \leq 2, y(1) = 1,$ con $h = 0.1$
b. $y' = 1 + {}^{y}/_{t} + ({}^{y}/_{t})^2, 1 \leq t \leq 3, y(1) = 0,$ con $h = 0.2$
c. $y' = -(y+1)(y+3), 0 \leq t \leq 2, y(0) = -2,$ con $h = 0.2$
d. $y' = -5y + 5t^2 + 2t, 0 \leq t \leq 1, y(0) = \frac{1}{3},$ con $h = 0.1$

Respuesta:
A: 1.17
B: 4.514
C: -1.018
D: 0.979

4. Aquí se dan las soluciones reales para los problemas de valor inicial en el ejercicio 3. Calcule el error real en las aproximaciones del ejercicio 3.

a. $y(t) = \frac{t}{1+\ln t}$

b. $y(t) = t\tan(\ln t)$

c. $y(t) = -3 + \frac{2}{1+e^{-2t}}$

d. $y(t) = t^2 + \frac{1}{3}e^{-5t}$

Respuesta:
A: real: 1.17 pred: 1.181
B: real 4.514 pred: 5.874
C: real: -1.018 pred: -1.035
D: real: 0.979 pred: 1.002

Pasos:
A)

Real:	[1.0, 1.004, 1.0149, 1.0298, 1.0475, 1.06, 1.088, 1.11, 1.1336, 1.157, 1.181]
Aproximado	[1, 1.0, 1.008, 1.0216, 1.038, 1.057, 1.078, 1.1004, 1.123, 1.1467, 1.17]
Error	[0.0, 0.004, 0.006, 0.008, 0.009, 0.009, 0.0099, 0.01, 0.01, 0.01, 0.01]

B)

Real:	[0.0, 0.221, 0.489, 0.81, 1.199, 1.66, 2.21, 2.875, 3.678, 4.658, 5.874]
Aproximado	[0, 0.2, 0.438, 0.721, 1.052, 1.437, 1.884, 2.40, 3.00, 3.70, 4.51],
Error	[0.0, 0.021, 0.05, 0.091, 0.147, 0.224, 0.329, 0.47, 0.67, 0.958, 1.359]

C)

Real:	[-2.0, -1.80, -1.62, -1.46, -1.3359, -1.238, -1.166, -1.1146, -1.078, -1.05, -1.0359]
Aproximado	[-2, -1.8, -1.608, -1.438, -1.301, -1.199, -1.127, -1.079, -1.049, -1.029, -1.018]
Error	[0.0, 0.0026, 0.012, 0.024, 0.034, 0.039, 0.0388, 0.0349, 0.029, 0.023, 0.0178]

D)

Real:	[0.333, 0.212, 0.162, 0.164, 0.205, 0.27, 0.376, 0.50, 0.64, 0.8137, 1.0],
Aproximado	[-0.5, -0.25, -0.099, 0.01, 0.110, 0.215, 0.33, 0.46, 0.618, 0.789, 0.979],
Error	[0.833, 0.462, 0.262, 0.154, 0.095, 0.0623, 0.044, 0.0338, 0.0279, 0.0246, 0.0227]

5. Utilice los resultados del ejercicio 3 y la interpolación lineal para aproximar los siguientes valores de $y(t)$. Compare las aproximaciones asignadas para los valores reales obtenidos mediante las funciones determinadas en el ejercicio 4.

a. $y(0.25)$ y $y(0.93)$

b. $y(t) = y(1.25)$ y $y(1.93)$

c. $y(2.10)$ y $y(2.75)$

d. $y(t) = y(0.54)$ y $y(0.94)$

Respuesta:
A: No se puede evaluar porque su intervalo va de 1 a 2.

B:
t = 1.25, Error: 0.023930903973006623
t = 1.93, Error: 0.18780121684809004

C: No se puede evaluar porque su intervalo es de 0 a 2

D:
t = 0.54, Error: 0.052001837579916554
t = 0.94, Error: 0.021381759033898495

6. Use el método de Taylor de orden 2 para aproximar las soluciones para cada uno de los siguientes problemas de valor inicial.

a. $y' = te^{3t} - 2y, 0 \leq t \leq 1, y(0) = 0,$ con $h = 0.5$

b. $y' = 1 + (t - y)^2, 2 \leq t \leq 3, y(2) = 1,$ con $h = 0.5$

c. $y' = 1 + {}^{y}/_{t}, 1 \leq t \leq 2, y(1) = 2,$ con $h = 0.25$

d. $y' = \cos 2t + \operatorname{sen} 3t, 0 \leq t \leq 1, y(0) = 1,$ con $h = 0.25$

Respuesta:
A: 2.0232
B: 2.425
C: 5.344
D: 2.20

7. Repita el ejercicio 6 con el método de Taylor de orden 4

Respuesta:
A: 4.2635
B: 2.4832
C: 5.273
D: 1.833

Métodos Numéricos Deber 12

```
%load_ext autoreload
```

```
%autoreload 2
from src import ODE_euler, ODE_euler_nth
from math import exp,cos, sin, log, tan
import numpy as np
```

Pregunta 1

```
# Literal A)
f = lambda t, y: t * exp(3 * t) - 2 * y
y_t0 = 0

a = 0
b = 1

h = 0.5
N = int((b - a) / h)
ys_a,ts_a, h = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_a[len(ys_a)-1]
```

```
1.1204222675845161
```

```
# Literal B)
f = lambda t, y: 1 + (t - y)**2
y_t0 = 1

a = 2
b = 3

h = 0.5
N = int((b - a) / h)
ys_b,ts_b, h_b = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_b[len(ys_b)-1]
```

```
2.625
```

```
# Literal C)
f = lambda t, y: 1 + y / t

y_t0 = 2
a = 1
b = 2

h = 0.25
N = int((b - a) / h)
```

```python
ys_c,ts_c, h_c = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_c[len(ys_c)-1]
```

```
5.269047619047619
```

```python
#Literal D)
f = lambda t, y: cos(2 * t) + sin(3 * t)

y_t0 = 1
a = 0
b = 1

h = 0.25
N = int((b - a) / h)
ys_d,ts_d, h_d = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_d[len(ys_d)-1]
```

```
2.2364572532353817
```

Pregunta 2

```python
def error(real, pred):
    return abs(real-pred)
```

```python
# Solución real
def y_real_a(t):
    return (1/5) * t * exp(3*t) - (1/25) * exp(3*t) + (1/25) * exp(-2*t)

yr_a = [y_real_a(ts_ai) for ts_ai in ts_a ]
error_a = [error(y_real, y_pred) for y_real, y_pred in zip(yr_a, ys_a)]
print(f"Real:\t\t{yr_a},\nAproximado\t{ys_a},\nError\t\t{error_a}")
```

```
Real:       [0.0, 0.2836165218671416, 3.2190993190394916],
Aproximado  [0, 0.0, 1.1204222675845161],
Error       [0.0, 0.2836165218671416, 2.0986770514549757]
```

```python
# Solución real B
def y_real_b(t):
    return t + 1 / (1 - t)

yr_b = [y_real_b(ts_bi) for ts_bi in ts_b ]
error_b = [error(y_real, y_pred) for y_real, y_pred in zip(yr_b, ys_b)]
print(f"Real:\t\t{yr_b},\nAproximado\t{ys_b},\nError\t\t{error_b}")
```

```
Real:       [1.0, 1.8333333333333335, 2.5],
Aproximado  [1, 2.0, 2.625],
Error       [0.0, 0.16666666666666652, 0.125]
```

```python
        # Solución real C
        def y_real_c(t):
            return t * log(t) + 2 * t

        yr_c = [y_real_c(ts_ci) for ts_ci in ts_c ]
        error_c = [error(y_real, y_pred) for y_real, y_pred in zip(yr_c, ys_c)]
        print(f"Real:\t\t{yr_c},\nAproximado\t{ys_c},\nError\t\t{error_c}")
```

```
Real:       [2.0, 2.7789294391427624, 3.6081976621622465, 4.47932762888699,
5.386294361119891],
Aproximado  [2, 2.75, 3.55, 4.391666666666667, 5.269047619047619],
Error       [0.0, 0.02892943914276236, 0.058197662162246644, 0.08766096222032349,
0.11724674207227181]
```

```python
        # Soludión real D
        def y_real_d(t):
            return (1/2) * sin(2*t) - (1/3) * cos(3*t) + 4/3

        yr_d = [y_real_d(ts_di) for ts_di in ts_d ]
        error_d = [error(y_real, y_pred) for y_real, y_pred in zip(yr_d, ys_d)]
        print(f"Real:\t\t{yr_d},\nAproximado\t{ys_d},\nError\t\t{error_d}")
```

```
Real:       [1.0, 1.3291498130108277, 1.7304897585147139, 2.041472034209607,
2.1179795456129895],
Aproximado  [1, 1.25, 1.6398053304784268, 2.0242546535964756, 2.2364572532353817],
Error       [0.0, 0.07914981301082769, 0.09068442803628707, 0.017217380613131272,
0.11847770762239218]
```

Pregunta 3

```python
        # Literal A)
        f = lambda t, y: (y/t) - (y/t)**2
        y_t0 = 1

        a = 1
        b = 2

        h = 0.1
        N = int((b - a) / h)
        ys_a,ts_a, h = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
        ys_a[len(ys_a)-1]
```

```
1.1706515695646647
```

```python
        # Literal B)
        f = lambda t, y: 1 + (y/t) + (y/t)**2
        y_t0 = 0

        a = 1
        b = 3
```

```
h = 0.2
N = int((b - a) / h)
ys_b,ts_b, h = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_b[len(ys_a)-1]
```

```
4.5142774281767
```

```
# Literal C)
f = lambda t, y: -(y + 1) * (y + 3)
y_t0 = -2

a = 0
b = 2

h = 0.2
N = int((b - a) / h)
ys_c,ts_c, h = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_c[len(ys_a)-1]
```

```
-1.0181518381465764
```

```
# Literal D)
f = lambda t, y: -5 * y + 5 * t**2 + 2 * t
y_t0 = -0.5

a = 0
b = 1

h = 0.1
N = int((b - a) / h)
ys_d,ts_d, h = ODE_euler(a=a, b=b, y_t0=y_t0, f=f, N=N)
ys_d[len(ys_a)-1]
```

```
0.9795312499999999
```

Pregunta 4

```
# Solución real A
def y_real_a(t):
    return t / (1 + log(t))

yr_a = [y_real_a(ts_ai) for ts_ai in ts_a ]
error_a = [error(y_real, y_pred) for y_real, y_pred in zip(yr_a, ys_a)]
print(f"Real:\t\t{yr_a},\nAproximado\t{ys_a},\nError\t\t{error_a}")
```

```
Real:       [1.0, 1.0042817279362024, 1.0149523140337415, 1.0298136889579848,
1.0475339192525197, 1.067262354181873, 1.088432686945791, 1.1106550521462644,
```

```
1.1336535567333055, 1.1572284330546696, 1.1812322182992827],
Aproximado  [1, 1.0, 1.0082644628099173, 1.0216894717270375, 1.038514734248178,
1.0576681921408762, 1.0784610936317547, 1.100432164699466, 1.1232620515812632,
1.1467235965295264, 1.1706515695646647],
Error       [0.0, 0.004281727936202406, 0.006687851223824204, 0.00812421723094725,
0.009019185004341734, 0.009594162040996945, 0.009971593314036298, 0.010222887446798445,
0.010391505152042235, 0.010504836525143224, 0.010580648734618059]
```

```python
# Solución real B
def y_real_b(t):
    return t * np.tan(np.log(t))

ts_b = np.array(ts_b)
yr_b = y_real_b(ts_b)
yr_b = [float (yr_bi) for yr_bi in yr_b]
error_b = [error(float (y_real), y_pred) for y_real, y_pred in zip(yr_b, ys_b)]
print(f"Real:\t\t{yr_b},\nAproximado\t{ys_b},\nError\t\t{error_b}")
```

```
Real:       [0.0, 0.22124277275763113, 0.48968166375094263, 0.812752740561542,
1.19943864032594, 1.661281755721567, 2.213501813480633, 2.8765514199948425,
3.6784753308518447, 4.658665058239517, 5.874099978184171],
Aproximado  [0, 0.2, 0.4388888888888889, 0.721242756361804, 1.0520380316573712,
1.4372511475238394, 1.8842608053291532, 2.402269588561542, 3.0028371645572136,
3.7006007049327985, 4.5142774281767],
Error       [0.0, 0.021242772757631118, 0.05079277486205375, 0.09150998419973799,
0.14740060866856886, 0.2240306081977277, 0.32924100815147983, 0.4742818314333004,
0.6756381662946311, 0.9580643533067188, 1.359822550007471]
```

```python
# Solución real C
def y_real_c(t):
    return -3 + 2 / (1 + exp(-2*t))

yr_c = [y_real_c(ts_ci) for ts_ci in ts_c ]
error_c = [error(y_real, y_pred) for y_real, y_pred in zip(yr_c, ys_c)]
print(f"Real:\t\t{yr_c},\nAproximado\t{ys_c},\nError\t\t{error_c}")
```

```
Real:       [-2.0, -1.802624679775096, -1.620051037744775, -1.4629504330019645,
-1.335963229732151, -1.2384058440442354, -1.1663453929878447, -1.1146483517977375,
-1.0783314455935287, -1.053193987153732, -1.035972419924183],
Aproximado  [-2, -1.8, -1.608, -1.4387328000000001, -1.3017369739591682, -1.199251224666308,
-1.1274909449059896, -1.079745355150198, -1.0491190774237251, -1.0299539832076265,
-1.0181518381465764],
Error       [0.0, 0.0026246797750959505, 0.012051037744774895, 0.024217633001964334,
0.03422625577298288, 0.0391546193779273, 0.03885444808185512, 0.034902996647539375,
0.02921236816980355, 0.02324000394610537, 0.017820581777606703]
```

```python
# Soludión real D
def y_real_d(t):
    return t**2 + (1/3) * exp(-5*t)

yr_d = [y_real_d(ts_di) for ts_di in ts_d ]
```

```
error_d = [error(y_real, y_pred) for y_real, y_pred in zip(yr_d, ys_d)]
print(f"Real:\t\t{yr_d},\nAproximado\t{ys_d},\nError\t\t{error_d}")
```

```
Real:       [0.3333333333333333, 0.2121768865708778, 0.16262648039048078, 0.16437672004947662,
0.2051117610788709, 0.27736166620796626, 0.3765956894559546, 0.5000657944741062,
0.6461052129629113, 0.8137029988460805, 1.0022459823330283],
Aproximado  [-0.5, -0.25, -0.09999999999999998, 0.010000000000000037, 0.11000000000000004,
0.21500000000000002, 0.3325, 0.46625, 0.618125, 0.7890625, 0.9795312499999999],
Error       [0.8333333333333333, 0.4621768865708778, 0.26262648039048075, 0.15437672004947658,
0.09511176107887087, 0.06236166620796624, 0.0440956894559546, 0.03381579447410615,
0.02798021296291131, 0.02464049884608055, 0.022714732333028453]
```

Pregunta 5

```
# Encuentra el intervalo para t_approx1 y t_approx2
def linear_interpolate(t, t0, y0, t1, y1):
    return y0 + ((y1 - y0) / (t1 - t0)) * (t - t0)

def interpolate_values(ts, ys, t_values, y_real):
    """
    Realiza la interpolación lineal para una lista de valores t_values dados una list

    Parámetros:
    - ts: Lista de puntos de malla.
    - ys: Lista de valores aproximados correspondientes a ts.
    - t_values: Lista de valores t para los cuales se desea calcular la interpolación

    Retorna:
    - Lista de valores interpolados para cada valor en t_values.
    """

    interpolated_values = []

    for t_approx in t_values:
        for i in range(len(ts) - 1):
            if ts[i] <= t_approx <= ts[i + 1]:
                approx = linear_interpolate(t_approx, ts[i], ys[i], ts[i + 1], ys[i +
                exact = y_real(t_approx)
                error = abs(approx - exact)
                interpolated_values.append((t_approx, approx, exact, error))
                break

    return interpolated_values
```

```
# Literal A

# Puntos para interpolar
t_approx1 = 0.25
t_approx2 = 0.93
```

```python
interpolated_ys_a = interpolate_values(ts_a, ys_a, [t_approx1,t_approx2],y_real_a)

# Mostrar resultados
for t, approx, exact, error_v in interpolated_ys_a:
    print(f"t = {t}, Error: {error_v}")
```

```python
ts_a
```

```
[1,
 1.1,
 1.2000000000000002,
 1.3000000000000003,
 1.4000000000000004,
 1.5000000000000004,
 1.6000000000000005,
 1.7000000000000006,
 1.8000000000000007,
 1.9000000000000008,
 2.000000000000001]
```

```python
# Literal B

# Puntos para interpolar
t_approx1 = 1.25
t_approx2 = 1.93

interpolated_ys_b = interpolate_values(ts_b, ys_b, [t_approx1,t_approx2], y_real_b)

# Mostrar resultados
for t, approx, exact, error_v in interpolated_ys_b:
    print(f"t = {t}, Error: {error_v}")
```

```
t = 1.25, Error: 0.023930903973006623
t = 1.93, Error: 0.18780121684809004
```

```python
ts_c
```

```
[0,
 0.2,
 0.4,
 0.6000000000000001,
 0.8,
 1.0,
 1.2,
 1.4,
 1.5999999999999999,
 1.7999999999999998,
 1.9999999999999998]
```

```
# Literal C

# Puntos para interpolar
t_approx1 = 2.10
t_approx2 = 2.75

interpolated_ys_c = interpolate_values(ts_c, ys_c, [t_approx1,t_approx2], y_real_c)

# Mostrar resultados
for t, approx, exact, error_v in interpolated_ys_c:
    print(f"t = {t}, Error: {error_v}")
```

```
# Literal D

# Puntos para interpolar
t_approx1 = 0.54
t_approx2 = 0.94

interpolated_ys_d = interpolate_values(ts_d, ys_d, [t_approx1,t_approx2], y_real_d)

# Mostrar resultados
for t, approx, exact, error_v in interpolated_ys_d:
    print(f"t = {t}, Error: {error_v}")
```

```
t = 0.54, Error: 0.052001837579916554
t = 0.94, Error: 0.021381759033898495
```

Pregunta 6

```
# Literal A)
f = lambda t, y: t * exp(3 * t) - 2 * y
f_p = lambda t, y: exp(3 * t) * (3 * t + 1) - 2 * f(t, y)
y_t0 = 0

a = 0
b = 1

h = 0.5
N = int((b - a) / h)
ys_a,ts_a, h = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p])
ys_a[len(ys_a)-1]
```

```
2.0232389682729033
```

```
# Literal B)
f = lambda t, y: 1 + (t - y)**2
f_p = lambda t, y: 2 * (t - y) * (1 - f(t, y))
y_t0 = 1

a = 2
```

```
b = 3

h = 0.5
N = int((b - a) / h)
ys_b,ts_b, h_b = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p])
ys_b[len(ys_b)-1]
```

```
2.42578125
```

```
# Literal C)
f = lambda t, y: 1 + y / t
f_p = lambda t, y: (1 / t) * (f(t, y) - f(t, y) / t)
y_t0 = 2
a = 1
b = 2

h = 0.25
N = int((b - a) / h)
ys_c,ts_c, h_c = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p])
ys_c[len(ys_c)-1]
```

```
5.3442279856387
```

```
#Literal D)
f = lambda t, y: cos(2 * t) + sin(3 * t)
f_p = lambda t, y: -2 * sin(2 * t) + 3 * cos(3 * t)

y_t0 = 1
a = 0
b = 1

h = 0.25
N = int((b - a) / h)
ys_d,ts_d, h_d = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p])
ys_d[len(ys_d)-1]
```

```
2.201643950842383
```

Pregunta 7

```
# Literal A)
f = lambda t, y: t * exp(3 * t) - 2 * y
f_p = lambda t, y: exp(3 * t) * (3 * t + 1) - 2 * f(t, y)
f_p2 = lambda t, y: exp(3 * t) * (9 * t + 6) - 2 * f_p(t, y)
f_p3 = lambda t, y: exp(3 * t) * (27 * t + 18) - 2 * f_p2(t, y)

y_t0 = 0

a = 0
```

```
b = 1

h = 0.5
N = int((b - a) / h)
ys_a,ts_a, h = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p, f_p
ys_a[len(ys_a)-1]
```

4.263501556162347

```
# Literal B)
f = lambda t, y: 1 + (t - y)**2
f_p = lambda t, y: 2 * (t - y) * (1 - f(t, y))
f_p2 = lambda t, y: 2 * (1 - f(t, y))**2 - 2 * f_p(t, y)
f_p3 = lambda t, y: -4 * (t - y) * (1 - f(t, y)) * f_p(t, y) - 2 * f_p2(t, y)

y_t0 = 1

a = 2
b = 3

h = 0.5
N = int((b - a) / h)
ys_b,ts_b, h_b = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p, f
ys_b[len(ys_b)-1]
```

2.483265566291687

```
# Literal C)
f = lambda t, y: 1 + y / t
f_p = lambda t, y: (1 / t) * (f(t, y) - f(t, y) / t)
f_p2 = lambda t, y: -(1/t**2) * (f_p(t, y) + 2 * f(t, y) / t)
f_p3 = lambda t, y: (2/t**3) * (f_p(t, y) + 3 * f(t, y) / t)

y_t0 = 2
a = 1
b = 2

h = 0.25
N = int((b - a) / h)
ys_c,ts_c, h_c = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p, f
ys_c[len(ys_c)-1]
```

5.273549610534127

```
#Literal D)
f = lambda t, y: cos(2 * t) + sin(3 * t)
f_p = lambda t, y: -2 * sin(2 * t) + 3 * cos(3 * t)
```

```
f_p2 = lambda t, y: -4 * cos(2 * t) - 9 * sin(3 * t)
f_p3 = lambda t, y: 8 * sin(2 * t) - 27 * cos(3 * t)

y_t0 = 1
a = 0
b = 1

h = 0.25
N = int((b - a) / h)
ys_d,ts_d, h_d = ODE_euler_nth(a=a, b=b, y_t0=y_t0, f=f, N=N, f_derivatives = [f_p, f_
ys_d[len(ys_d)-1]
```

```
1.8333234115498083
```