mth3010a-task-2

July 6, 2024

1 Solve all problems while demonstrating each step clearly. The Assignment is worth a total of 10 points

Consider the initial value problem (IVP) given by the following differential equation: $\frac{dy}{dt} = f(t, y) = t - y$ with the initial condition $y(0) = 1$. Solve this IVP using the Euler, Heun (Improved Euler), and Runge-Kutta (4th order) methods over the interval $t \in [0, 2]$ with a step $h = 0.1$. Compare the numerical solutions obtained from each method with **ten** steps with the exact solution.

1.1 1. Implement the Euler method to solve the given IVP.

$$y_{n+1} = y_n + h * f(t_n, y_n)$$

[35]:
```python
import numpy as np
[i for i in np.arange(0, 3, 0.1) if i <= 2.0]
```

[35]:
```
[0.0,
 0.1,
 0.2,
 0.30000000000000004,
 0.4,
 0.5,
 0.6000000000000001,
 0.7000000000000001,
 0.8,
 0.9,
 1.0,
 1.1,
 1.2000000000000002,
 1.3,
 1.4000000000000001,
 1.5,
 1.6,
 1.7000000000000002,
 1.8,
 1.9000000000000001,
 2.0]
```

$$y(0) = 1$$

$$y_{0+1} = y_0 + 0.1 \cdot f(t_0, y_0)$$

$$= 1 + 0.1 \cdot (0 - 1)$$

$$= 0.9$$

$$y_{1+1} = y_1 + 0.1 * f(t_1, y_1)$$

$$= 0.9 + 0.1 * (0.1 - 0.9)$$

$$= 0.82$$

$$y_{2+1} = y_2 + 0.1 * f(t_2, y_2)$$

$$= 0.82 + 0.1 * (0.2 - 0.82)$$

$$= 0.758$$

$$y_{3+1} = y_3 + 0.1 * f(t_3, y_3)$$

$$= 0.758 + 0.1 * (0.3 - 0.758)$$

$$= 0.7122$$

$$y_{4+1} = y_4 + 0.1 * f(t_4, y_4)$$

$$= 0.7122 + 0.1 * (0.4 - 0.7122)$$

$$= 0.68098$$

$$y_{5+1} = y_5 + 0.1 * f(t_5, y_5)$$

$$= 0.68098 + 0.1 * (0.5 - 0.68098)$$

$$= 0.662882$$

$$y_{6+1} = y_6 + 0.1 * f(t_6, y_6)$$

$$= 0.662882 + 0.1 * (0.6 - 0.662882)$$

$$= 0.6565938$$

$$y_{7+1} = y_7 + 0.1 * f(t_7, y_7)$$

$$= 0.6565938 + 0.1 * (0.7 - 0.6565938)$$

$$= 0.66093442$$

$$y_{8+1} = y_8 + 0.1 * f(t_8, y_8)$$

$$= 0.66093442 + 0.1 * (0.8 - 0.66093442)$$

$$= 0.674840978$$

$$y_{9+1} = y_9 + 0.1 * f(t_9, y_9)$$

$$= 0.674840978 + 0.1 * (0.9 - 0.674840978)$$

$$= 0.697356802$$

1.1.1 Observation

From this the code can be written as

```
x - 0.1 ( y - x)
```

here

x – current aproximate value

y – current time value

The code will need to track these two components only for `Euler's method`

1.2 2. Implement the Heun method to solve the given IVP.

$$y_{n+1} = y_n + \frac{h}{2}[f(t_n, y_n) + f(t_n + h, y_n + hf(t_n, y_n))$$

$$y(0) = 1$$

step 1

$$y_{0+1} = y_0 + \frac{0.1}{2}[f(t_0, y_0) + f(t_0 + 0.1, y_0 + 0.1f(t_0, y_0))$$

$$y_1 = 1 + \frac{0.1}{2}[(0 - 1) + ((0 + 0.1) - (1 + 0.1(0 - 0.1))]$$

$$= 0.91$$

step 2

$$y_{1+1} = y_1 + \frac{0.1}{2}[f(t_1, y_1) + f(t_1 + 0.1, y_1 + 0.1f(t_1, y_1))$$

$$y_1 = 0.9195 + \frac{0.1}{2}[(0.1 - 0.91) + ((0.1 + 0.1) - (0.91 + 0.1(0.1 - 0.91))]$$

$$= 0.83805$$

step 3

$$y_{2+1} = y_2 + \frac{0.1}{2}[f(t_2, y_2) + f(t_2 + 0.1, y_2 + 0.1f(t_2, y_2))$$

$$y_1 = 0.83805 + \frac{0.1}{2}[(0.2 - 0.83805) + ((0.2 + 0.1) - (0.83805 + 0.1(0.2 - 0.83805))]$$

$$= 0.78243525$$

step 4

$$y_{3+1} = y_3 + \frac{0.1}{2}[f(t_3, y_3) + f(t_3 + 0.1, y_3 + 0.1f(t_3, y_3))$$

$$y_1 = 0.78243525 + \frac{0.1}{2}[(0.3 - 0.78243525) + ((0.3 + 0.78243525) - (0.78243525 + 0.1(0.3 - 0.78243525))]$$

$$= 0.74165820125$$

step 5

$$y_{4+1} = y_0 + \frac{0.1}{2}[f(t_4, y_4) + f(t_4 + 0.1, y_0 + 0.1f(t_0, y_0))$$

$$y_1 = 0.74165820125 + \frac{0.1}{2}[(0.4 - 0.74165820125) + ((0.4 + 0.74165820125) - (0.74165820125 + 0.1(0.4 - 0.74165820125))$$

$$= 0.71420067213$$

step 6

$$y_{5+1} = y_0 + \frac{0.1}{2}[f(t_5, y_5) + f(t_5 + 0.1, y_5 + 0.1f(t_5, y_5))$$

$$y_1 = 0.71420067213 + \frac{0.1}{2}[(0.5 - 0.71420067213) + ((0.5 + 0.71420067213) - (0.71420067213 + 0.1(0.5 - 0.71420067213))$$

$$= 0.69885160828$$

step 7

$$y_{6+1} = y_6 + \frac{0.1}{2}[f(t_6, y_6) + f(t_6 + 0.1, y_6 + 0.1f(t_6, y_6))$$

$$y_1 = 0.69885160828 + \frac{0.1}{2}[(0.6 - 0.69885160828) + ((0.6 + 0.69885160828) - (0.69885160828 + 0.1(0.6 - 0.69885160828))$$

$$= 0.69446070549$$

step 8

$$y_{7+1} = y_0 + \frac{0.1}{2}[f(t_7, y_7) + f(t_7 + 0.1, y_7 + 0.1f(t_7, y_7))$$

$$y_1 = 0.69446070549 + \frac{0.1}{2}[(0.7 - 0.69446070549) + ((0.7 + 0.69446070549) - (0.69446070549 + 0.1(0.7 - 0.69446070549))$$

$$= 0.69998693847$$

step 9

$$y_{8+1} = y_0 + \frac{0.1}{2}[f(t_8, y_8) + f(t_8 + 0.1, y_8 + 0.1f(t_8, y_8))$$

$$y_1 = 0.69998693847 + \frac{0.1}{2}[(0.8 - 0.69998693847) + ((0.8 + 0.69998693847) - (0.69998693847 + 0.1(0.8 - 0.69998693847))$$

$$= 0.71448817932$$

step 10

$$y_{0+1} = y_0 + \frac{0.1}{2}[f(t_0, y_0) + f(t_0 + 0.1, y_0 + 0.1f(t_0, y_0))$$

$$y_1 = 0.71448817932 + \frac{0.1}{2}[(0.9 - 0.71448817932) + ((0.9 + 0.71448817932) - (0.71448817932 + 0.1(0.9 - 0.71448817932))$$

$$= 0.73711180228$$

1.3 3. Implement the Runge-Kutta method (4th order) to solve the given IVP

$$y_{n+1} = y_n + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$y(0) = 1$$

$$k1 = f(t_0, y_0)$$

$$k_2 = f(t_0 + 0.5h, y_0 + 0.5 * hk1)$$

$$k_3 = f(t_0 + 0.5h, y_0 + 0.5 * hk2)$$

$$K_4 = f(t_0 + h, y_0 + hk3)$$

$$y_{0+1} = 1 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 1 + \frac{0.1}{6}(-1 + 2 * -0.9 + 2 * -0.905 - 0.8095)$$

$$= 0.909675$$

step2

$$y_{0+1} = 0.909675 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.909675 + \frac{0.1}{6}(-0.809675 + 2 * -0.71919125 + 2 * -0.7237154375 - 0.72348922812)$$

$$= 0.83602537328$$

step3

$$y_{0+1} = 0.83602537328 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.83602537328 + \frac{0.1}{6}(-0.63602537328 + 2 * -0.45422410462 + 2 * -0.7237154375 - 0.72348922812)$$

$$= 0.83602537328$$

step4

$$y_{0+1} = 0.83602537328 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.83602537328 + \frac{0.1}{6}(-0.63746 + 2 * -0.55559 + 2 * -0.55968 - 0.48149)$$

$$= 0.78164$$

step5

$$y_{0+1} = 0.78164 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.78164 + \frac{0.1}{6}(-0.48164 + 2 * -0.40756 + 2 * -0.41126 - 0.34051)$$

$$= 0.74064$$

step6

$$y_{0+1} = 0.74064 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.74064 + \frac{0.1}{6}(-0.34064 + 2 * -0.27361 + 2 * -0.27696 - 0.21294)$$

$$= 0.71306$$

step7

$$y_{0+1} = 0.71306 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.71306 + \frac{0.1}{6}(-0.21306 + 2 * -0.15241 + 2 * -0.15544 - 0.09752)$$

$$= 0.69762$$

step8

$$y_{0+1} = 0.69762 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.69762 + \frac{0.1}{6}(-0.09762 + 2 * -0.04274 + 2 * -0.04549 - 0.00692)$$

$$= 0.69317$$

step9

$$y_{0+1} = 0.69317 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.69317 + \frac{0.1}{6}(-0.0.00683 + 2 * -0.056s49 + 2 * -0.054 - 0.10143)$$

$$= 0.69866$$

step10

$$y_{0+1} = 0.69866 + \frac{0.1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= 0.69866 + \frac{0.1}{6}(-0.10134 + 2 * -0.14627 + 2 * -0.14403 - 0.18694)$$

$$= 0.71314$$

1.4 4. Write a python code to check your results and plot all the numerical solutions (the 4. three) along with the exact solution in one graph.

[36]:
```
def euler_method(f, y0, t0, t_end, h):
    N = int((t_end - t0) / h)
    t = t0
    y = y0

    t_values = [t]
    y_values = [y]

    for _ in range(N):
        y += h * f(t, y)
        t += h
        t_values.append(t)
        y_values.append(y)

    return t_values, y_values
```

[37]:
```
def f(t, y):
    return t - y
```

[38]:
```
# set up
y0 = 1
t0 = 0
```

```
t_end = 2 # time end
h = 0.1
```

[39]:
```
euler_t, euler_y = euler_method(f, y0, t0, t_end, h)
```

1.5 Euler output

[40]:
```
for t, y in zip(euler_t, euler_y):
    print(f"t = {t:.1f}, y = {y:.4f}")
```

```
t = 0.0, y = 1.0000
t = 0.1, y = 0.9000
t = 0.2, y = 0.8200
t = 0.3, y = 0.7580
t = 0.4, y = 0.7122
t = 0.5, y = 0.6810
t = 0.6, y = 0.6629
t = 0.7, y = 0.6566
t = 0.8, y = 0.6609
t = 0.9, y = 0.6748
t = 1.0, y = 0.6974
t = 1.1, y = 0.7276
t = 1.2, y = 0.7649
t = 1.3, y = 0.8084
t = 1.4, y = 0.8575
t = 1.5, y = 0.9118
t = 1.6, y = 0.9706
t = 1.7, y = 1.0335
t = 1.8, y = 1.1002
t = 1.9, y = 1.1702
t = 2.0, y = 1.2432
```

[41]:
```
def heun_method(f, y0, t0, t_end, h):
    N = int((t_end - t0) / h)
    t = t0
    y = y0

    t_values = [t]
    y_values = [y]

    for _ in range(N):
        y1 = y + h * f(t, y) # yn + hf(tn, yn)

        y = y + (h / 2) * (f(t, y) + f(t + h, y1))
        t += h

        t_values.append(t)
```

```
        y_values.append(y)

    return t_values, y_values
```

[42]:
```
heun_t, heun_y = heun_method(f, y0, t0, t_end, h)
```

[43]:
```
for t, y in zip(heun_t, heun_y):
    print(f"t = {t:.1f}, y = {y:.5f}")
```

```
t = 0.0, y = 1.00000
t = 0.1, y = 0.91000
t = 0.2, y = 0.83805
t = 0.3, y = 0.78244
t = 0.4, y = 0.74160
t = 0.5, y = 0.71415
t = 0.6, y = 0.69881
t = 0.7, y = 0.69442
t = 0.8, y = 0.69995
t = 0.9, y = 0.71446
t = 1.0, y = 0.73708
t = 1.1, y = 0.76706
t = 1.2, y = 0.80369
t = 1.3, y = 0.84634
t = 1.4, y = 0.89444
t = 1.5, y = 0.94746
t = 1.6, y = 1.00496
t = 1.7, y = 1.06648
t = 1.8, y = 1.13167
t = 1.9, y = 1.20016
t = 2.0, y = 1.27164
```

[44]:
```
def Runge_method(f , y0, t0, t_end, h):
    N = int((t_end - t0) / h)
    t = t0
    y = y0

    t_values = [t]
    y_values = [y]
    k1_val = []
    k2_val = []
    k3_val = []
    k4_val = []
    for _ in range(N):
        k1 = f(t, y)
        k2 = f(t + 0.5*h, y + 0.5 * h * k1)
        k3 = f(t + 0.5*h, y + 0.5 * h * k2)
        k4 = f(t + h, y + h * k3)
```

```python
        y = y + h/6 * (k1 + 2*k2 + 2*k3 + k4)
        t += h

        t_values.append(t)
        y_values.append(y)
        k1_val.append(k1)
        k2_val.append(k2)
        k3_val.append(k3)
        k4_val.append(k4)

    return t_values, y_values, k1_val, k2_val, k3_val, k4_val
```

[45]:
```python
Runge_t, Runge_y, Runge_k1, Runge_k2, Runge_k3, Runge_k4 = Runge_method(f, y0,␣
 ↪t0, t_end, h)
```

[46]:
```python
for t, y in zip(Runge_t, Runge_y):
    print(f"t = {t:.1f}, y = {y:.5f}")
```

```
t = 0.0, y = 1.00000
t = 0.1, y = 0.90968
t = 0.2, y = 0.83746
t = 0.3, y = 0.78164
t = 0.4, y = 0.74064
t = 0.5, y = 0.71306
t = 0.6, y = 0.69762
t = 0.7, y = 0.69317
t = 0.8, y = 0.69866
t = 0.9, y = 0.71314
t = 1.0, y = 0.73576
t = 1.1, y = 0.76574
t = 1.2, y = 0.80239
t = 1.3, y = 0.84506
t = 1.4, y = 0.89319
t = 1.5, y = 0.94626
t = 1.6, y = 1.00379
t = 1.7, y = 1.06537
t = 1.8, y = 1.13060
t = 1.9, y = 1.19914
t = 2.0, y = 1.27067
```

$$e^{\int 1 dt} = e^t$$

$$e^t \frac{dy}{dt} + e^t y = te^t$$

$$\int \frac{d}{dt}(e^t y) = \int te^t$$

$$e^t y = \int e^t$$

$$e^t y = te^t - e^t + C$$

$$y = t - 1 + Ce^{-t}$$

[47]:
```
# Plot
t_val =np.array(np.arange(0, 2.1, 0.1))
t_val
```

[47]:
```
array([0. , 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1. , 1.1, 1.2,
       1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2. ])
```

[48]:
```
actual = t_val - 1 + np.exp(-t_val)
actual
```

[48]:
```
array([0.        , 0.00483742, 0.01873075, 0.04081822, 0.07032005,
       0.10653066, 0.14881164, 0.1965853 , 0.24932896, 0.30656966,
       0.36787944, 0.43287108, 0.50119421, 0.57253179, 0.64659696,
       0.72313016, 0.80189652, 0.88268352, 0.96529889, 1.04956862,
       1.13533528])
```

[49]:
```
import pandas as pd

plot_data = pd.DataFrame({
    't_val': t_val,
    'Actual': actual,
    'Euler': euler_y,
    'Heun': heun_y,
    'Runge-Kutta': Runge_y
})

plot_data
```

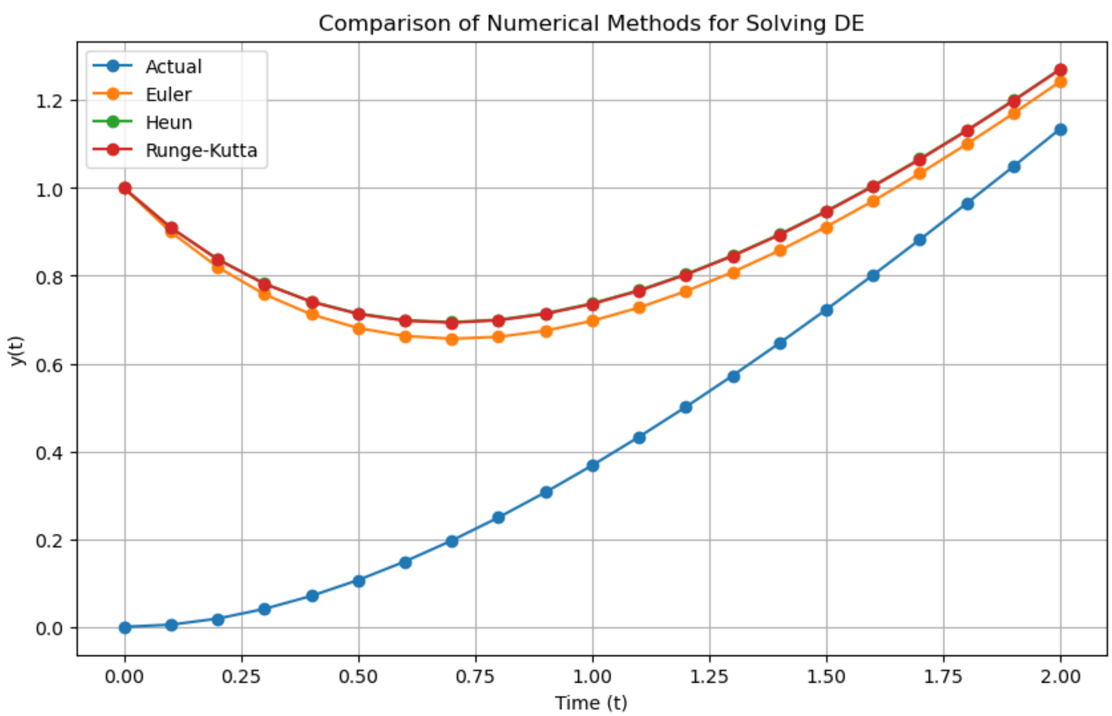
[49]:

	t_val	Actual	Euler	Heun	Runge-Kutta
0	0.0	0.000000	1.000000	1.000000	1.000000
1	0.1	0.004837	0.900000	0.910000	0.909675
2	0.2	0.018731	0.820000	0.838050	0.837462
3	0.3	0.040818	0.758000	0.782435	0.781637
4	0.4	0.070320	0.712200	0.741604	0.740641
5	0.5	0.106531	0.680980	0.714152	0.713062
6	0.6	0.148812	0.662882	0.698807	0.697624
7	0.7	0.196585	0.656594	0.694420	0.693171
8	0.8	0.249329	0.660934	0.699951	0.698659
9	0.9	0.306570	0.674841	0.714455	0.713140
10	1.0	0.367879	0.697357	0.737082	0.735760
11	1.1	0.432871	0.727621	0.767059	0.765743
12	1.2	0.501194	0.764859	0.803689	0.802389

13	1.3	0.572532	0.808373	0.846338	0.845064
14	1.4	0.646597	0.857536	0.894436	0.893195
15	1.5	0.723130	0.911782	0.947465	0.946261
16	1.6	0.801897	0.970604	1.004955	1.003794
17	1.7	0.882684	1.033544	1.066485	1.065368
18	1.8	0.965299	1.100189	1.131669	1.130598
19	1.9	1.049569	1.170170	1.200160	1.199138
20	2.0	1.135335	1.243153	1.271645	1.270671

[50]:
```
import matplotlib.pyplot as plt

ax = plot_data.plot(x='t_val', y=['Actual', 'Euler', 'Heun', 'Runge-Kutta'],␣
 ↪figsize=(10, 6), marker='o')
ax.set_title('Comparison of Numerical Methods for Solving DE')
ax.set_xlabel('Time (t)')
ax.set_ylabel('y(t)')
plt.grid(True)
plt.show()
```



[ ]:
```
```