SciComp2-M17

October 10, 2021

0.0.1 25.5

Solve from t = 0 to 3, with h=0.1 using

```
*a) Heun (without corrector, which means only with Predictor method, using eq. 25.15)
*b) Ralston's second-order RK method
```

dy/dt = ysin^3(t) with y(0) = 1

[1]:
```
import math
import matplotlib.pyplot as plt
```

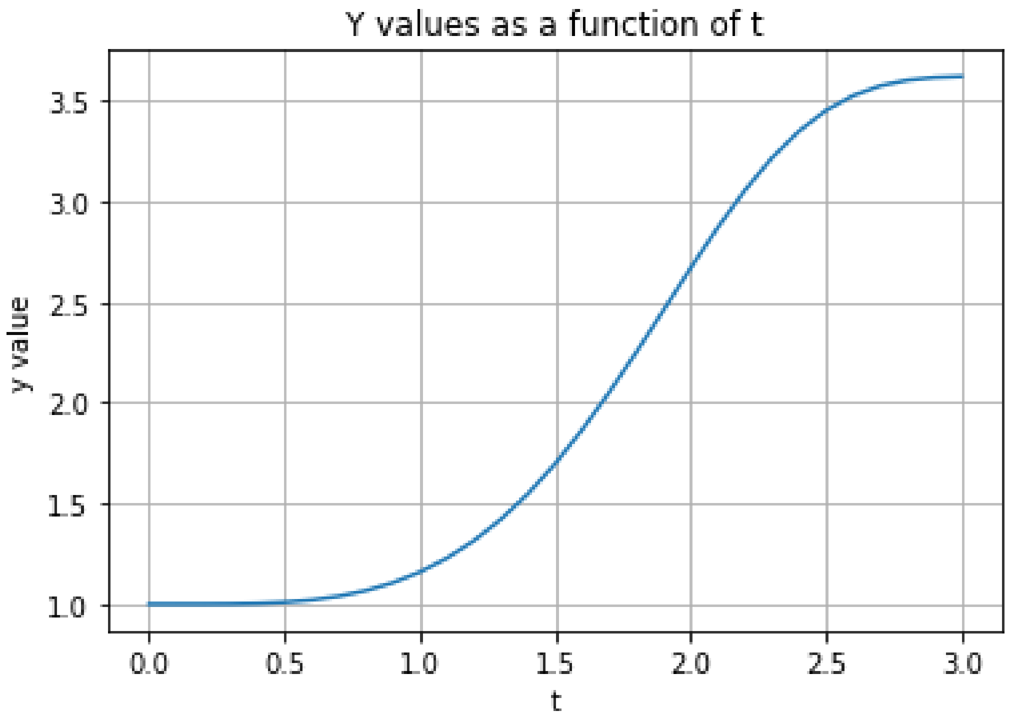
0.0.2 Heun method

[2]:
```
def plot(t, y):
    fig, ax = plt.subplots()
    ax.plot(t, y)
    ax.set(xlabel = 't', ylabel = 'y value', title = 'Y values as a function of␣
 ↪t')
    ax.grid()
    plt.show()
```

[3]:
```
Heun_y_vals = [1.0]
t_vals = [0.0]
y_derivatives = []
h = 0.1
j = 0
i = h

while i <= 3:
    val = Heun_y_vals[j] * (math.sin(i)) ** 3
    y_derivatives.append(val)
    Heun_y_vals.append(Heun_y_vals[j] + y_derivatives[j] * h)
    i += h
    j += 1
    t_vals.append(i)

print(f"Heun's Values: {Heun_y_vals}")
#print(t)
```

```
plot(t_vals, Heun_y_vals)
```

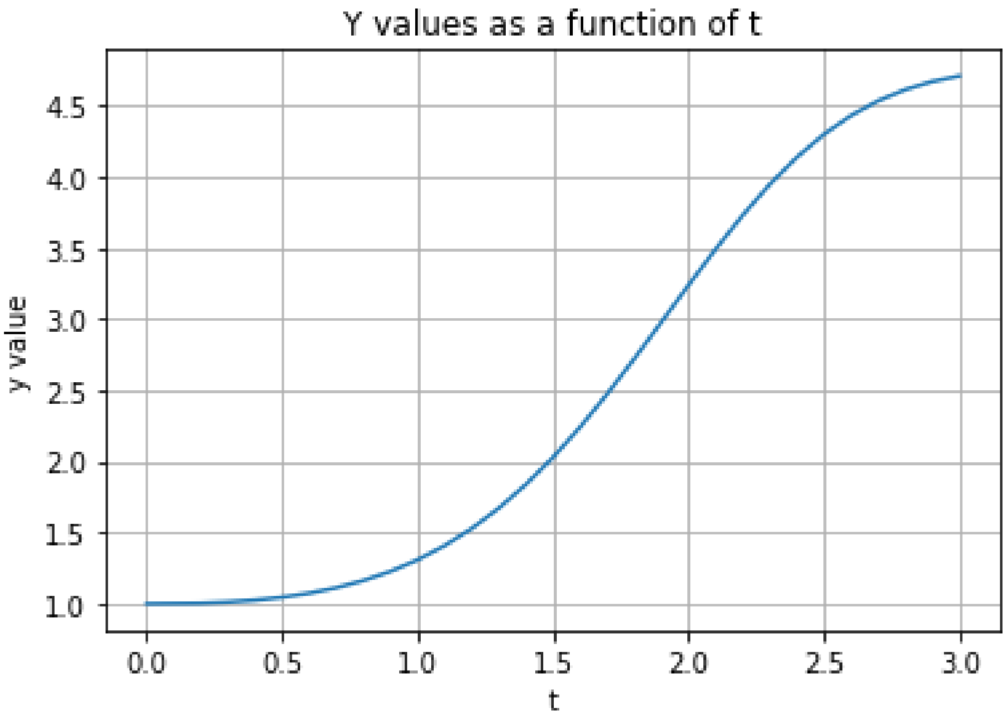
```
Heun's Values: [1.0, 1.0000995010819786, 1.0008837170793092, 1.0034668405730476,
1.0093927121722186, 1.0205157562767815, 1.0388870755458857, 1.0666628563665415,
1.106038851499425, 1.1592006323627035, 1.2282684996259663, 1.3152104958275317,
1.4216977532474424, 1.5488842813790398, 1.6971096014922207, 1.8655483691436092,
2.051864668553397, 2.251962978531938, 2.4599490648196, 2.66840427563857,
2.869022099008072, 3.0535586188805475, 3.214935210654488, 3.348248597264794,
3.4514350629779473, 3.525417993433325, 3.5737127623975775, 3.601610042447935,
3.6151489936347323, 3.6200998278913508]
```



0.0.3 Ralston's second order Runge-Kutta method

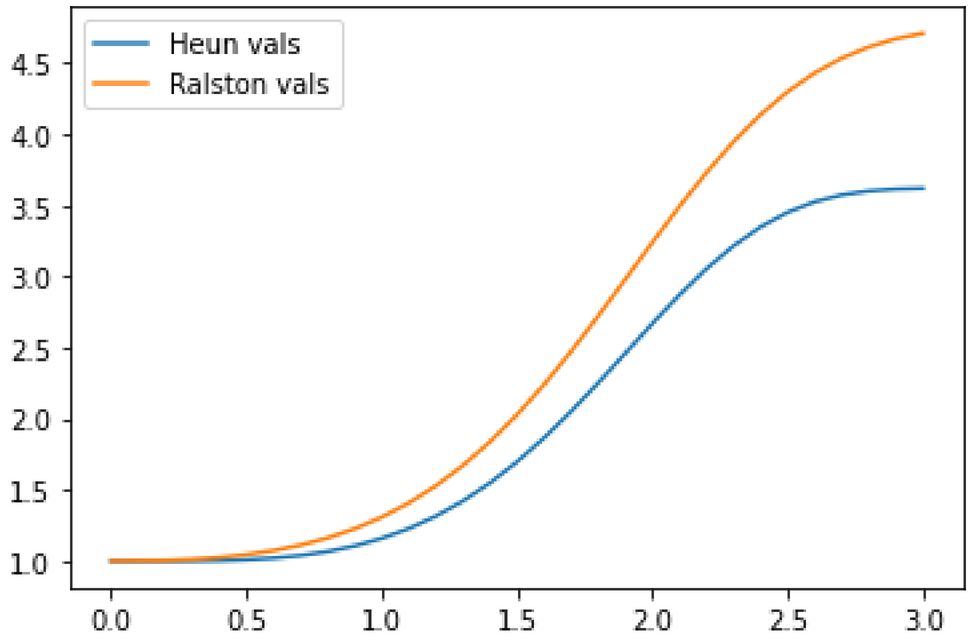
[4]:
```
Ralston_y_vals = [1.0]
t_vals = [0.0]
k1_vals = []
k2_vals = []

h = 0.1
m = 0
t = h

while t <= 3:
    k1_vals.append(Ralston_y_vals[m] * math.sin(t))
    k2_vals.append((Ralston_y_vals[m] + (0.75 * k1_vals[m] * h)) * (math.sin(t␣
↪+ (0.75 * h)) ** 3))
    Ralston_y_vals.append(Ralston_y_vals[m] + (((1/3) * k1_vals[m]) + ((2/3) *␣
↪k2_vals[m])) * h)
    m += 1
    t += h
    t_vals.append(t)

print(f"Ralston's method: y_values : {Ralston_y_vals}\n")

plot(t_vals, Ralston_y_vals)
```

```
Ralston's method: y_values : [1.0, 1.0036822718866794, 1.0116887305544042,
1.025042123815665, 1.0450754886769964, 1.0733857981732051, 1.1117897793162652,
1.162283119379014, 1.2269977666250178, 1.30814551232975, 1.4079311684594842,
1.5284169903257443, 1.6713231784897715, 1.83775907000019, 2.0278972660196333,
2.240627897557459, 2.473258357787944, 2.72134619804993, 2.97875681300619,
3.238011126903092, 3.4909280728129035, 3.729483608303731, 3.946729324163448,
4.137572592919976, 4.299239451693948, 4.4313190570585395, 4.535396425063185,
4.614377876064314, 4.671669253131586, 4.710371283971129]
```



[5]:
```
def plot2vals(t, y1, y2):
    plt.plot(t, y1, label = 'Heun vals')
    plt.plot(t, y2, label = 'Ralston vals')
    plt.legend()
    plt.show()
```

[6]:
```
# Plotting values by 2 functions together
plot2vals(t_vals, Heun_y_vals, Ralston_y_vals)
```



[ ]:
```
```