Section 2.5: Improved Euler's Method

Prior Knowledge

To approximate an ordinary differential equation of the form

$$\frac{dy}{dx} = f(x, y), \quad y(x_0) = y_0$$

we need to recall the limit definition of the derivative, which is

$$\frac{dy}{dx} = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

Last class, we use the following algorithm,

Algorithm: The Euler Method (Edwards, Penney, Calvis, 2023)

Given the initial value problem

$$\frac{dy}{dx} = f(x, y), \quad y(x_0) = y_0$$

Euler's method with step size h consists of applying the iterative formula

$$y_{n+1} = h \cdot f(x_n, y_n) + y_n \quad (n \geq 0)$$

to calculate successive approximations $y_1, y_2, y_3, \cdot$ to the [true] values $y(x_1), y(x_2), y(x_3), \cdot$ of the [exact] solution $y = y(x)$ at the points $x_1, x_2, x_3, \ldots$, respectively.

NEW: A way to in which we can increased accuracy which is also easily obtained is known as the improved Euler method (also known as Runge-Kutta Method 2nd order.

Let's watch the following video (From 5 mins to 9:30 mins) from Phanimations https://youtu.be/dShtlMl69kY?si=I53pfrTqZN3fbHXR&t=304

Algorithm: The Improved Euler Method (Edwards, Penney, Calvis, 2023)

Given the initial value problem

$$\frac{dy}{dx} = f(x, y), \quad y(x_0) = y_0$$

the improved Euler method with step size h consists in applying the iterative formulas

$$\begin{aligned} k_1 &= f(x_n, y_n) \\ u_{n+1} &= y_n + h \cdot k_1 \\ k_2 &= f(x_{n+1}, u_{n+1}) \\ y_{n+1} &= y_n + h \cdot \frac{1}{2}(k_1 + k_2) \end{aligned}$$

to compute successive approximations $y_1, y_2, y_3, \ldots$ to the (true) values $y(x_1), y(x_2), y(x_3), \ldots$ of the (exact) solution $y = y(x)$ at the points $x_1, x_2, x_3, \ldots$, respectively.

(The following code was developed and/or tweaked from Dr Jon Shiach's video https://www.youtube.com/watch?v=N7Oh0mk4YGc&t=378s)

Problem 1:

Apply the Improved Euler's method to approximate the solution to the initial value problem on the interval $\left[0, \frac{1}{2}\right]$, with step size $h = 0.25$. Compare the three-decimal-place values of the two approximations (One found with Euler the other with Improved Euler) at $x = \frac{1}{2}$ with the value of $y(1/2)$ of the actual solution.

$$y' = y, \quad y(0) = 3, \quad y(x) = 3e^x$$

In [3]:
```
import numpy as np
import math

# Improved Euler method
def improved_euler(f, tn, yn, h):
    k1 = f(tn, yn)
    k2 = f(tn + h, yn + h * k1)
    return yn + h * 0.5 * (k1 + k2)

#Solver function
def solveIVP(f, tspan, y0, h, solver):
    t = np.arange(tspan[0], tspan[1] + h, h)
    y = np.zeros(len(t))
    y[0] = y0

    for n in range(len(t) - 1):
        y[n+1] = solver(f, t[n], y[n], h)

    return t, y

# Euler method
def euler(f, tn, yn, h):
    return yn + h * f(tn,yn)

# Define IVP
def f(t,y):
    return y # becuase y'=f(t,y)=y

# Problem Specific
tspan = [0, 0.5]
x0 = 0
y0 = 3
x = 0.5
h = 0.25

# Solve IVP
t, y = solveIVP(f, tspan, y0, h, euler)

# Exact solution
def exact(x):
    return 3 * math.exp(x)

e = exact(x)

# Collect Euler results
t1, y1 = solveIVP(f, tspan, y0, h, euler)
```

```
# Collect Improved Euler results
t2, y2 = solveIVP(f, tspan, y0, h, improved_euler)

# Print table
print(" Steps (h) | Euler y(0.5) | Improved Euler y(0.5) |  Exact y(0.5) ")
print("-------------------------------------------------------------------")
for n in range(len(t)):
    print(f" {t1[n]:6.2f}    |    {y1[n]:9.3f} |   {y2[n]:9.3f}           |    {exact(x):9.3f} ")
```

```
 Steps (h) | Euler y(0.5) | Improved Euler y(0.5) |  Exact y(0.5)
-------------------------------------------------------------------
   0.00    |        3.000 |       3.000           |        4.946
   0.25    |        3.750 |       3.844           |        4.946
   0.50    |        4.688 |       4.925           |        4.946
```

As you could see the Improved Euler's Method is better at approximating our function!!!

Problem 2:

Apply the Improved Euler's method to approximate the solution to the initial value problem on the interval $\left[0, \frac{1}{2}\right]$, with step size $h = 0.1$. Compare the three-decimal-place values of the two approximations (One found with Euler the other with Improved Euler) at $x = \frac{1}{2}$ with the value of $y(1/2)$ of the actual solution.

$$y' = -2xy, \quad y(0) = 2, \quad y(x) = 2e^{-x^2}$$

In [5]:
```
# Define IVP
def f(t,y):
    return -2 * t * y # becuase y'=f(t,y)=-2xy

# Problem Specific
tspan = [0, 0.5]
x0 = 0
y0 = 2
h = 0.1

# Collect Euler results
t1, y1 = solveIVP(f, tspan, y0, h, euler)

# Collect Improved Euler results
t2, y2 = solveIVP(f, tspan, y0, h, improved_euler)

# Print table
print(" Steps (h) | Euler y(0.5) | Improved Euler y(0.5) ")
print("-------------------------------------------------------------------")
for n in range(len(t)):
    print(f" {t1[n]:6.2f}    |    {y1[n]:9.3f} |   {y2[n]:9.3f}           ")
```

```
 Steps (h) | Euler y(0.5) | Improved Euler y(0.5)
-------------------------------------------------------------------
   0.00    |        2.000 |       2.000
   0.10    |        2.000 |       1.980
   0.20    |        1.960 |       1.921
```

Sources:

1. Differential Equations and Boundary Value Problems (2023) by Edwards, Penney, Calvis
2. Dr. Jon Shiach's Euler Method (Python) video found at https://www.youtube.com/watch?v=N7Oh0mk4YGc&t=378s
3. Phanimations video found at https://www.youtube.com/watch?v=N7Oh0mk4YGc&t=378s

In [ ]:

```

```