Métodos numéricos para solução de EDOs

Exemplo 1: $y' = 1 - t + 4y$

```
import numpy as np
import matplotlib.pylab as plt

def f(t, y):
    return 1 - t + 4*y
```

Euler:

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

```

def EDO_euler(f, y0, t0, NUMBER_OF_STEPS=100, h=0.01):

    y = np.zeros(NUMBER_OF_STEPS, dtype=np.float32)
    t = np.zeros(NUMBER_OF_STEPS, dtype=np.float32)

    y[0] = 1
    t[0] = 0

    for n in range(0, NUMBER_OF_STEPS - 1):
        K1 = f(t[n], y[n])
        y[n+1] = y[n] + K1*h
        t[n+1] = t[n]+h

    return (t, y)
```

Euler Melhorado:

$$K_1 = f(t_n, y_n)$$

$$K_2 = f(t_{n+1}, y_n + hK_1)$$

$$y_{n+1} = y_n + h(\frac{K_1 + K_2}{2})$$

```
def EDO_heun(f, y0, t0, NUMBER_OF_STEPS=100, h=0.01):
    y = np.zeros(NUMBER_OF_STEPS, dtype=np.float32)
    t = np.zeros(NUMBER_OF_STEPS, dtype=np.float32)

    y[0] = 1
    t[0] = 0

    for n in range(0, NUMBER_OF_STEPS - 1):
        t[n+1] = t[n]+h
        K1 = f(t[n], y[n])
        K2 = f(t[n+1], y[n] + K1*h)
        y[n+1] =  y[n] + 0.5*(K1 + K2)*h

return (t, y)
```

Plotando os gráficos

```python

te, ye = EDO_euler(f, 1, 0)
th, yh = EDO_heun(f, 1, 0)

plt.plot(te, ye, "ro")
plt.plot(th, yh, "bs")
plt.show()
```