International IT University
Faculty of Computer technologies and cyber security
Department: MCM

Report

In the discipline «Numerical Analysis»

Executed: Taldybayev B.A.

Group: IT3-2203

Lecturer: Шахан Н.Ш.

Almaty, 2025

Task 6: Predator-Prey model

1. $$\begin{cases} \frac{dX}{dt} = \alpha X - \beta XY \\ \frac{dY}{dt} = \delta XY - \gamma Y \end{cases}$$, where α = 0.1, β = 0.02, δ = 0.01, γ = 0.1 and at t = 0 X(0) = 100, Y(0) = 30

2. Euler's method:

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

3. Runge-Kutte of the 2-nd order (k1):

$$k_1 = f(t_n, y_n)$$

k2:

$$k_2 = f(t_n + h/2, y_n + (h/2)k_1)$$

So:

$$y_{n+1} = y_n + hk_2$$

4. Runge-Kutte of the 4-th order:
The same as 2-nd order, but there are also k3 and k4:

$$k_3 = hf(t_n + h/2, y_n + k_2/2)$$

$$k_4 = hf(t_n + h, y_n + k_3)$$

So:

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

Code and graph:

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


alpha = 0.1
beta = 0.02
delta = 0.01
gamma = 0.1


def predator_prey(t, state):
    X, Y = state
    dXdt = alpha * X - beta * X * Y
    dYdt = delta * X * Y - gamma * Y
    return np.array([dXdt, dYdt])


def euler_method(f, y0, t):
    n = len(t)
    y = np.zeros((n, len(y0)))
    y[0] = y0
    for i in range(n - 1):
        h = t[i+1] - t[i]
        y[i+1] = y[i] + h * f(t[i], y[i])
    return y


def runge_kutta_2nd_order(f, y0, t):
    n = len(t)
    y = np.zeros((n, len(y0)))
    y[0] = y0
    for i in range(n - 1):
        h = t[i+1] - t[i]
        k1 = f(t[i], y[i])
        k2 = f(t[i] + h / 2, y[i] + h / 2 * k1)
        y[i+1] = y[i] + h * k2
    return y


def runge_kutta_4th_order(f, y0, t):
    n = len(t)
    y = np.zeros((n, len(y0)))
    y[0] = y0
    for i in range(n - 1):
        h = t[i+1] - t[i]
        k1 = h * f(t[i], y[i])
        k2 = h * f(t[i] + h / 2, y[i] + k1 / 2)
        k3 = h * f(t[i] + h / 2, y[i] + k2 / 2)
        k4 = h * f(t[i] + h, y[i] + k3)
        y[i+1] = y[i] + (k1 + 2 * k2 + 2 * k3 + k4) / 6
    return y


X0, Y0 = 40, 9
t = np.linspace(0, 200, 1000)
y0 = np.array([X0, Y0])

sol_euler = euler_method(predator_prey, y0, t)
sol_rk2 = runge_kutta_2nd_order(predator_prey, y0, t)
sol_rk4 = runge_kutta_4th_order(predator_prey, y0, t)
```

```python
plt.figure(figsize=(10, 6))

plt.plot(t, sol_euler[:, 0], label="Prey (Euler)", linestyle="dotted",
color='green')
plt.plot(t, sol_euler[:, 1], label="Predator (Euler)", linestyle="dotted",
color='green')

plt.plot(t, sol_rk2[:, 0], label="Prey (RK2)", linestyle="dashed",
color='blue')
plt.plot(t, sol_rk2[:, 1], label="Predator (RK2)", linestyle="dashed",
color='blue')

plt.plot(t, sol_rk4[:, 0], label="Prey (RK4)", color='red')
plt.plot(t, sol_rk4[:, 1], label="Predator (RK4)", color='red')

plt.xlabel("Time")
plt.ylabel("Population")
plt.title("Predator-Prey Model: Euler vs RK2 vs RK4")
plt.legend()
plt.grid()
plt.show()
```