Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет по лабораторной работе №1

По дисциплине: Имитационное моделирование робототехнических систем

Тема: Сравнительный анализ методов интегрирования

Автор: Толстоусов Я.В.., группа R4134c

Принял: Ракшин Е.А

Санкт-Петербург, 2025

Цель работы: сравнение методов интегрирования – методы явного и неявного Эйлера, метод Ранге-Кутта 4-го порядка.

Данное уравнение: $-6.46\ddot{x} - 7.76\dot{x} - 6.39x = -4.13$

Аналитическое решение уравнения при нулевых начальных условиях:

$$x(t) = e^{(-0.6t)}*(\cos(0.79t)+0.42\sin(0.79t))+0.65$$

Работа программы:

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

a, b, c, d = -6.46, -7.76, -6.39, -4.13

def ode(x):
    """
    State vector x = [x, x_dot]
    """
    x_ddot = (d - c*x[0] - b*x[1])/a
    return np.array([x[1], x_ddot])

def analytical_solution(t):
    """
    Аналитическое решение для начальных условий x(0)=0.1, x'(0)=0.0
    Решение: x(t) = e^(-0.6t)*(0.55*cos(0.79t) + 0.42*sin(0.79t)) + 0.65
    """

    x_pos = np.exp(-0.6*t)*(0.55*np.cos(0.79*t) + 0.42*np.sin(0.79*t)) +
0.65


    term1 = -0.6*np.exp(-0.6*t)*(0.55*np.cos(0.79*t) +
0.42*np.sin(0.79*t))
    term2 = np.exp(-0.6*t)*(-0.55*0.79*np.sin(0.79*t) +
0.42*0.79*np.cos(0.79*t))
    x_vel = term1 + term2

    return np.array([x_pos, x_vel])

def forward_euler(fun, x0, Tf, h):
    t = np.arange(0, Tf + h, h)
    x_hist = np.zeros((len(x0), len(t)))
    x_hist[:, 0] = x0

    for k in range(len(t) - 1):
        x_hist[:, k + 1] = x_hist[:, k] + h * fun(x_hist[:, k])
```

```
    return x_hist, t

def backward_euler(fun, x0, Tf, h, tol=1e-8, max_iter=100):
    t = np.arange(0, Tf + h, h)
    x_hist = np.zeros((len(x0), len(t)))
    x_hist[:, 0] = x0

    for k in range(len(t) - 1):
        x_hist[:, k + 1] = x_hist[:, k]

        for i in range(max_iter):
            x_next = x_hist[:, k] + h * fun(x_hist[:, k + 1])
            error = np.linalg.norm(x_next - x_hist[:, k + 1])
            x_hist[:, k + 1] = x_next
            if error < tol:
                break

    return x_hist, t

def runge_kutta4(fun, x0, Tf, h):
    t = np.arange(0, Tf + h, h)
    x_hist = np.zeros((len(x0), len(t)))
    x_hist[:, 0] = x0

    for k in range(len(t) - 1):
        k1 = fun(x_hist[:, k])
        k2 = fun(x_hist[:, k] + 0.5 * h * k1)
        k3 = fun(x_hist[:, k] + 0.5 * h * k2)
        k4 = fun(x_hist[:, k] + h * k3)

        x_hist[:, k + 1] = x_hist[:, k] + (h / 6.0) * (k1 + 2*k2 + 2*k3 +
k4)

    return x_hist, t


Tf = 10.0
h = 0.1
x0 = np.array([0.1, 0.0])


t_analytical = np.arange(0, Tf + h, h)
x_analytical = np.array([analytical_solution(t) for t in t_analytical]).T


x_fe, t_fe = forward_euler(ode, x0, Tf, h)
x_be, t_be = backward_euler(ode, x0, Tf, h)
x_rk4, t_rk4 = runge_kutta4(ode, x0, Tf, h)
```

```
plt.figure(figsize=(12, 8))

plt.plot(t_fe, x_fe[0, :], label='Forward Euler')
plt.plot(t_be, x_be[0, :], label='Backward Euler')
plt.plot(t_rk4, x_rk4[0, :], label='RK4')
plt.plot(t_analytical, x_analytical[0, :], label='Analytical',
color='black', linestyle='--', linewidth=2)

plt.xlabel('Time')
plt.ylabel('Position')
plt.legend()
plt.title('Comparison of Numerical Methods with Analytical Solution')
plt.grid(True)
plt.show()
```

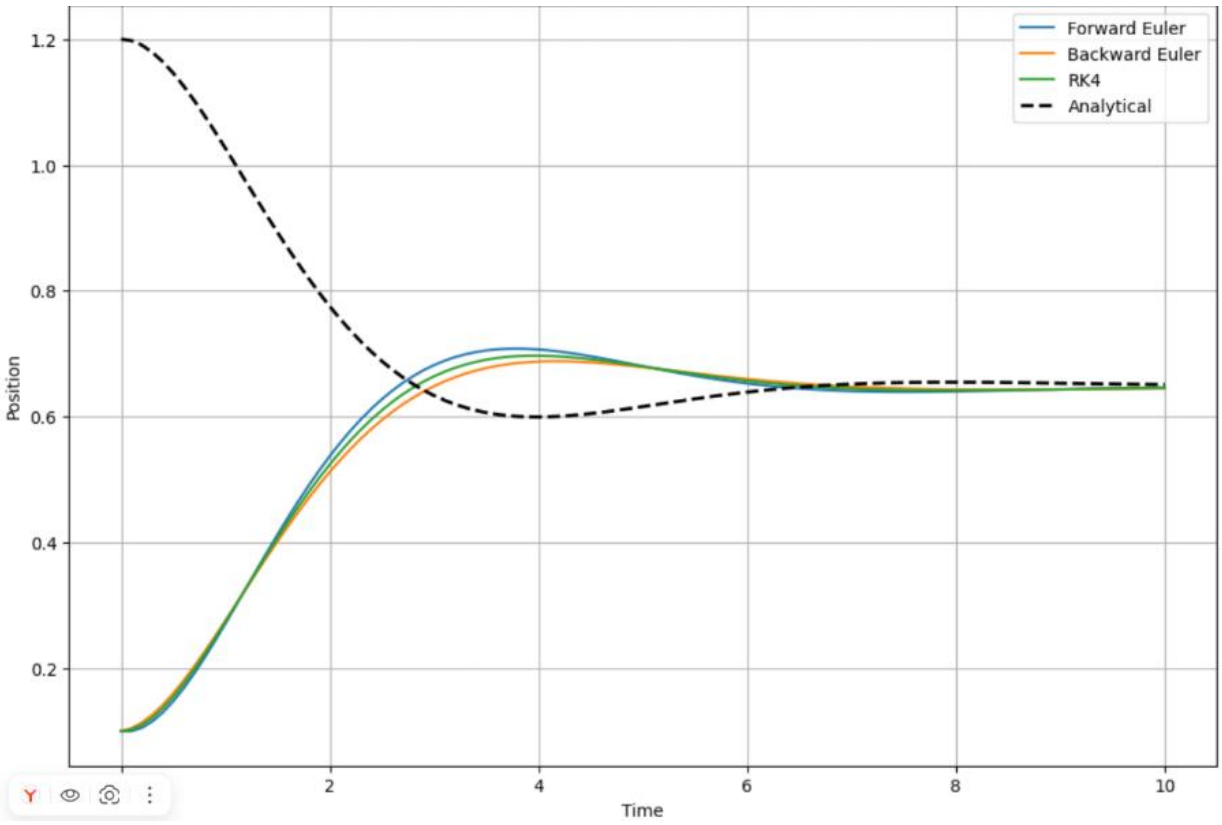

Рис. 1. Положение

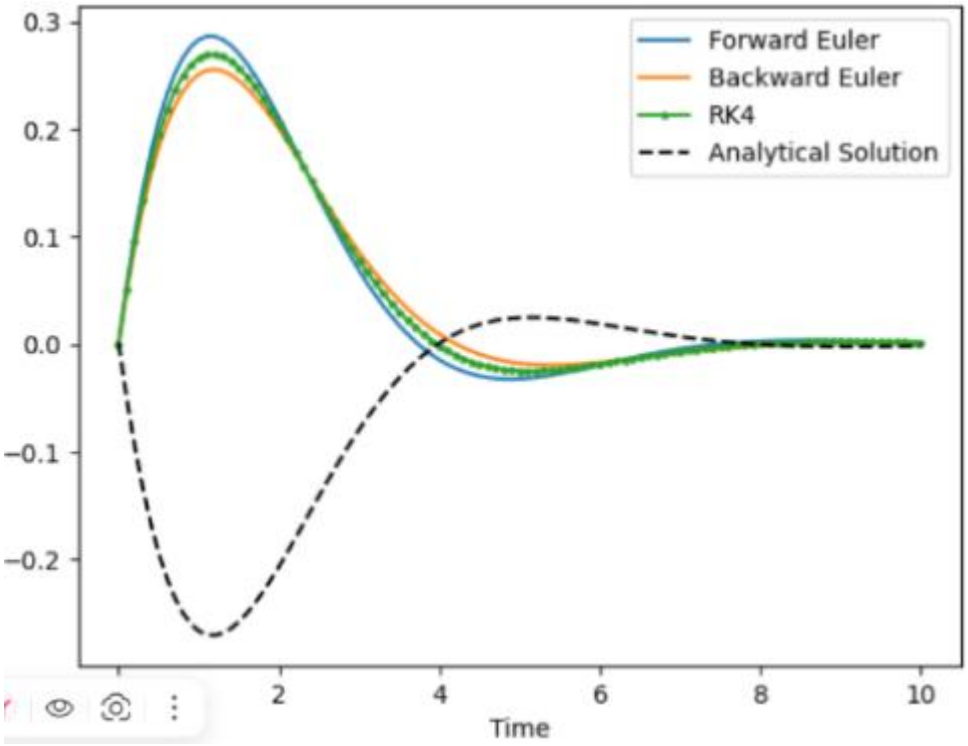

Рис. 2. Скорость

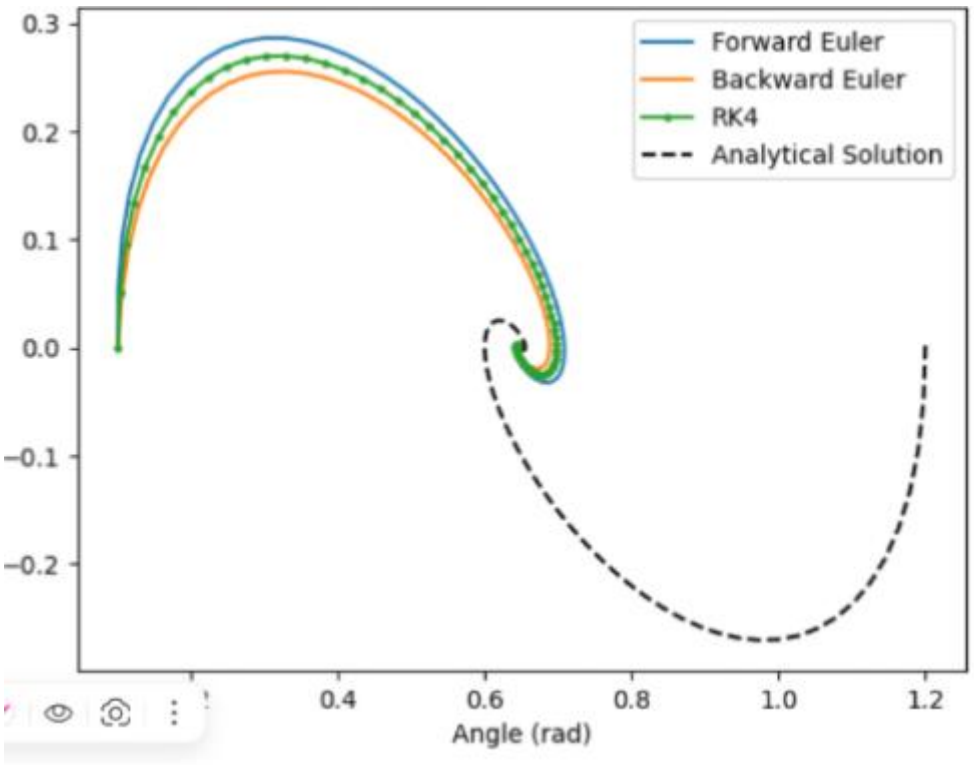

Рис. 3. Фазовый портрет

Вывод: в ходе данной работы было выполнено три метода интегрирования, каждый из которых показал хорошую точность в сравнении с аналитическим решением, на графиках положения, скорости и фазового портрета видно также, что есть ошибки между данными методами и аналитическим решением. Также график аналитического решения инверсивен относительно других решений.