МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»
(Университет ИТМО)

Факультет систем управления и робототехники

ОТЧЕТ
по лабораторной работе №2
по дисциплине
«Имитационное моделирование робототехнических систем»

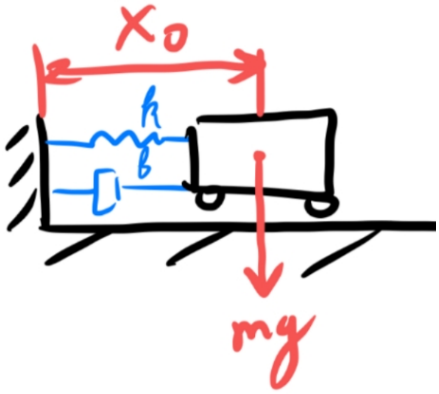
Студент: Петрекеев К.С.

Группа № R4133c 505881

Преподаватель: Ракшин Егор Александрович

Санкт-Петербург 2025

Задание



Вариант 2

- m = 1
- k = 9.4
- b = 0.04
- I = 0.41
- theta_0 = 0.942507068803016
- x_0 = 0.93

Численное решение

Лагранжиан системы вычисляется:

$$L = K - P = \frac{1}{2} m \dot{x}^2 - \frac{1}{2} k x^2$$

Уравнение Лагранжа:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{x}}\right) - \frac{\partial L}{\partial x} = Q$$

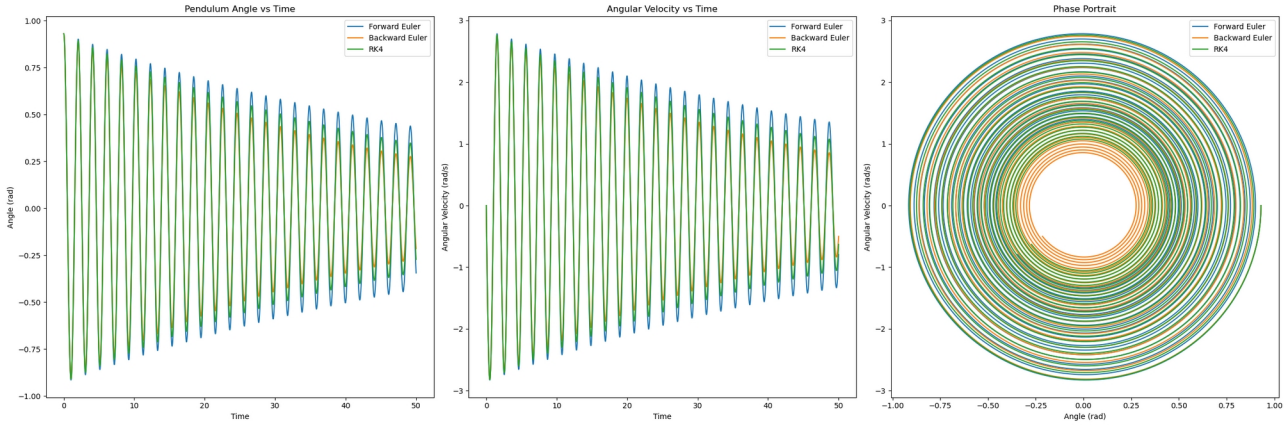
$$m\ddot{x} + b\dot{x} + kx = 0$$

Код программы, решающий уравнение с использованием разных интеграторов:

```
m = 1
k = 9.4
b = 0.04
x_0 = 0.93
l = 0.41 # for var1
theta_0 = 0.942507068803016 # for var1

def Dynamics(y):
    dydt = np.array([y[1], (-k * y[0] - b * y[1]) / m])
    return dydt

x0 = np.array([x_0, 0.0]) # [angle, angular_velocity]
Tf = 50
h = 0.001

x_fe, t_fe = forward_euler(Dynamics, x0, Tf, h)
```

```
x_be, t_be = backward_euler(Dynamics, x0, Tf, h)

x_rk4, t_rk4 = runge_kutta4(Dynamics, x0, Tf, h)
```

Получим результат численного решения:



Система устойчива, стремится к положению равновесия. Имеет затухающий колебательный характер.

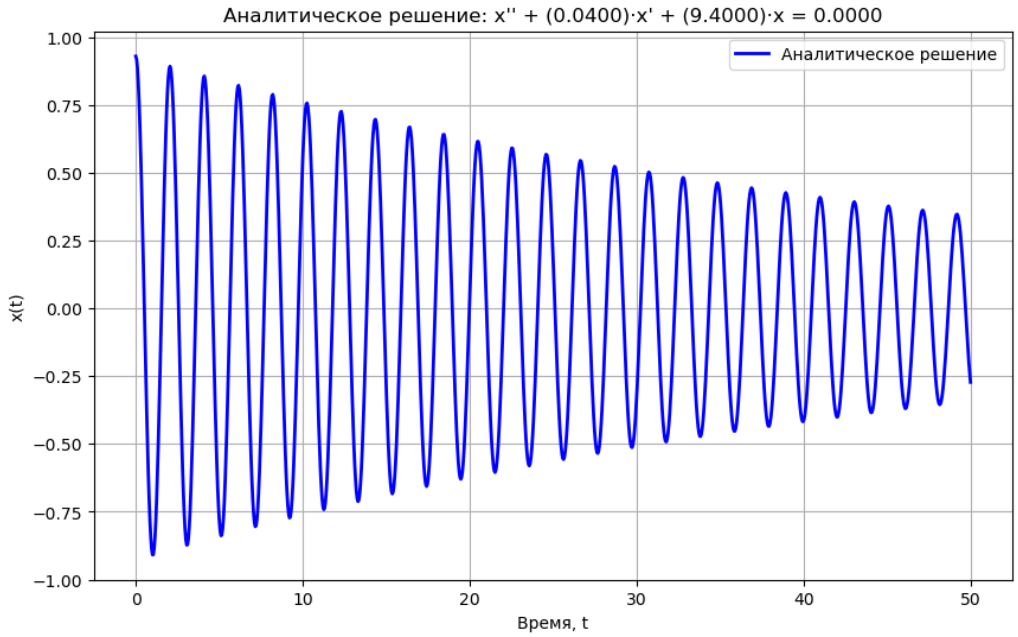
Аналитическое решение

Для решения аналитическим способом используем скрипт:

```
# λ² + B*λ + C = 0
D = B**2 - 4*C
lambda1 = (-B + np.sqrt(D)) / 2
A = m
B = b
C = k
D = 0

x0 = x_0 # x(0)
x0_dot = 0.0 # x'(0)

discriminant = B**2 - 4*C*A
lambda1 = (-B + np.sqrt(discriminant)) / 2
lambda2 = (-B - np.sqrt(discriminant)) / 2

x_particular = D / C

if discriminant > 0:
    A_matrix = np.array([[1, 1], [lambda1, lambda2]])
    b_vector = np.array([x0 - x_particular, x0_dot])
```

```
    C1, C2 = np.linalg.solve(A_matrix, b_vector)
    def analytical_solution(t):
        return C1 * np.exp(lambda1 * t) + C2 * np.exp(lambda2 * t) + x_particular
elif discriminant == 0:
    C1 = x0 - x_particular
    C2 = x0_dot - lambda1 * C1
    def analytical_solution(t):
        return (C1 + C2 * t) * np.exp(lambda1 * t) + x_particular
else:
    alpha = -B / 2
    beta = np.sqrt(-discriminant) / 2
    C1 = x0 - x_particular
    C2 = (x0_dot - alpha * C1) / beta
    def analytical_solution(t):
        return np.exp(alpha * t) * (C1 * np.cos(beta * t) + C2 * np.sin(beta * t)) + x_particular

t = np.linspace(0, 50, 1000)
x_analytical = analytical_solution(t)
```

Решение принимает вид:



Действительно, система стремится к равновесию. Сравнивая с численными методами, самым точным также остается RK4, показывающий наиболее приближенные значения состояния системы в каждый момент времени.

При аналитическом решении корни уравнения являются комплексно-сопряженными с отрицательной действительной частью, следствием чего и является затухающий колебательный характер системы.