ИТМО
МФКТУ
ФСУИР

Практическая работа №1

Дисциплина: Имитационное моделирование робототехнических систем

Студент: Воробьёв Роман, R4133c

Преподаватель: Ракшин Егор Александрович

Санкт-Петербург
2025

Дано: дифференциальное уравнение вида $a\ddot{x} + b\dot{x} + cx = d$,
где $a = 8.57, b = -6.64, c = -4.86, d = 6.65$.

1. Аналитическое решение

$$a\ddot{x} + b\dot{x} + cx = d$$

Для решения неоднородного ДУ сначала найдем общее решение – решение соответствующего ОДУ.

Решим характеристическое уравнение:

$$a\lambda^2 + b\lambda + c = 0$$

$$D = b^2 - 4ac = (-6{,}64)^2 + 4 \cdot 8{,}57 \cdot 4{,}86 = 210{,}6904$$

Дискриминант положителен, значит корни – действительные числа, а решение ОДУ имеет вид $x_{общ}(t) = C_1 e^{\lambda_1 t} + C_2 e^{\lambda_2 t}$,

$$\text{где } \begin{cases} \lambda_1 = \frac{-b-\sqrt{D}}{2a} = -0{,}4595 \\ \lambda_2 = \frac{-b+\sqrt{D}}{2a} = 1{,}2342 \end{cases}$$

Найдем частное решение:

$$x_ч = d/c = -1{,}368.$$

Полное решение будет иметь вид:

$$x(t) = x_{общ}(t) + x_ч = C_1 e^{-0{,}4595 \cdot t} + C_2 e^{1{,}2342 \cdot t} - 1{,}368.$$

C_1 и C_2 определяются начальными условиями.

Пусть $x(0) = 1, \dot{x}(0) = 1$, тогда

$$\begin{cases} C_1 + C_2 - 1{,}368 = 1, \\ -0{,}4595 \cdot C_1 + 1{,}2342 \cdot C_2 = 1 \end{cases} \Rightarrow \begin{cases} C_1 = 1{,}135 \\ C_2 = 1{,}233. \end{cases}$$

2. Численное решение

Так как явный и неявный методы Эйлера и метод Рунге-Кутта позволяют решать уравнения вида $\dot{y} = f(x, y)$, необходимо привести наше уравнение 2 порядка к системе из 2 уравнений такого вида.

Для этого введем переменные $y = x, z = \dot{x}$. Система будет иметь вид:

$$\begin{cases} \dot{y} = z, \\ \dot{z} = \dfrac{d - bz - cy}{a}. \end{cases}$$

При этом численное решение на каждом шаге формируется для обоих уравнений системы.

Для реализации этого были немного изменены функции методов численного решения: на вход функции f() для вычисления производной переменных передаются 2 переменные. Сама функция f() реализует вычисление по выражениям приведенной выше системы.

3. Анализ результатов

На рисунках 1 и 2 представлены функции, описывающие решения каждым из 4 способов: аналитическим, явным и неявным методами Эйлера и методом Рунге-Кутта.

Видно, что чем дальше идет решение уравнения от начальной точки t = 0, тем больше расхождение решения каждого из численных методов с аналитическим решением. Это объясняется накоплением ошибки, так как решение на предыдущем шаге определяет решение на следующем.

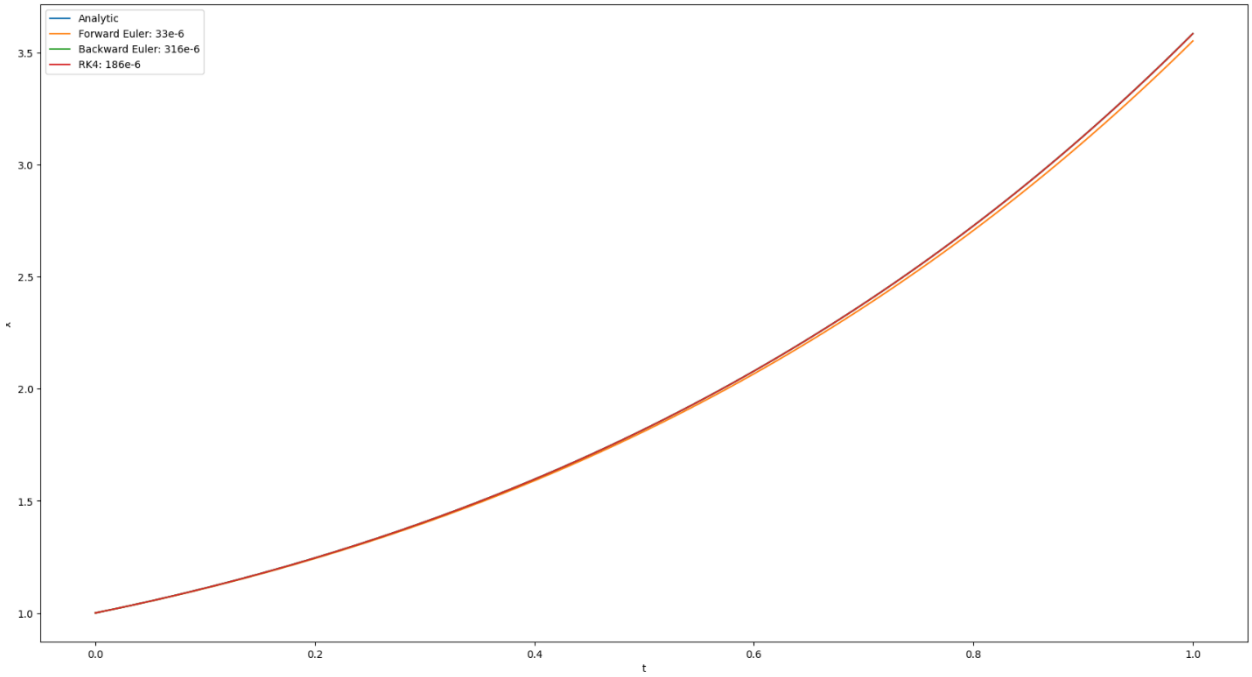

Рисунок 1 – Функция x(t) для 4 способов решения ДУ: аналитического, явного и неявного методов Эйлера и метода Рунге-Кутта

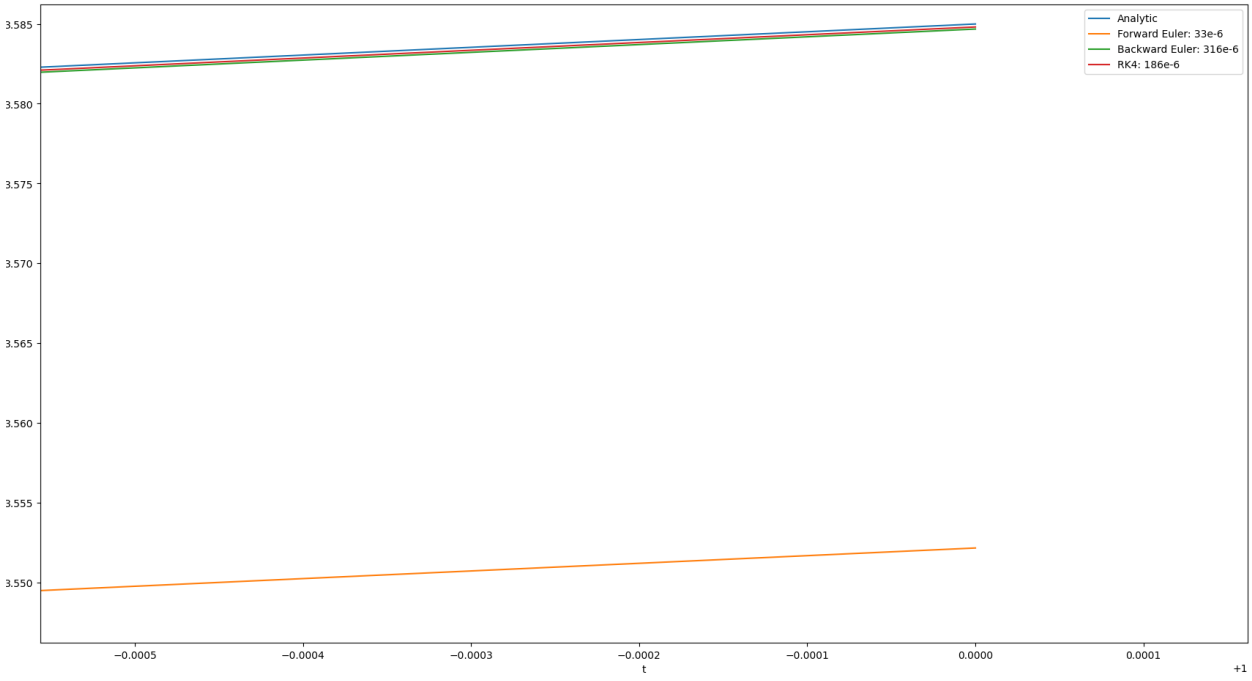

Рисунок 2 – Функция x(t) в конечной точке t = 1

По расчету ошибок численных решений, представленному на рисунке 3, видно, что явный метод Эйлера имеет точность на 2 порядка меньше, чем у неявного метода Эйлера и метода Рунге-Кутта.

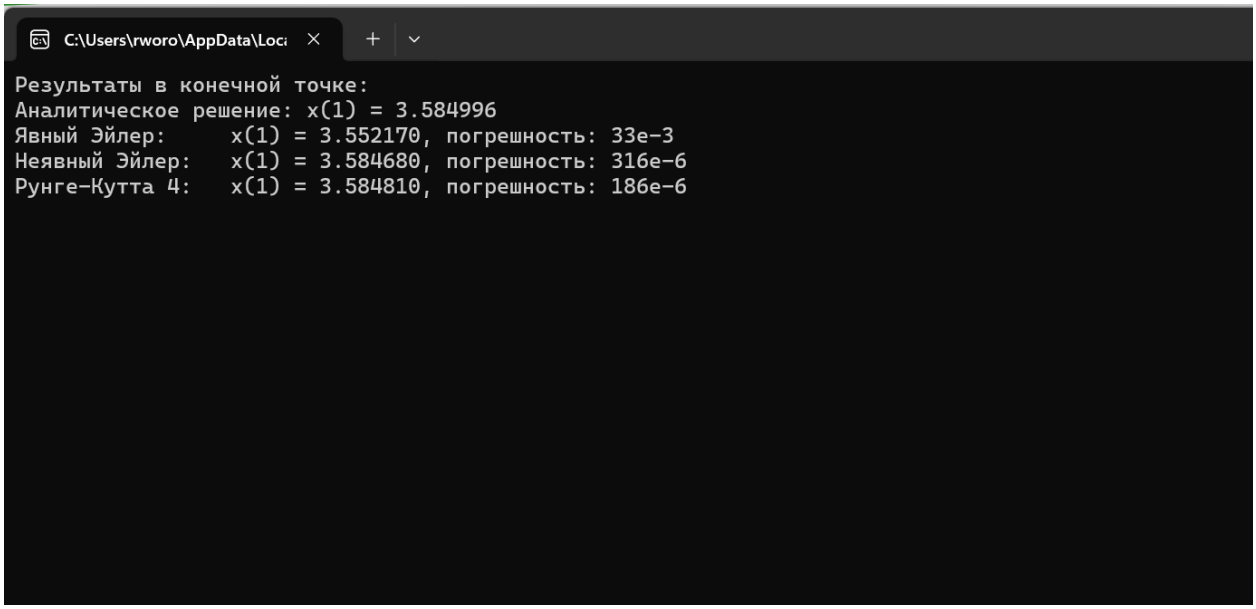

Рисунок 3 – Решение при t = 1 и погрешности численных методов

Здесь – на сравнительно небольшом отрезке поиска решения неявный метод Эйлера и метод Рунге-Кутта имеют одинаковый порядок точности. Однако, если рассмотреть решение на отрезке от 0 до 10, расхождение между методами увеличивается.

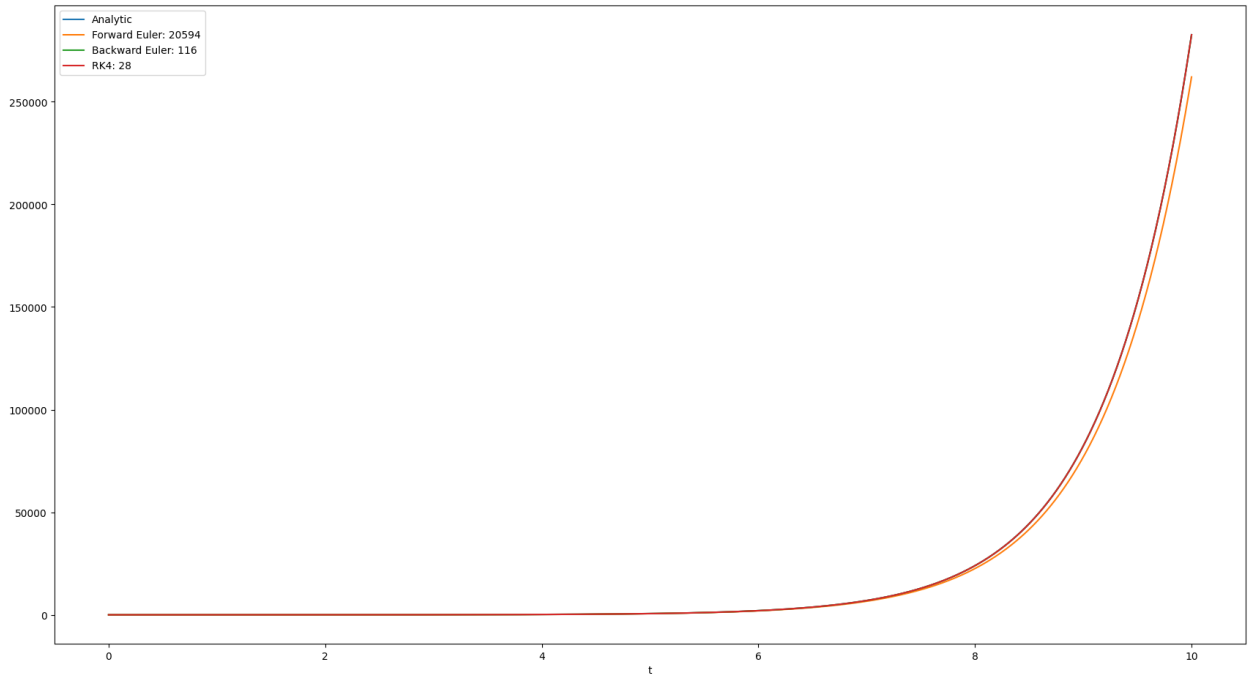

Рисунок 4 – Функция x(t) на отрезке от 0 до 10

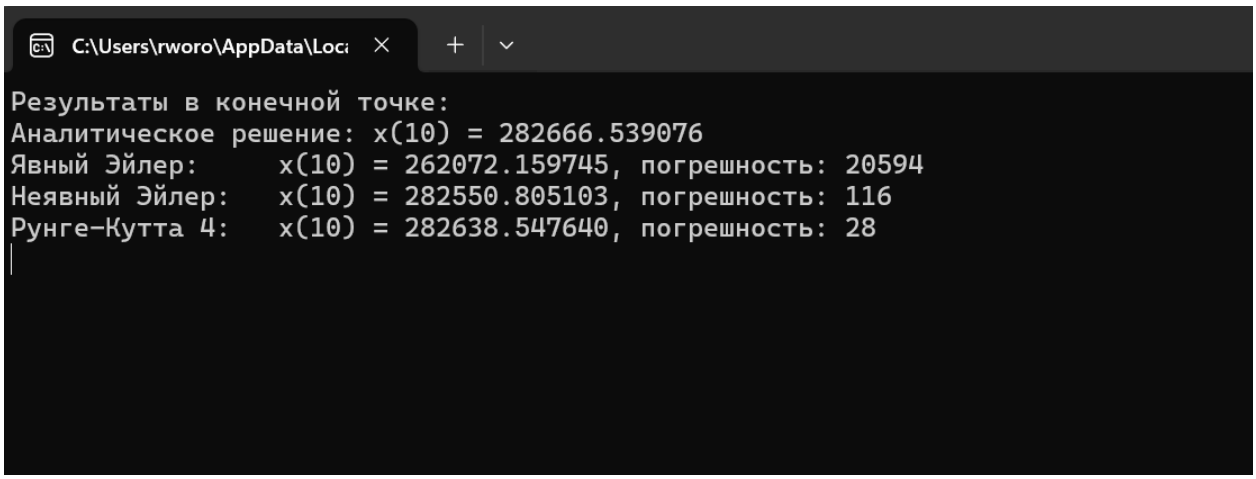

Рисунок 5 – Решение при t = 10 и погрешности численных методов

4. Выводы

Сравнение точности решения ДУ 2 порядка численными методами показывает, что наиболее точным является метод Рунге-Кутта, на порядок меньше точность неявного метода Эйлера и на 3 порядка меньше точность явного метода Эйлера, который ожидаемо является наименее точным.

Радикальное повышение точности происходит благодаря качественному изменению подхода. В отличие от явного метода Эйлера, в неявном методе Эйлера и методе Рунге-Кутта осуществляется расчет приращений на нескольких отрезках внутри одного шага. Это достигается за счет вычисления «предположительного» решения на следующем шаге и расчета производной в промежуточных точках.

Также точность повышается при увеличении числа промежуточных точек, за счет чего метод Рунге-Кутта оказывается точнее неявного метода Эйлера.

5. Код

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


def forward_euler(fun, x0, Tf, h):
    """
    Explicit Euler integration method
    """
    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(t[k], x_hist[:, k])

    return x_hist, t

def backward_euler(fun, x0, Tf, h):
    """
    Implicit Euler integration method using fixed-point iteration
    """
    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(t[k] + 0.5*h, x_hist[:, k] +
0.5*h*fun(t[k], x_hist[:, k]))  # Initial guess

    return x_hist, t


def runge_kutta4(fun, x0, Tf, h):
    """
    4th order Runge-Kutta integration method
    """
    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(t[k], x_hist[:, k])
        k2 = fun(t[k] + 0.5*h, x_hist[:, k] + 0.5 * h * k1)
        k3 = fun(t[k] + 0.5*h, x_hist[:, k] + 0.5 * h * k2)
        k4 = fun(t[k] + h, 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

#Analytic solution
def analitic_solution(Tf, h):
    t = np.arange(0, Tf + h, h)
    D = b**2 - 4*a*c
    p1 = 0.5*(-b - np.sqrt(D))/a
    p2 = 0.5*(-b + np.sqrt(D))/a
    return 1.135*np.exp(p1*t) + 1.233*np.exp(p2*t) - 1.368, t


a = 8.57
b = -6.64
```

```
c = -4.86
d = 6.65

#Solution parameters
Tf = 10.0
h = 0.01
y0 = 1.0
z0 = 1.0

#Function for system of DEs
# y' = z
# z' = (d - b*z - c*y)/a

def f(x, y_vec):
    y, z = y_vec
    return np.array([z, (d - b*z - c*y)/a])


x0 = np.array([y0, z0])

x_an, t_an = analitic_solution(Tf, h)

# Forward Eiler
x_fe, t_fe = forward_euler(f, x0, Tf, h)

# Backward Eiler
x_be, t_be = backward_euler(f, x0, Tf, h)

# Runge-Kutta
x_rk4, t_rk4 = runge_kutta4(f, x0, Tf, h)

fe_err = np.abs(x_fe[0, :] - x_an)
be_err = np.abs(x_be[0, :] - x_an)
rk4_err = np.abs(x_rk4[0, :] - x_an)

print("Результаты в конечной точке:")
print(f"Аналитическое решение: x({10}) = {x_an[-1]:.6f}")
print(f"Явный Эйлер:     x({10}) = {x_fe[0, -1]:.6f}, погрешность: {fe_err[-
1]:.0f}")
print(f"Неявный Эйлер:   x({10}) = {x_be[0, -1]:.6f}, погрешность: {be_err[-
1]:.0f}")
print(f"Рунге-Кутта 4:   x({10}) = {x_rk4[0, -1]:.6f}, погрешность: {rk4_err[-
1]:.0f}")

# Plot results
plt.figure(figsize=(24, 8))

plt.plot(t_an, x_an, label=f'Analytic')
plt.plot(t_fe, x_fe[0, :], label=f'Forward Euler: {fe_err[-1]:.0f}')
plt.plot(t_be, x_be[0, :], label=f'Backward Euler: {be_err[-1]:.0f}')
plt.plot(t_rk4, x_rk4[0, :], label=f'RK4: {rk4_err[-1]:.0f}')
plt.xlabel('t')
plt.ylabel('x')
plt.legend()

plt.tight_layout()
plt.show()
```