Национальный исследовательский университет ИТМО
Факультет систем управления и робототехники

Отчет
о выполнении практического задания №1
по дисциплине «Имитационное моделирование робототехнических систем»

Выполнил
студент гр. R4134c
ИСУ 505887 А. С. Абраменко

Преподаватель Е. А. Ракшин

«___» __________ 2025 г.

Санкт-Петербург

2025

Задание

1. Решить аналитически ОДУ в виде:

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

Коэффициенты ОДУ в соответствии с таблицей:

a	b	c	d
-0,03	-3,04	2,87	-1,58

2. Из списка возьмите коэффициенты вашего ОДУ и решите их с помощью трех интеграторов: явный/неявный метод Эйлера, метод Рунга-Кутты.

3. Сравните результаты этих методов с аналитическим решением.

Ход работы

1. Аналитическое решение ОДУ.

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

Поделим на a и переобозначим:

$$\ddot{x} + p\dot{x} + qx = r,$$

где $p = \frac{b}{a} = \frac{-3,04}{-0,03}$,

$q = \frac{c}{a} = \frac{2,87}{-0,03}$,

$r = \frac{d}{a} = \frac{-1,58}{-0,03}$

Решим однородное уравнение с помощью решения характеристического уравнения:

$$\ddot{x} + p\dot{x} + qx = 0$$

$$\lambda^2 + p\lambda + q = 0$$

Корни характеристического уравнения:

$$\lambda_{1,2} = \frac{-p \pm \sqrt{p^2 - 4q}}{2},$$

где $p^2 - 4q \approx\ 10651.1 > 0$

Тогда:

$$\lambda_1 = 0{,}9315$$
$$\lambda_2 = -102{,}2645$$

Получаем:

$$x_h(t) = C_1 e^{\lambda_1 t} + C_2 e^{\lambda_2 t}$$

Поиск частного решения:

$$c \neq 0 => x_p = \frac{r}{q}$$

Общее решение:

$$x = x_h + \ x_p = C_1 e^{\lambda_1 t} + C_2 e^{\lambda_2 t} + \frac{r}{q}$$
$$x = C_1 e^{0{,}9315t} + C_2 e^{-102{,}2645t} - 0{,}5504$$

Поиск констант C_1 и C_2 из н.у. $x(0) = 0, \dot{x}(0) = 0$:

$$\begin{cases} C_1 + C_2 = 0{,}6504 \\ 0{,}9315C_1 - 102{,}2645C_2 = 0 \end{cases}$$
$$\begin{cases} C_1 = 0{,}6445 \\ C_2 = 0{,}0059 \end{cases}$$

2. Реализация интеграторов и сравнительная характеристика

Реализация интегральных методов явного и неявного Эйлера, а также метода Рунге-Кутты представлена в приложении. Программа также содержит найденное ранее аналитическое решение, а также расчет СКО интегральных методов относительного аналитического решения.

В данном случае система содержит компоненты с очень разными временными масштабами, что можно увидеть по значениям $\lambda_1 = 0{,}9315$ и $\lambda_2 = -102{,}2645$. При данных значениях отношение временных масштабов:

$$\frac{102{,}2645}{0{,}9315} = 110,$$

что указывает на высокую жесткость системы – процессы имеют сильно различающиеся скорости. Медленная экспоненциальная компонента система изменяется за секунды, в то время как быстрая экспоненциальная компонента изменяется за сотые доли секунды. Данное различие скоростей процессов создает проблемы для численного интегрирования.

Явный метод Эйлера для быстрой затухающей компоненты создает дополнительное затухание. Решение получается гладким, но обычно проигрывает в точности другим методам.

В реализации метода неявного Эйлера используется метод простых итераций. Для сходимости итерационного процесса необходимо выполнение условия сходимости:

$$|h\lambda| < 1$$

Критический шаг для сходимости в данной системе:

$$h_{\text{кр}} = \frac{1}{102{,}2645} = 0{,}00978$$

Таким образом, при $h > 0{,}00978$ итерационный процесс не сходится к решению, а начинает расходиться. Для того, чтобы расхождения не происходило, необходимо использовать исключительно малые шаги итераций.

Метод Рунге-Кутты является еще более чувствительным к жесткости системы. Данный метод точно отслеживает все компоненты системы, но при недостаточно малом шаге это может приводить к накоплению ошибок. Критический шаг для метода Рунге-Кутты:

$$h|\lambda| < 2{,}8$$

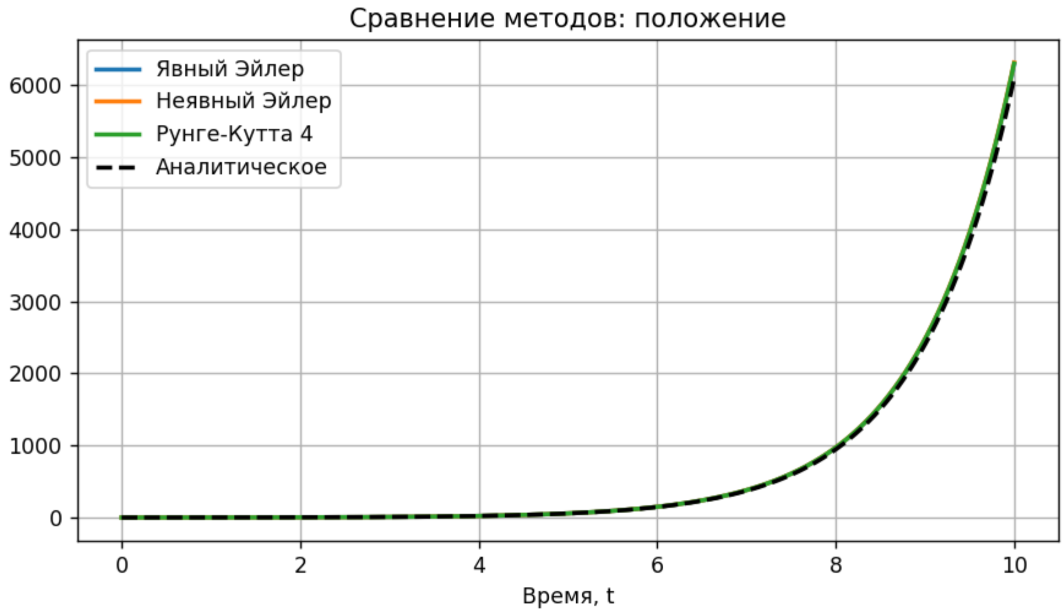
$$h_{\text{кр}} = \frac{2{,}8}{102{,}2645} = 0.0274$$

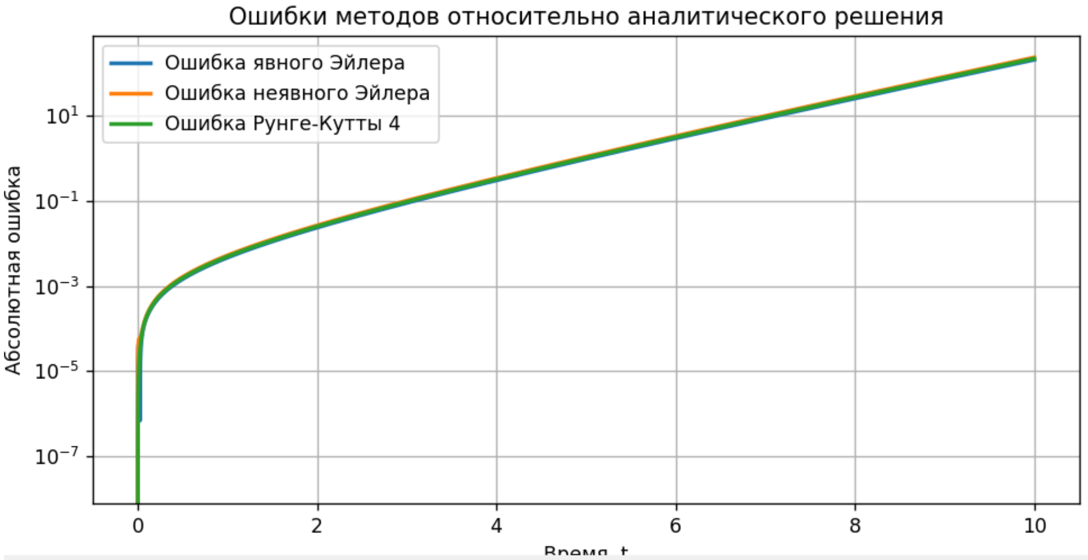
Таким образом, для данной системы рациональным решением может быть использование малых шагов интегрирования, а также использование специализированных методов интегрирования для жестких систем.

3. Результаты тестирований

1. При $T = 10$, $h = 0{,}0005$:

```
Среднеквадратичные ошибки:
Явный Эйлер: 4.404623e+01
Неявный Эйлер: 5.009016e+01
Рунге-Кутта 4: 4.706411e+01
```

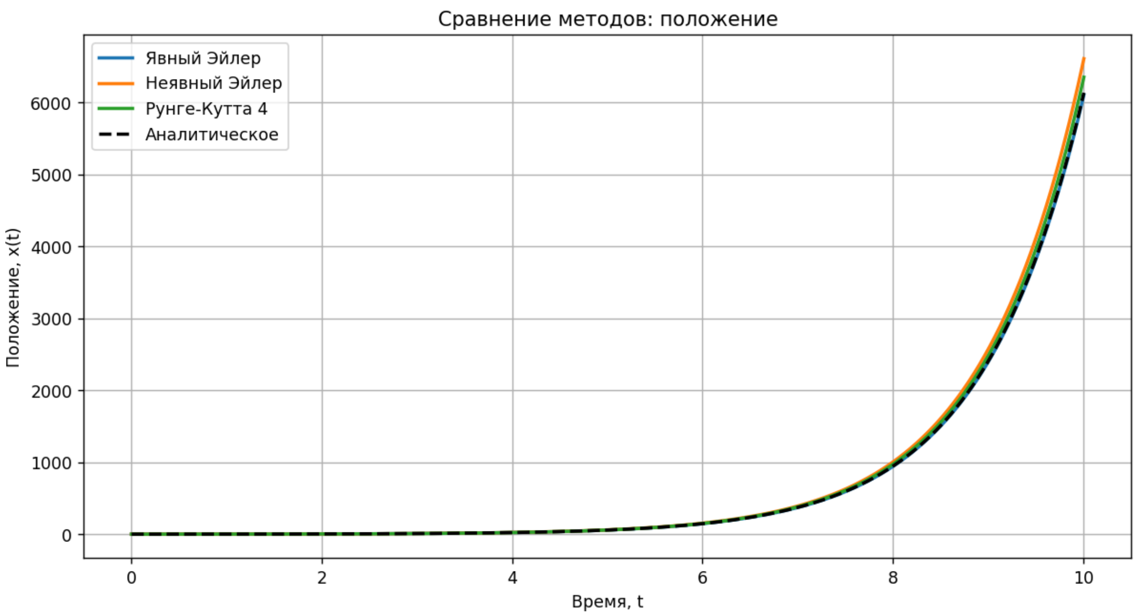


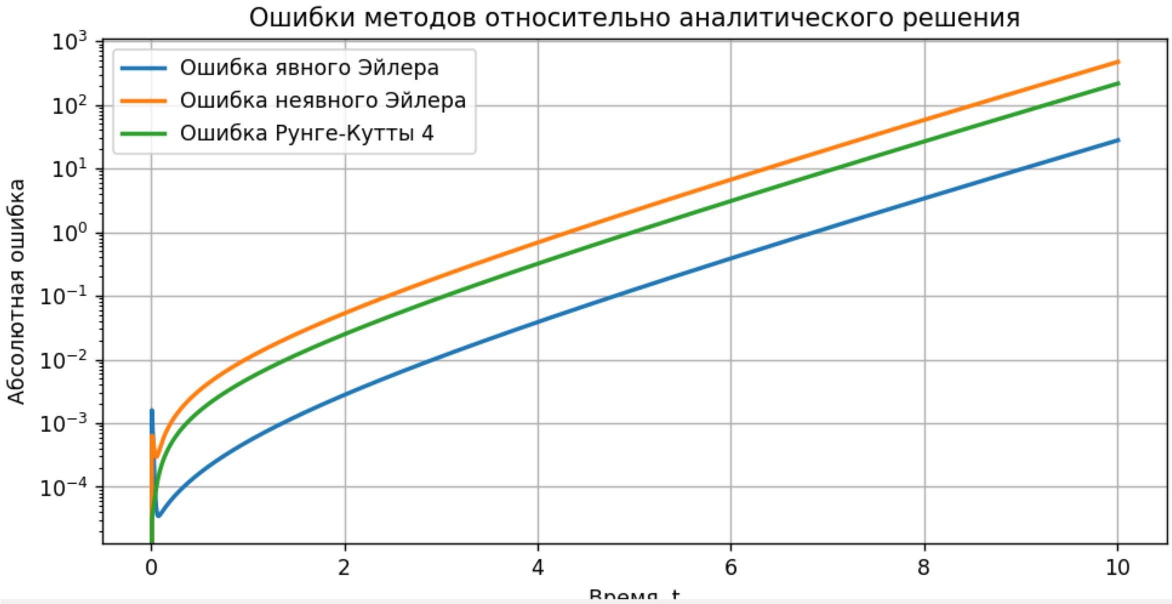


При данном значении шага все методы устойчивы, но из-за высокой жесткости системы неявный Эйлер и метод Рунге-Кутты показывают более высокие значения ошибки, сравнимые с ошибкой метода явного Эйлера.

2. При $T = 10$, $h = 0{,}009$:

```
Среднеквадратичные ошибки:
Явный Эйлер: 6.104144e+00
Неявный Эйлер: 1.040436e+02
Рунге-Кутта 4: 4.762697e+01
```



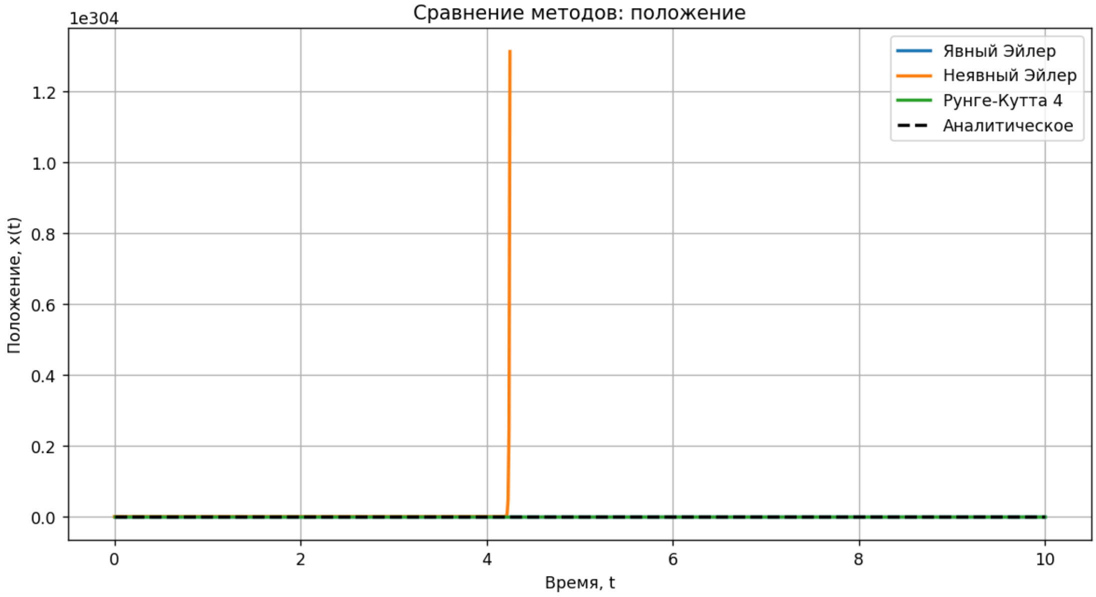


При данном критическом для метода неявного Эйлера значении шага все методы также остаются устойчивыми, но ошибка для неявного Эйлера значительно увеличена, а для явного Эйлера – уменьшена. Последнее может быть связано с высокой скоростью затухания быстрых компонент в жестких системах: при данном значении шага быстрая компонента подавляется за один шаг. Соответственно, происходит быстрый переход к установившемуся режиму.

3. При $T = 10$, $h = 0{,}01$:

```
Среднеквадратичные ошибки:
Явный Эйлер: 1.182688e+01
```

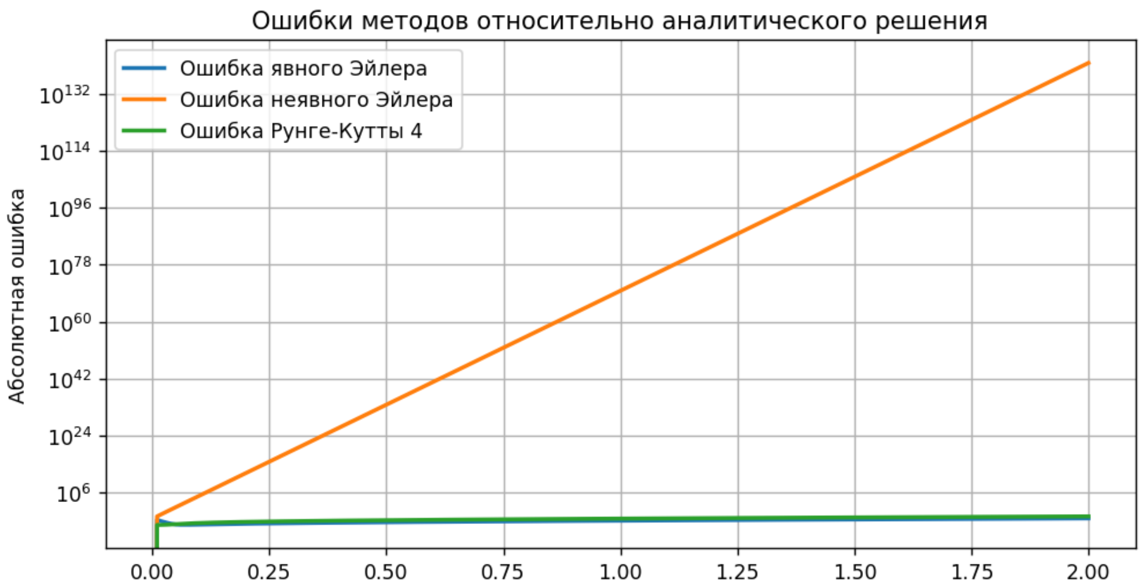
```
Неявный Эйлер: nan
Рунге-Кутта 4: 4.727395e+01
```



При данном значении шага, превышающем критическое значение для неявного Эйлера, его использование приводит к нежелательным результатам. Значение ошибки для метода Рунге-Кутты также немного увеличено, но при этом метод обеспечивает сходимость решения.

При попытке превысить критическое значение шага для метода Рунге-Кутты возникла ошибка в системе, что также подтверждает неприменимость метода для жестких систем с большим шагом интегрирования.

Наглядный график роста СКО для неявного Эйлера при $T = 2$, $h = 0{,}01$:



Выводы

По результатам выполненной работы был сделан вывод о том, что применимость различных методов интегрирования значительно зависит от жесткости системы.

Высокая скорость затухания быстрой компоненты позволила получить наиболее точный результат при использовании метода явного Эйлера, однако для методов неявного Эйлера и Рунге-Кутты, которые для традиционных систем дают более точные решения, разность скоростей процессов оказалась слишком велика. В результате применимость метода неявного Эйлера оказалась очень ограниченной, а метод Рунге-Кутты показал результат, точность которого не оправдывает вычислительные затраты и не превышает точности метода явного Эйлера.

Таким образом, при высокой жесткости системы необходимо использовать специализированные методы интегрирования или малые шаги интегрирования. Для данной в работе системы использование традиционных методов интегрирования нежелательно.

ПРИЛОЖЕНИЕ

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

# Коэффициенты ОДУ: a*x'' + b*x' + c*x = d
a = -0.03
b = -3.04
c = 2.87
d = -1.58

def system_dynamics(x):

    #Динамика системы для ОДУ: a*x'' + b*x' + c*x = d

    x1 = x[0]  # положение
    x2 = x[1]  # скорость

    x1_dot = x2  # x1' = x2
    x2_dot = (d - b*x2 - c*x1) / a  # x2' = x''

    return np.array([x1_dot, x2_dot])

def analytical_solution(t, x0):
```

```python
    """
    Аналитическое решение ОДУ
    x(t) = C1*exp(0.932*t) + C2*exp(-102.265*t) - 0.5504
    """
    A = 0.932
    B = -102.265
    C = -0.5504

    matrix = np.array([[1, 1], [A, B]])
    rhs = np.array([x0[0] - C, x0[1]])
    C1, C2 = np.linalg.solve(matrix, rhs)

    return C1*np.exp(A*t) + C2*np.exp(B*t) + C

# Функции интеграторов
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

# Параметры интегрирования
x0 = np.array([0.0, 0.0])  # н.у.
Tf = 10.0  # Время
h = 0.001  # Шаг

# Численное интегрирование
x_fe, t_fe = forward_euler(system_dynamics, x0, Tf, h)
x_be, t_be = backward_euler(system_dynamics, x0, Tf, h)
x_rk4, t_rk4 = runge_kutta4(system_dynamics, x0, Tf, h)

# Аналитическое решение
x_analytical = analytical_solution(t_fe, x0)

# Визуализация результатов
plt.figure(figsize=(20, 12))

# График положения
plt.subplot(2, 2, 1)
plt.plot(t_fe, x_fe[0, :], label='Явный Эйлер', linewidth=2)
plt.plot(t_be, x_be[0, :], label='Неявный Эйлер', linewidth=2)
plt.plot(t_rk4, x_rk4[0, :], label='Рунге-Кутта 4', linewidth=2)
plt.plot(t_fe, x_analytical, 'k--', label='Аналитическое', linewidth=2)
plt.xlabel('Время, t')
plt.ylabel('Положение, x(t)')
plt.legend()
plt.title('Сравнение методов: положение')
plt.grid(True)

# График скорости
plt.subplot(2, 2, 2)
plt.plot(t_fe, x_fe[1, :], label='Явный Эйлер', linewidth=2)
plt.plot(t_be, x_be[1, :], label='Неявный Эйлер', linewidth=2)
plt.plot(t_rk4, x_rk4[1, :], label='Рунге-Кутта 4', linewidth=2)
plt.xlabel('Время, t')
plt.ylabel('Скорость, dx/dt')
plt.legend()
plt.title('Сравнение методов: скорость')
plt.grid(True)

# Фазовый портрет
plt.subplot(2, 2, 3)
plt.plot(x_fe[0, :], x_fe[1, :], label='Явный Эйлер', linewidth=2)
plt.plot(x_be[0, :], x_be[1, :], label='Неявный Эйлер', linewidth=2)
plt.plot(x_rk4[0, :], x_rk4[1, :], label='Рунге-Кутта 4', linewidth=2)
```

```python
plt.xlabel('Положение, x')
plt.ylabel('Скорость, dx/dt')
plt.legend()
plt.title('Фазовый портрет')
plt.grid(True)

# Ошибки методов относительно аналитического решения
plt.subplot(2, 2, 4)
error_fe = np.abs(x_fe[0, :] - x_analytical)
error_be = np.abs(x_be[0, :len(t_fe)] - x_analytical[:len(t_be)])
error_rk4 = np.abs(x_rk4[0, :] - x_analytical)

plt.semilogy(t_fe, error_fe, label='Ошибка явного Эйлера', linewidth=2)
plt.semilogy(t_be, error_be, label='Ошибка неявного Эйлера', linewidth=2)
plt.semilogy(t_rk4, error_rk4, label='Ошибка Рунге-Кутты 4', linewidth=2)
plt.xlabel('Время, t')
plt.ylabel('Абсолютная ошибка')
plt.legend()
plt.title('Ошибки методов относительно аналитического решения')
plt.grid(True)

plt.tight_layout()
plt.show()

# Вывод среднеквадратичных ошибок
print("Среднеквадратичные ошибки:")
print(f"Явный Эйлер: {np.sqrt(np.mean(error_fe**2)):.6e}")
print(f"Неявный Эйлер: {np.sqrt(np.mean(error_be**2)):.6e}")
print(f"Рунге-Кутта 4: {np.sqrt(np.mean(error_rk4**2)):.6e}")
```