**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

Федеральное государственное бюджетное образовательное учреждение   
высшего образования

**«Сибирский государственный университет науки и технологий   
имени академика М.Ф. Решетнева»**

Институт информатики и телекоммуникаций

Кафедра информатики и вычислительной техники

**ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ**

Вычислительная математика

|  |
| --- |
| Лабораторная работа 6 |

Руководитель Сенашов А. В.

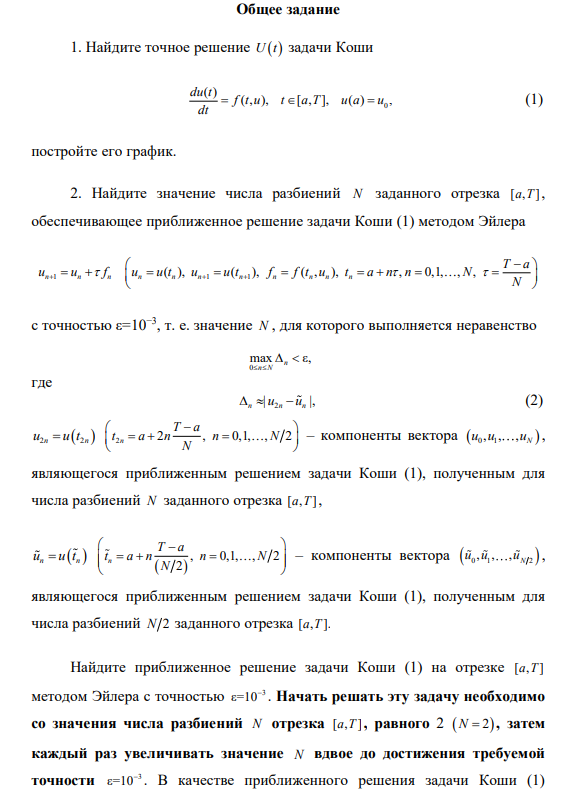
подпись, дата инициалы, фамилия

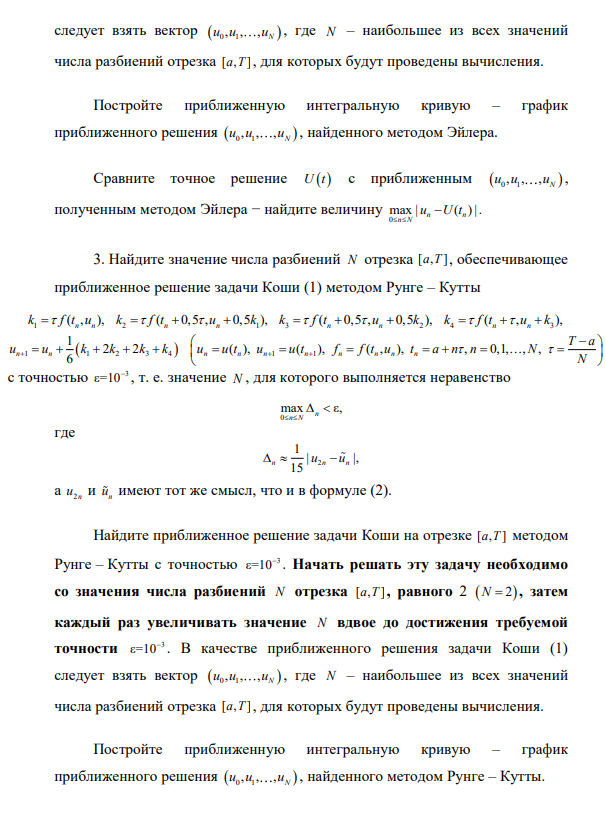
Обучающийся БПИ22-02, 221219040 К. В. Трифонов

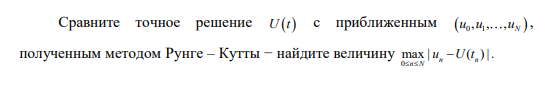
номер группы, зачетной книжки подпись, дата инициалы, фамилия

Красноярск 2023 г

# ХОД РАБОТЫ







*Вариант 27:*



import numpy as np

import matplotlib.pyplot as plt

from scipy.integrate import solve\_ivp

# Задание параметров

a = 1

T = 2

u0 = 1

epsilon = 1e-3

# Функция для задачи Коши

def f(t, u):

    return (u\*\*2 \* np.log(t) - u) / t

# Точное решение

def exact\_solution(t):

    sol = solve\_ivp(f, [a, T], [u0], t\_eval=t, method='RK45', rtol=1e-10, atol=1e-10)

    return sol.y[0]

# Метод Эйлера

def euler\_method(a, T, u0, N):

    tau = (T - a) / N

    t\_values = np.linspace(a, T, N + 1)

    u\_values = np.zeros(N + 1)

    u\_values[0] = u0

    for n in range(N):

        u\_values[n + 1] = u\_values[n] + tau \* f(t\_values[n], u\_values[n])

    return t\_values, u\_values

# Метод Рунге-Кутты 4-го порядка

def runge\_kutta\_method(a, T, u0, N):

    tau = (T - a) / N

    t\_values = np.linspace(a, T, N + 1)

    u\_values = np.zeros(N + 1)

    u\_values[0] = u0

    for n in range(N):

        k1 = tau \* f(t\_values[n], u\_values[n])

        k2 = tau \* f(t\_values[n] + tau/2, u\_values[n] + k1/2)

        k3 = tau \* f(t\_values[n] + tau/2, u\_values[n] + k2/2)

        k4 = tau \* f(t\_values[n] + tau, u\_values[n] + k3)

        u\_values[n + 1] = u\_values[n] + (k1 + 2\*k2 + 2\*k3 + k4) / 6

    return t\_values, u\_values

# Функция для нахождения N с заданной точностью

def find\_optimal\_N(method, epsilon):

    N = 2

    while True:

        t\_values, u\_values = method(a, T, u0, N)

        t\_values\_2, u\_values\_2 = method(a, T, u0, 2 \* N)

        delta = max(abs(u\_values\_2[::2] - u\_values))

        if delta < epsilon:

            break

        N \*= 2

    return N, t\_values, u\_values

# Нахождение оптимального N для метода Эйлера

N\_euler, t\_values\_euler, u\_values\_euler = find\_optimal\_N(euler\_method, epsilon)

print(f"Оптимальное N для метода Эйлера: {N\_euler}")

# Нахождение оптимального N для метода Рунге-Кутты

N\_rk, t\_values\_rk, u\_values\_rk = find\_optimal\_N(runge\_kutta\_method, epsilon)

print(f"Оптимальное N для метода Рунге-Кутты: {N\_rk}")

# Вычисление точного решения

t\_exact = np.linspace(a, T, 1000)

u\_exact = exact\_solution(t\_exact)

# Интерполяция численных решений для сравнения с точным решением

u\_euler\_interp = np.interp(t\_exact, t\_values\_euler, u\_values\_euler)

u\_rk\_interp = np.interp(t\_exact, t\_values\_rk, u\_values\_rk)

# Вычисление максимального расхождения

max\_diff\_euler = np.max(np.abs(u\_exact - u\_euler\_interp))

max\_diff\_rk = np.max(np.abs(u\_exact - u\_rk\_interp))

print(f"Максимальное расхождение для метода Эйлера: {max\_diff\_euler}")

print(f"Максимальное расхождение для метода Рунге-Кутты: {max\_diff\_rk}")

# Построение графиков

plt.figure(figsize=(12, 6))

plt.plot(t\_exact, u\_exact, label="Точное решение", linewidth=2)

plt.plot(t\_values\_euler, u\_values\_euler, 'o-', label="Метод Эйлера")

plt.plot(t\_values\_rk, u\_values\_rk, 's-', label="Метод Рунге-Кутты")

plt.xlabel("t")

plt.ylabel("u(t)")

plt.legend()

plt.title("Сравнение точного и численных решений задачи Коши")

plt.grid(True)

plt.show()

Результат:

Оптимальное N для метода Эйлера: 256

Оптимальное N для метода Рунге-Кутты: 2

Максимальное расхождение для метода Эйлера: 0.0013366856253342885

Максимальное расхождение для метода Рунге-Кутты: 0.03882693355570166