Санкт-Петербургский политехнический университет
Высшая школа прикладной математики и вычислительной физики, ФизМех

Направление подготовки
«01.03.02 Прикладная математика и информатика»
Специальность «Биоинформатика»

Лабораторная работа №8
**тема "Решение задачи Коши"**

Выполнила студентка гр. 5030102/10401:

Кузнецова Юлия

Преподаватель:

Козлов Константин Николаевич

Санкт-Петербург

2024

# Содержание

# Формулировка задания

Необходимо создать программу, реализующую метод Хёйна для решения задачи Коши для систем обыкновенных дифференциальных уравнений (ОДУ) произвольной размерности. Программа должна позволять пользователю задавать функцию-правую часть системы ОДУ и начальные условия.

Программа должна вычислять решение ОДУ с пошаговым контролем точности по правилу Рунге. Если на текущем шаге точность не достигнута, то шаг уменьшается вдвое. Если достигнутая погрешность меньше заданной в 64 раза, то шаг увеличивается вдвое.

По полученным результатам программа должна построить следующие графики с использованием библиотеки Matplotlib:

- Изменение шага по отрезку для разных значений заданной точности
- Зависимость минимального шага от заданной точности
- Зависимость числа шагов от заданной точности
- Решение ОДУ для разных значений заданной точности

# Алгоритм метода Хойна

Параметры:

t: текущее время
v: текущее значение вектора решения
h: шаг интегрирования
call_counter - Счетчик вызовов функции fs

1. Вычислить коэффициенты Хёйна:

```
k1 = fs(t, v, call_counter)
k2 = fs(t + h, v + h * k1, call_counter)
```

2. Вычислить промежуточное значение:

```
v1 = v + (h / 2) * (k1 + k2)
```

3. Вычислить дополнительные коэффициенты:

```
k2 = fs(t + h / 2, v + h / 2 * k1, call_counter)
```

4. Вычислить следующее промежуточное значение:

```
v2 = v + (h / 4) * (k1 + k2)
```

5. Вычислить окончательное значение шага Хойна:

```
v2 = heun_step(t + h / 2, v2, h / 2)
```

6. Оценить ошибку:

```
error = np.linalg.norm(v2 - v1) / 3
```

7. Адаптировать шаг:

```
Если error > tolerance: step /= 2
```

```
Если error < tolerance / 64: step *= 2
```

8. Сделать шаг, если ошибка приемлема:

```
t += step
v = v1
Вывести решение
```

## Запуск функции

Для решения системы необходимо активизировать скрипт main.py
Командой: python main.py. Результат выводится в консоль.

Вывод организован в виде столбцов, где каждая строка соответствует одному шагу интегрирования:

1. Значение t.
2. Значение шага h.
3. Оценка по методу Рунге R.
4. Истраченное количество вычислений правой части N.
5. Значения функций-решений.

## Пример выполнения программы

```
1.5
2.5
0.1
10000
0.0001
3
def fs(t, v, kounter):
#
#
  A = np.array([[-0.4, 0.02, 0], [0, 0.8, -0.1], [0.003, 0, 1]])
  kounter[0] += 1
  return np.dot(A, v)
1 1 2
    1.500000     0.100000            0           0     1.000000     1.000000     2.000000
    1.600000      0.100000   8.45154e-05           5     0.962820     1.061398     2.210309
    1.700000      0.100000   9.33737e-05          10     0.927221     1.125613     2.442690
    1.750000      0.050000   1.28171e-05          20     0.909992     1.158775     2.568019
    1.800000      0.050000   1.34742e-05          25     0.893138     1.192634     2.699768
    1.850000      0.050000   1.41654e-05          30     0.876652     1.227187     2.838267
    1.900000      0.050000   1.48925e-05          35     0.860527     1.262426     2.983862
    1.950000      0.050000   1.56573e-05          40     0.844756     1.298342     3.136916
    2.000000      0.050000   1.64617e-05          45     0.829333     1.334924     3.297812
    2.050000      0.050000   1.73079e-05          50     0.814253     1.372157     3.466951
    2.100000      0.050000   1.81979e-05          55     0.799508     1.410026     3.644757
    2.150000      0.050000   1.91341e-05          60     0.785092     1.448511     3.831672
    2.200000      0.050000   2.01189e-05          65     0.771001     1.487590     4.028165
    2.250000      0.050000   2.11548e-05          70     0.757227     1.527236     4.234726
```

# Графики

Рисунок 1.  График зависимости числа шагов от точности

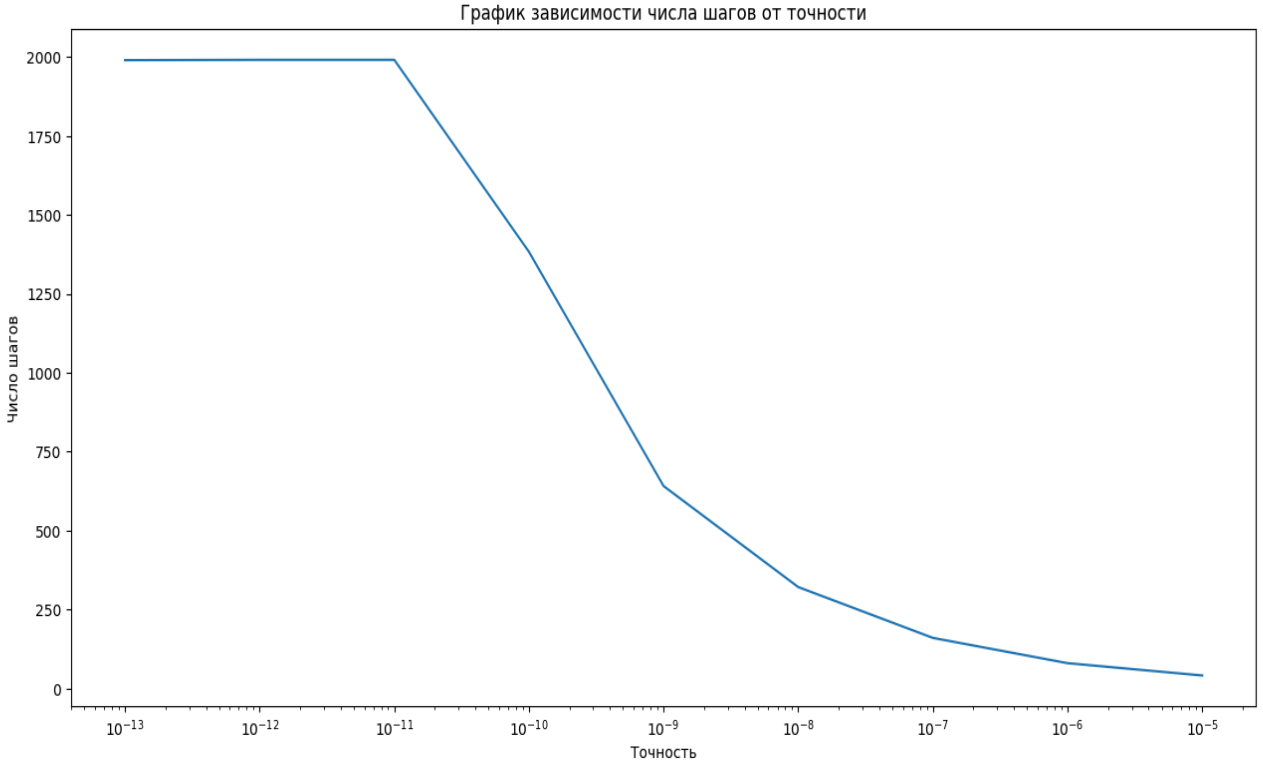


Рисунок 2. Зависимость минимального шага от точности

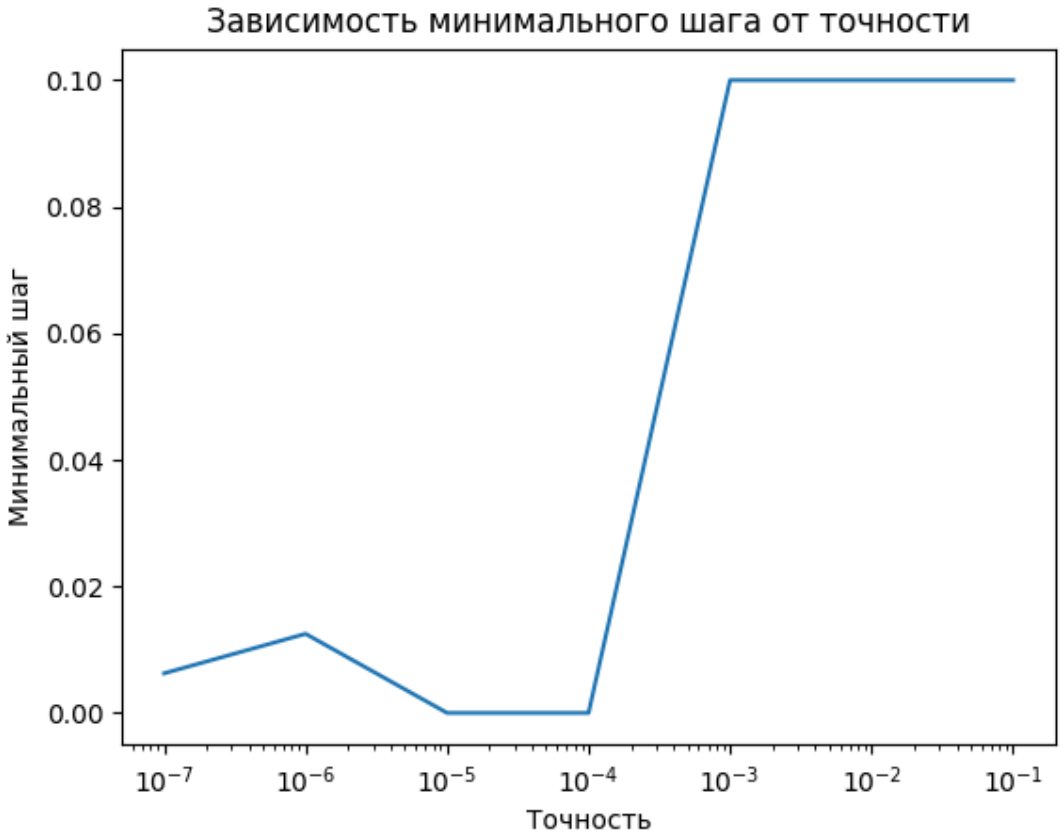

Рисунок 3. Изменение шага по отрезку для разных значений заданной точностью

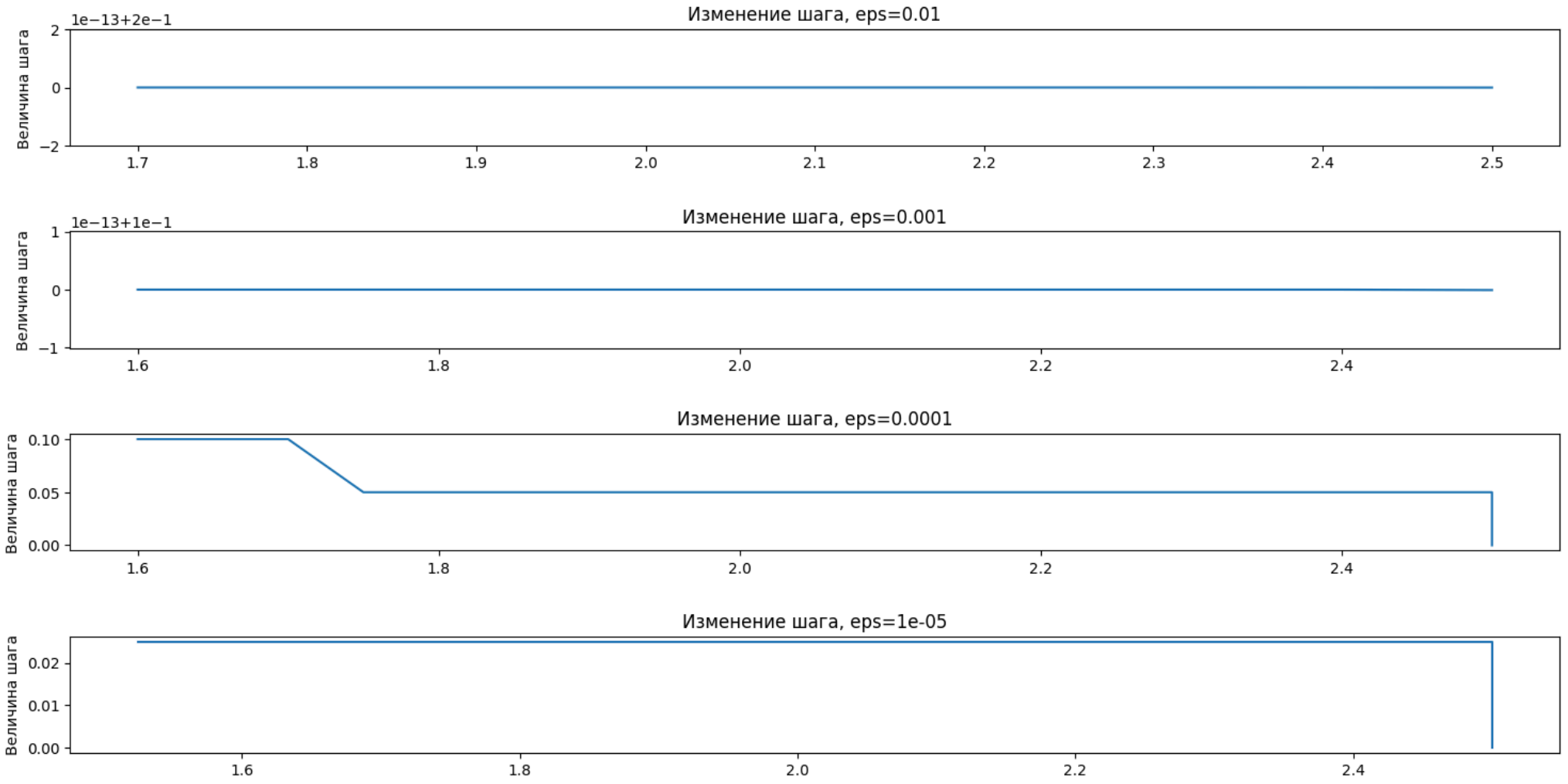

Рисунок 4. Решение для разных значений заданной точности