Санкт-Петербургский
Политехнический университет Петра Великого

Отчет по лабораторной работе №8

Решение задачи Коши.

Выполнил: *Сулейманов Вадим Дмитриевич*

Группа: *5030102/10401*

Преподаватель: *Козлов Константин Николаевич*

Санкт-Петербург
2024

Содержание

Постановка задачи

Требуется запрограммировать метод Хойна 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

```python
def heun_step(t, v, h):
    k1 = fs(t, v, call_counter)
    k2 = fs(t + h, v + h * k1, call_counter)
    return v + (h / 2) * (k1 + k2)

while t < end and call_counter[0] < max_calls:
    k1 = fs(t, v, call_counter)
    k2 = fs(t + step, v + step * k1, call_counter)
    v1 = v + (step / 2) * (k1 + k2)

    k2 = fs(t + step / 2, v + step / 2 * k1, call_counter)
    v2 = v + (step / 4) * (k1 + k2)

    v2 = heun_step(t + step / 2, v2, step / 2)

    error = np.linalg.norm(v2 - v1) / 3
    if error > tolerance:
        step /= 2
    elif error < tolerance / 64:
        step *= 2

    if error < tolerance:
        if t + step > end:
            step = end - t
        t += step
        v = v1
```

Результат

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
2.300000	0.050000	2.22444e-05	75	0.743766	1.567420	4.451871
2.350000	0.050000	2.33906e-05	80	0.730612	1.608110	4.680143
2.400000	0.050000	2.45962e-05	85	0.717758	1.649267	4.920111
2.450000	0.050000	2.58644e-05	90	0.705200	1.690849	5.172377
2.500000	0.050000	2.71984e-05	95	0.692932	1.732810	5.437568
2.500000	0.000000	2.86017e-05	100	0.680948	1.775097	5.716349

Графики

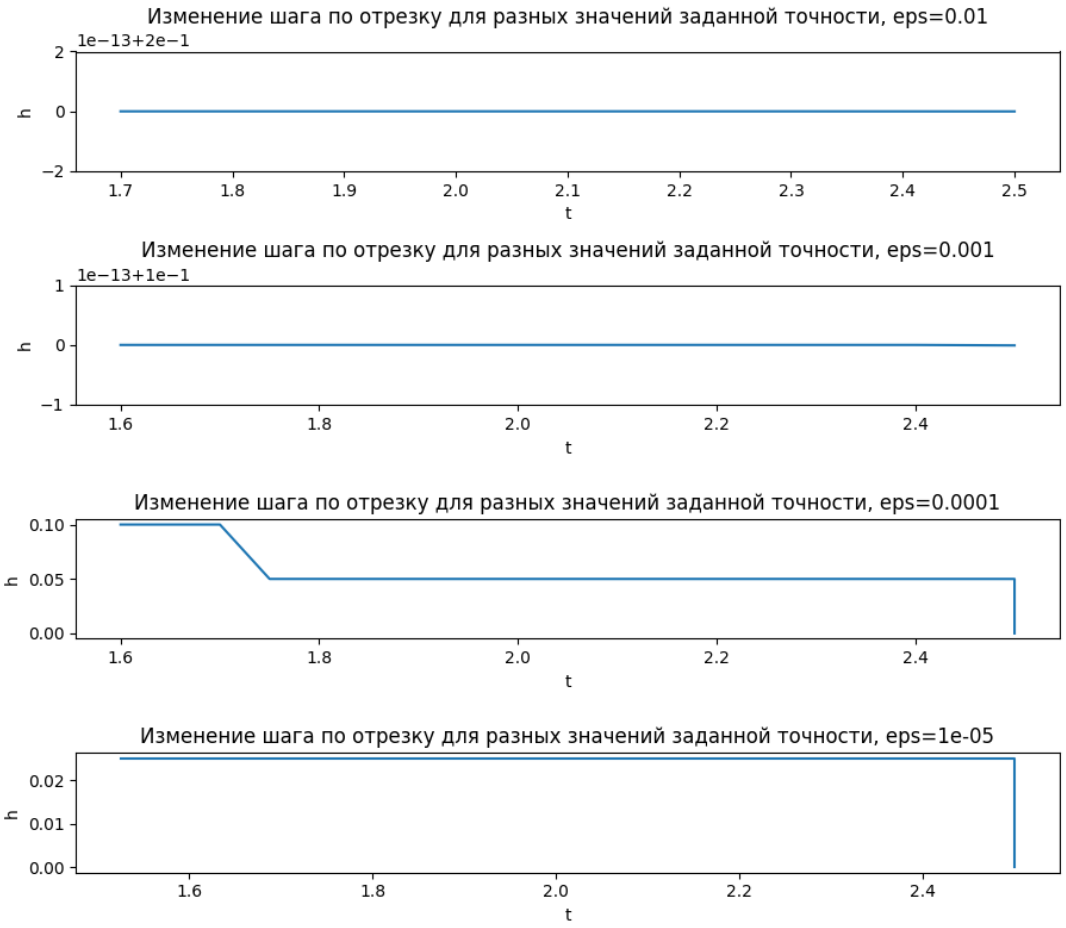

Рис.1 «Изменение шага по отрезку для разных значений заданной точности»

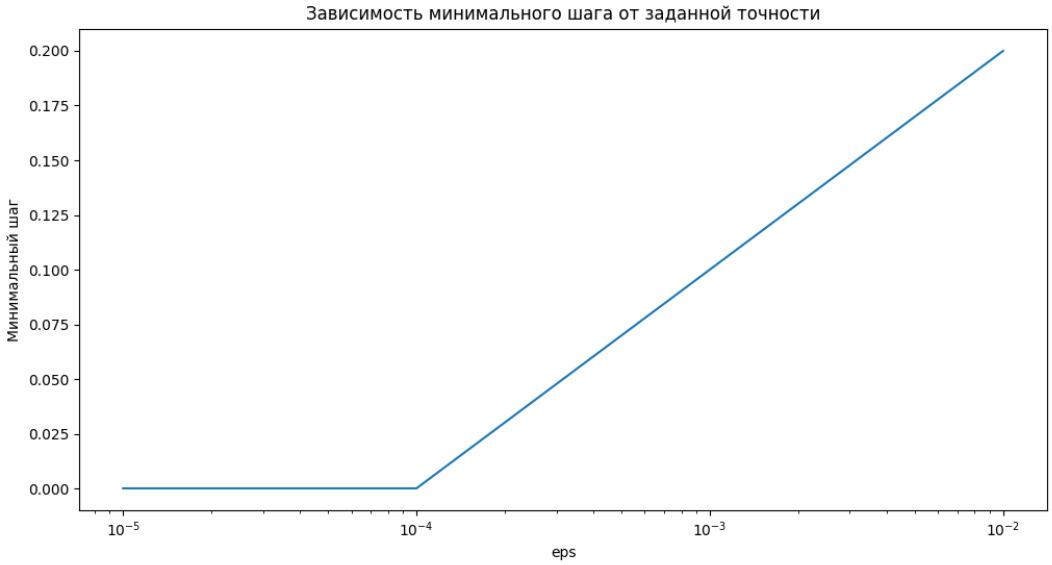

Рис.2 «Зависимость минимального шага от заданной точности»

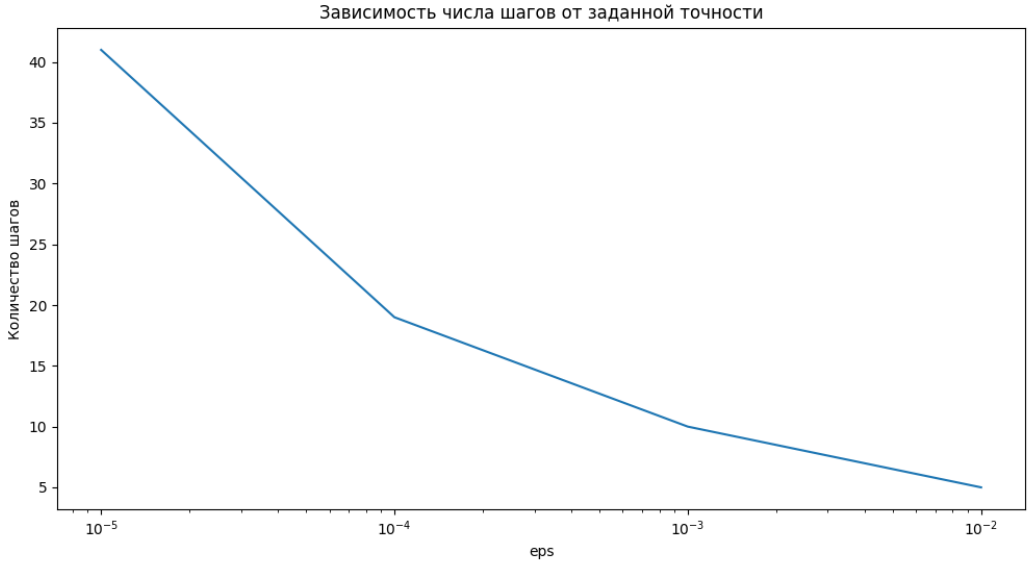

Рис.3 «Зависимость числа шагов от заданной точности»

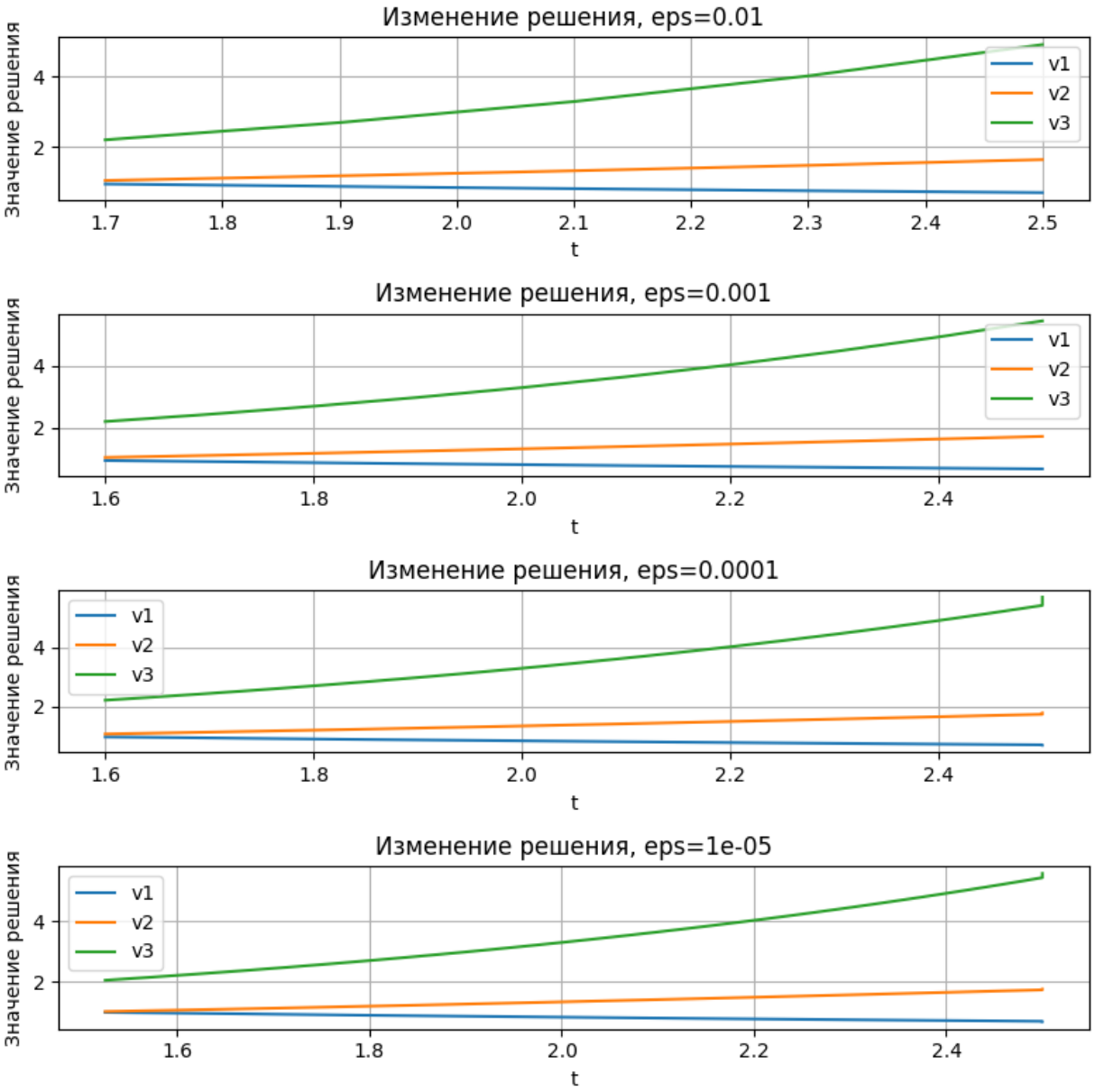

Рис.4 «Решения для разных значений заданной точности»