Санкт-Петербургский политехнический университет имени Петра великого
Институт компьютерных наук и технологий
Высшая школа интеллектуальных систем и суперкомпьютерных технологий

Отчет по лабораторной работе №3

Дисциплина "Вычислительная математика"

Выполнил студент гр. 3530901\10003 _____ Рубцов Е.А.

Руководитель _____ Цыган В.Н.

"___"_____ 2023 г.

Санкт-петербург
2023 г.

Содержание

1 Задание

Вариант 18: Привести дифференциальное уравнение: $t^2y'' + t^3y' + (t^2 - 2)y = 0$ к системе двух дифференциальных уравнений первого порядка.
Начальные условия: $y_{t=1} = 1; y'_{t=1} = -1$
Точное решение: $y(t) = \frac{1}{t}$
решить на интервале: $1 \leq t \leq 2$

1. Используя программу RKF45 с шагом печати $h_{print} = 0.1$ и выбранной вами погрешностью EPS в диапазоне 0.001 - 0.00001, а также составить собственную программу и решить с шагом интегрирования $h_{int} = 0.1$

2. Используя усовершенствованный метод ломанных Эйлера

Сравнить результаты, полученные заданными приближенными способами с точным решением.
Исследовать влияние величины шага интегрирования h_{int} на величины локальной и глобальной погрешностей решения заданного уравнения для чего решить уравнение, используя 2 – 3 значения шага интегрирования, существенно меньшие исходной величины 0.1 (например, $h_{int} = 0.05$, $h_{int} = 0.025$, $h_{int} = 0.0125$)

2 Инструменты

Для выполнения поставленного задания был выбран язык python версии 3.10. Были использованы следующие библиотеки:

1. NumPy – для улучшения скорости расчетов

2. SciPy – библиотека предоставляет неообходимые функции для решения дифф. уравнений

3. MatPlotLib – для отрисовки графиков

4. Tabulate – для более удобного вывода таблиц значений в консоль

3 Ход выполнения работы

3.1 Порядок действий:

1. Привести исходное уравнение к системе двух дифференциальных уравнений первого порядка

2. Получить решение полученной системы используя RKF45

3. Получить решение полученной системы используя усовершенствованный метод ломаных Эйлера

3.2 Задача №1

Сделаем коэффициент при производной второй степени равным 1, для этого разделим все уравнение на t^2:

$$y'' + ty' + \frac{t^2 - 2}{t^2}y = 0$$

Положим $\alpha_1 = t$ и $\alpha_2 = \frac{t^2-2}{t^2}$, получим:

$$y'' + \alpha_1 y' + \alpha_2 y = 0$$

Это уравнение в векторно-матричной форме имеет вид: $\frac{dx}{dt} = Ax + f(t)$, где f(t) = 0, а матрица A – это матрица Фробениуса вида: $A = \begin{pmatrix} -\alpha_1 & -\alpha_2 \\ 1 & 0 \end{pmatrix}, x = \begin{pmatrix} x^{(1)} \\ x^{(2)} \end{pmatrix} = \begin{pmatrix} y' \\ y \end{pmatrix}$ Таким образом получаем систему уравнений:

$$\begin{cases} x^{(2)'} = x^{(1)} \\ x^{(1)'} = -\alpha_1 x^{(1)} - \alpha_2 x^{(2)} \end{cases}$$

3.3 Задача №2

Решение:
Создадим функцию для получения значений системы уравнений для заданных начальных условий и t:

```
1
2   def func(t, X):
3       dX = np.zeros(X.shape)
4       dX[0] = X[1]
5       dX[1] = -t * X[1] - (np.power(t, 2) - 2) / np.power(t, 2) * X[0]
6       return dX
```

Так же создадим функцию для получения точного значения решения(для сравнения погрешностей):

```
1
2   def func_exact(t):
```

Далее создадим аналог функции RKF45, используя библиотеку SciPy:

```
1
2   def RKF45(f, T, x0):
3       rk_integ = ode(f).set_integrator("dopri5", atol=EPS).
        ↪ set_initial_value(x0, T[0])
4
5       X = np.array([x0, *[rk_integ.integrate(T[i]) for i in range(1, len(T)
        ↪ )]])
6
7       # Split the array to values of Y and values of Y derivative
8       return X[:, 0], X[:, 1]
```

3.4 Задача №3

Усовершенствованный метод ломаных Эйлера: $\begin{cases} x^*_{n+1/2} = x_n + \frac{h}{2} f(t_n, x_n), \\ x_{n+1} = x_n + hf\left(t_n + \frac{h}{2}, x^*_{n+1/2}\right) \end{cases}$

Реализуем его в качестве функции:

```
        X = np.zeros((len(T), len(x0)))
        X[0] = x0
        h = T[1] - T[0]

        for i in range(len(T) - 1):
            x_star = X[i] + h/2 * func(T[i], X[i])
            X[i + 1] = X[i] + h * f(T[i] + h/2, x_star)

        return X[:, 0]
```

3.5 Отрисовка графиков результата

Создадим функцию, которая позволит нам получать решение данного уравнения с заданным шагом и интервалом:

```
    def evaluate(h, rang):
        global T
        global Y_EXACT
        global Y_RKF45
        global Y_DER_RKF45
        global Y_RKF45_ERR
        global Y_EULER
        global Y_EULER_ERR

        x0 = np.array([1, -1])
        T = np.arange(rang[0], rang[1] + h, h)
        Y_EXACT = func_exact(T)

        Y_RKF45, Y_DER_RKF45 = RKF45(func, T, x0)
        Y_RKF45_ERR = Y_RKF45 - Y_EXACT

        Y_EULER = eulers_method(func, T, x0)
        Y_EULER_ERR = Y_EULER - Y_EXACT
```

Для удобства отрисовки графиков создадим отдельную функцию:

```
def draw_graphs(values, titles, output_filename):
    fig, *ax = plt.subplots(nrows=1, ncols=len(values))

    figsizes = [0, 4, 12, 12]
    fig.set_figwidth(figsizes[len(values)])

    for i in range(len(values)):
        ax[0][i].title.set_text(titles[i])
        ax[0][i].grid()
        ax[0][i].set(xlabel='t', ylabel='y')

        if len(values[i][0]) < 25:
            ax[0][i].plot(values[i][0], values[i][1], marker='o')
        else:
            ax[0][i].plot(values[i][0], values[i][1], marker=',')

    fig.savefig(output_filename, bbox_inches='tight')
    plt.close(fig)
```

Так же создадим функцию для вывода точных значений результата в консоль (для удобства выводятся значения только для 10 точек):

```
def print_table():
    column_titles = ["t", "Exact value", "RKF45", "RKF45 err", "Euler's
    ↪ method", "Euler's method err"]
    table = []
    sample_values = [1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9,
    ↪ 2.0]

    it = 0
    for i in range(len(sample_values)):
        while not np.isclose(sample_values[i], T[it], atol=1e-05):
            it += 1
        table.append([round(T[it], 5), Y_EXACT[it], Y_RKF45[it],
    ↪ Y_RKF45_ERR[it], Y_EULER[it], Y_EULER_ERR[it]])

    print(tabulate(table, column_titles, 'pretty'))
    print("RKF45 global err:", np.sum(Y_RKF45_ERR))
    print("Euler global err:", np.sum(Y_EULER_ERR))

    print("\n\n")
```

Далее для получения графиков результата будем вызывать эти три функции для различных значений h_{int}:

```
def main():
    evaluate(0.1, [1, 2])
    draw_graphs(
            np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
            ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
            "plots\\functions-01"
    )
    draw_graphs(
            np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
            ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
            "plots\\errors-01"
    )
    print("h = 0.1")
    print_table()

    evaluate(0.05, [1, 2])
    draw_graphs(
            np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
            ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
            "plots\\functions-005"
    )
    draw_graphs(
            np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
            ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
            "plots\\errors-005"
    )
    print("h = 0.05")
    print_table()
```

Аналогичным образом выводятся результаты для остальных значений h_{int}

Результат:

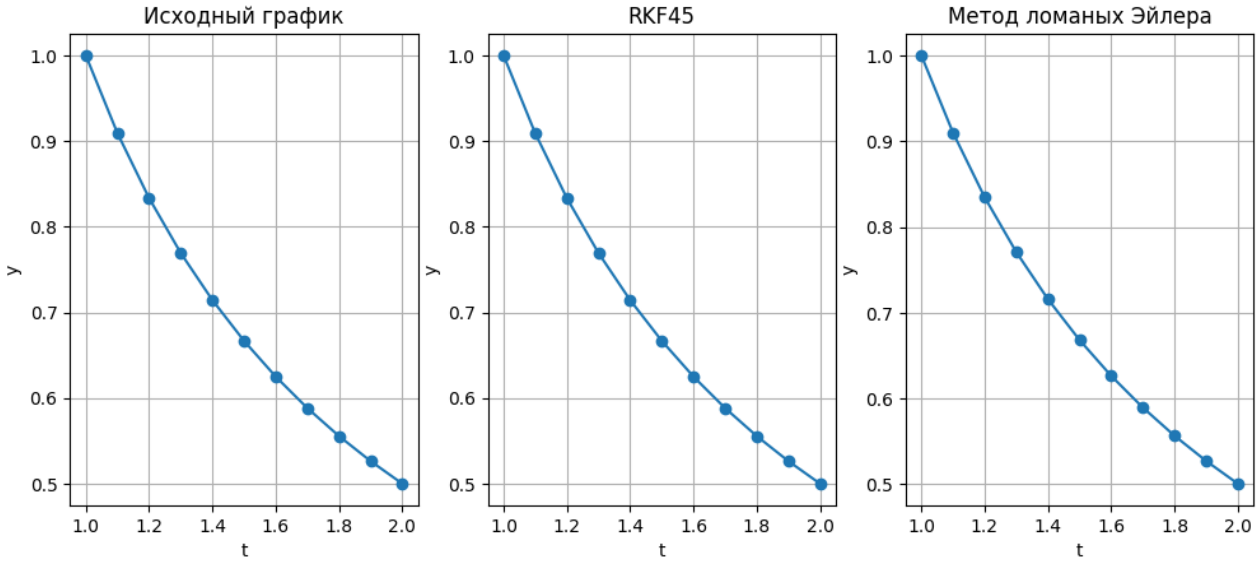

Рис. 1: Графики функций при h = 0.1

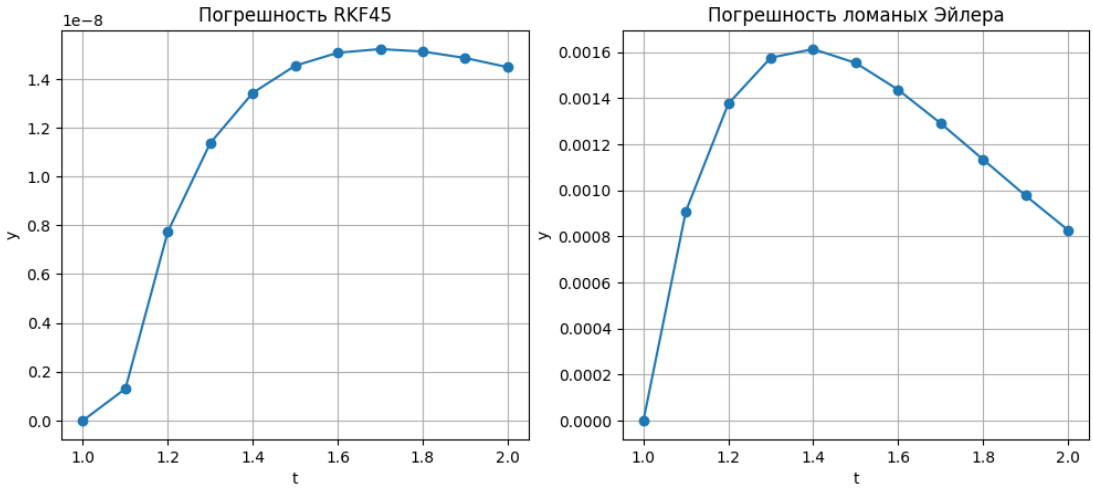

Рис. 2: Графики погрешности для h = 0.1

t	Exact value	RKF45	RKF45 err	Euler's method	Euler's method err
1.0	1.0	1.0	0.0	1.0	0.0
1.1	0.9090909090909091	0.9090909103954291	1.3045200475403362e−09	0.9099999999999999	0.0009090909090908594
1.2	0.8333333333333333	0.8333333410740418	7.740708518610973e−09	0.834709125442739	0.0013757921094057 49
1.3	0.769230769230769	0.7692307806021708	1.1371401709148188e−08	0.7708063655271175	0.0015755962963484027
1.4	0.7142857142857141	0.7142857277107874	1.3425073275286081e−08	0.7158987095779135	0.0016129952921993818
1.5	0.6666666666666665	0.6666666812064076	1.4539741077790325e−08	0.6682194413173079	0.0015527746506414086
1.6	0.6249999999999998	0.625000015068709	1.5068709169341332e−08	0.6264363458843467	0.0014363458843469346
1.7	0.5882352941176469	0.5882353093370457	1.5219398852295285e−08	0.5895260038964957	0.0012907097788488198
1.8	0.5555555555555554	0.5555555706758482	1.5120292795600676e−08	0.5566891641421717	0.0011336085866163748
1.9	0.526315789473684	0.5263158043286036	1.4854919627715901e−08	0.5272924120227146	0.0009766225490306368
2.0	0.4999999999999998	0.5000000144799004	1.4479900611874541e−08	0.5008271149717587	0.0008271149717589132

RKF45 global err: 1.2312466568520364e-07
Euler global err: 0.01269065102828748

Рис. 3: Точные значения при h = 0.1

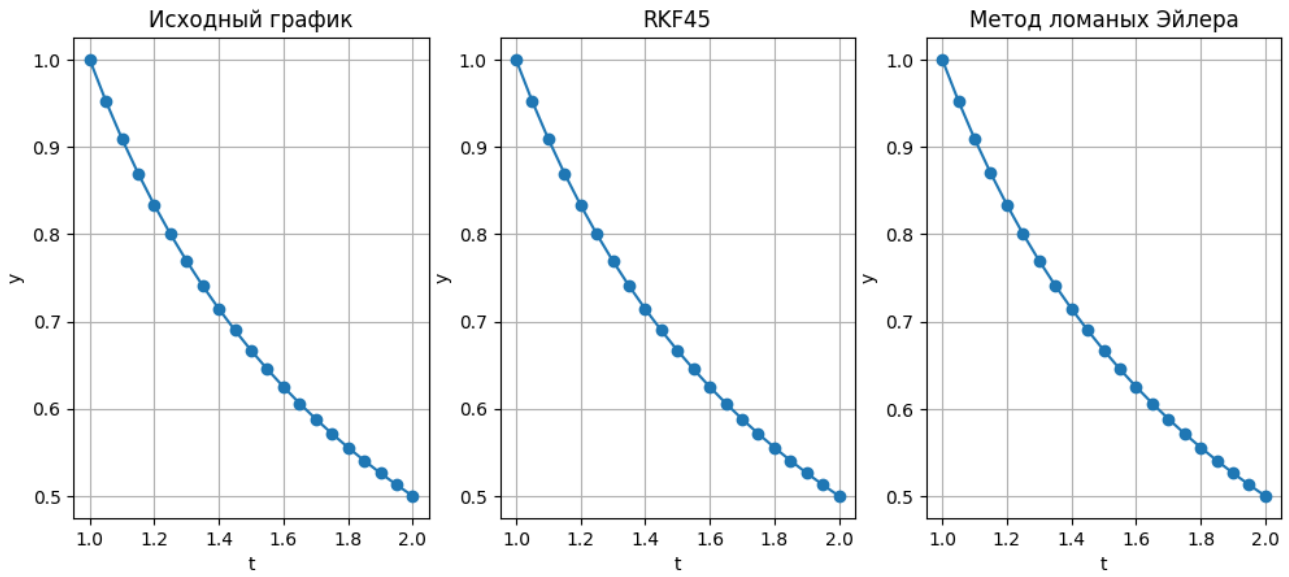

Рис. 4: Графики функций при h = 0.05

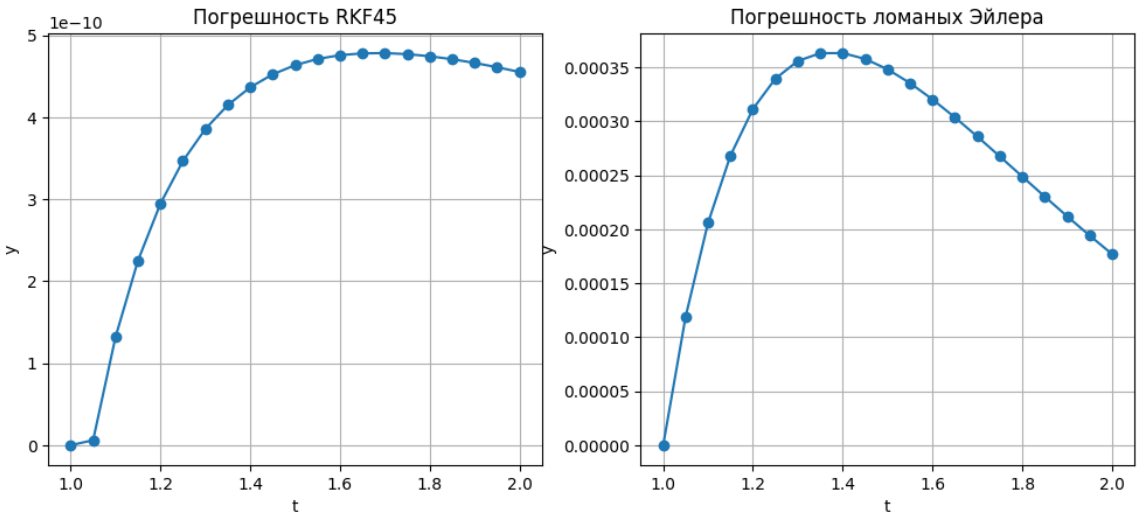

Рис. 5: Графики погрешности для h = 0.05

t	Exact value	RKF45	RKF45 err	Euler's method	Euler's method err
1.0	1.0	1.0	0.0	1.0	0.0
1.1	0.9090909090909091	0.9090909092230726	1.3216350236433527e−10	0.909296988659755	0.0002060795688458849 6
1.2	0.8333333333333333	0.8333333336279406	2.9460733852459953e−10	0.8336447823724864	0.0003114490391531443 7
1.3	0.769230769230769	0.7692307696163557	3.8558667370125477e−10	0.76958665144745 22	0.0003558822116683145 46
1.4	0.7142857142857141	0.7142857147221717	4.3645764780109175e−10	0.7146488902413343	0.0003631759556201874 5
1.5	0.6666666666666665	0.6666666671301543	4.634878036924306e−10	0.667014811881098	0.000348145214431494 5
1.6	0.6249999999999998	0.6250000004756532	4.756534055516681e−10	0.6253202922042894	0.0003202922042896405 6
1.7	0.5882352941176469	0.5882352945958439	4.781970375233868e−10	0.5885211233717113	0.0002858292540643958 5
1.8	0.5555555555555554	0.5555555560299045	4.743491155423385e−10	0.555804402009977	0.0002488464544216917 3
1.9	0.526315789473684	0.5263157899398864	4.662024100099416e−10	0.5265278056812706	0.0002120162075865961 2
2.0	0.4999999999999998	0.5000000004551748	4.5517500879554973e−10	0.5001770374899122	0.0001770374899123794 4

RKF45 global err: 7.864453088757273e-09
Euler global err: 0.005607481902541678

Рис. 6: Точные значения при h = 0.05

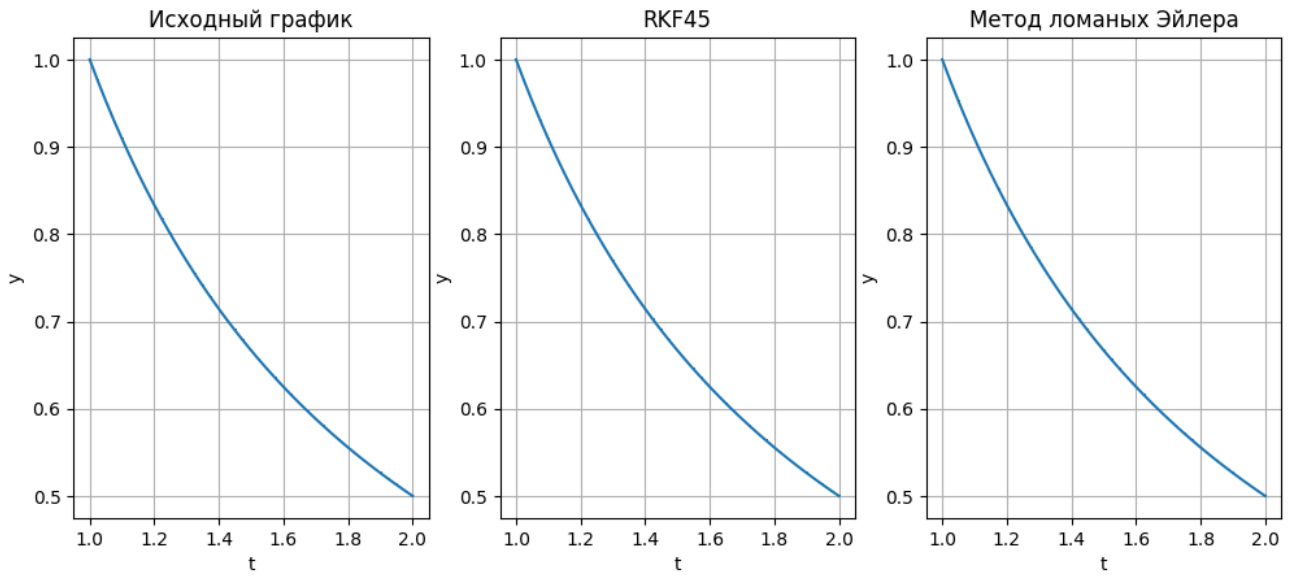

Рис. 7: Графики функций при h = 0.025

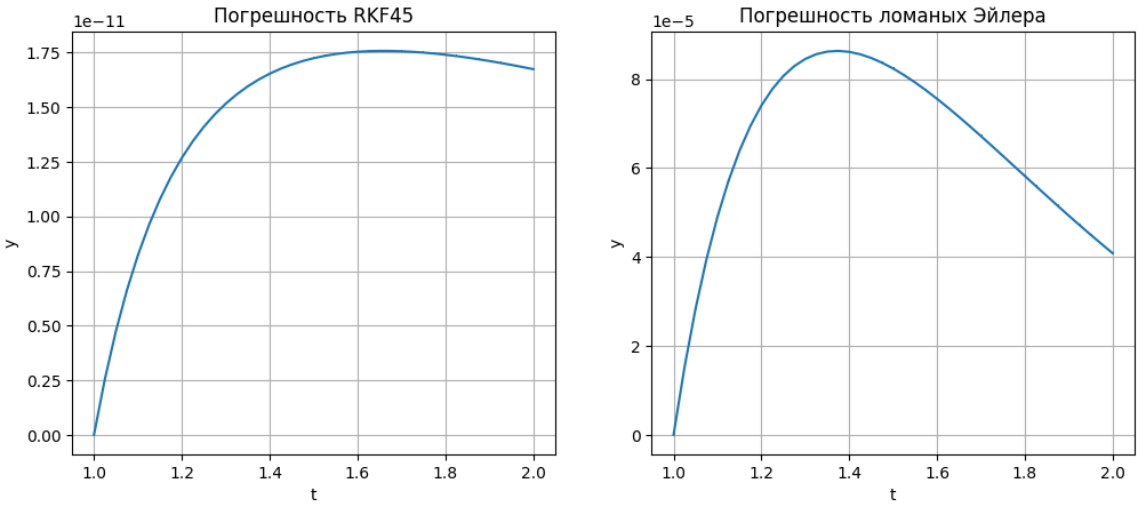

Рис. 8: Графики погрешности для h = 0.025

t	Exact value	RKF45	RKF45 err	Euler's method	Euler's method err
1.0	1.0	1.0	0.0	1.0	0.0
1.1	0.9090909090909094	0.9090909090991295	8.220091274324659e−12	0.9091399372700221	4.902817911267565e−05
1.2	0.8333333333333338	0.8333333333460126	1.2678746941219288e−11	0.8334073857939172	7.405246058334036e−05
1.3	0.7692307692307698	0.7692307692459308	1.5160983579676213e−11	0.7693152943395679	8.452510879808361e−05
1.4	0.7142857142857115	0.7142857143022479	1.653288617120552e−11	0.7143718331415433	8.611885582832102e−05
1.5	0.6666666666666674	0.6666666666839106	1.7243206862360694e−11	0.6667490410301061	8.237436343871973e−05
1.6	0.6250000000000009	0.6250000000175392	1.753830414230606e−11	0.6250755673192498	7.556731924895921e−05
1.7	0.5882352941176480	0.5882352941352099	1.7561951892730576e−11	0.5883024820313982	6.718791375026623e−05
1.8	0.5555555555555565	0.555555555572959	1.7402523866394404e−11	0.5556137739153012	5.821835974473277e−05
1.9	0.5263157894736851	0.526315789490802	1.7116974504460813e−11	0.5263650902501019	4.930077641684960e−05
2.0	0.5000000000000009	0.5000000000167453	1.674438365739661e−11	0.5000408433144072	4.084331440634692e−05

RKF45 global err: 6.032587762661024e-10
Euler global err: 0.0026305584190569054

Рис. 9: Точные значения при h = 0.025

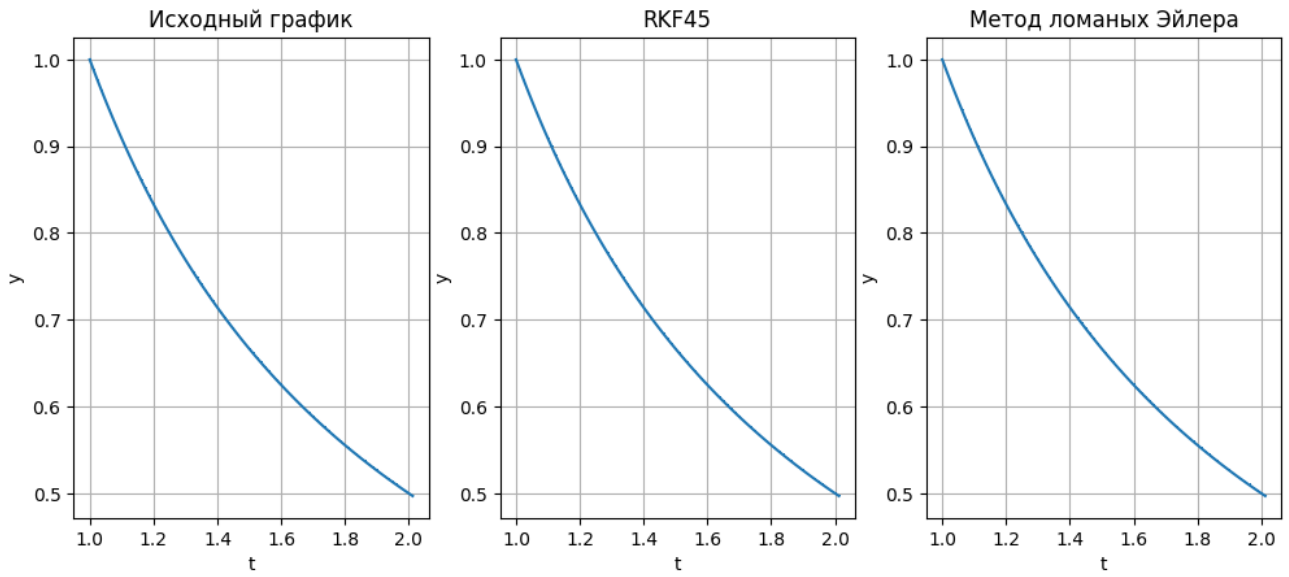

Рис. 10: Графики функций при h = 0.0125

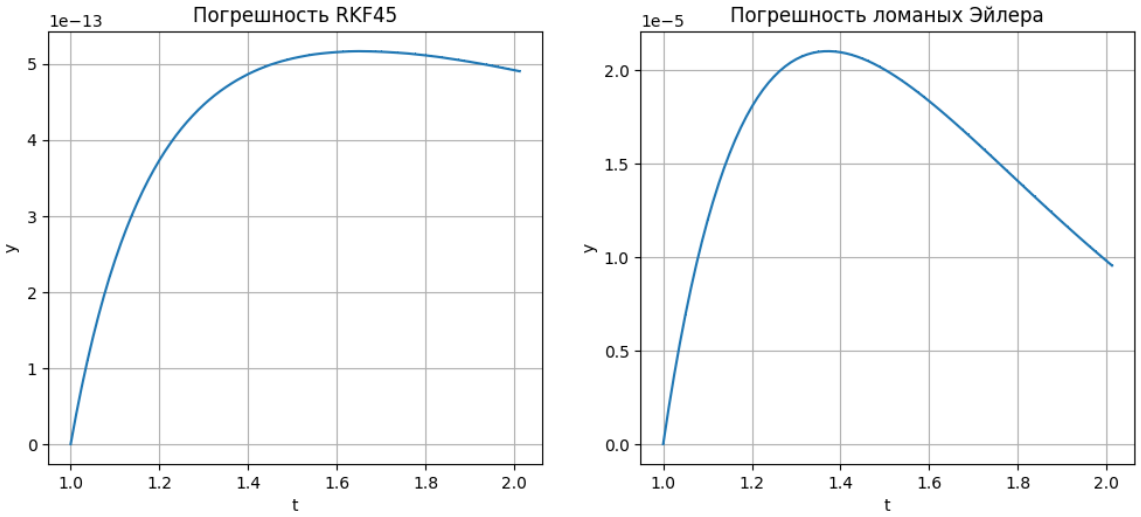

Рис. 11: Графики погрешности для h = 0.0125

t	Exact value	RKF45	RKF45 err	Euler's method	Euler's method err
1.0	1.0	1.0	0.0	1.0	0.0
1.1	0.9090909090909094	0.9090909090911512	2.418065747633591e−13	0.9091028646757606	1.1955584851230938e−05
1.2	0.8333333333333338	0.833333333333707	3.731459585765151e−13	0.8333513860137096	1.8052680375801877e−05
1.3	0.7692307692307698	0.769230769231216	4.461986335968504e−13	0.7692513637847261	2.0594553956310158e−05
1.4	0.714285714285715	0.7142857142862014	4.863887070882811e−13	0.7143066800928198	2.096580710486684e−05
1.5	0.6666666666666674	0.6666666666671744	5.07038855346309e−13	0.666686698505418	2.0031838750544928e−05
1.6	0.6250000000000009	0.6250000000005163	5.153655280309977e−13	0.6250183494864002	1.834948639933831e−05
1.7	0.588235294117648	0.5882352941181638	5.158096172408477e−13	0.5882515778131556	1.6283695507657292e−05
1.8	0.5555555555555565	0.5555555555560676	5.111466805374221e−13	0.55556963070375	1.4075148193515297e−05
1.9	0.5263157894736851	0.5263157894741878	5.027089855502709e−13	0.5263276707063623	1.1881232677257714e−05
2.0	0.5000000000000009	0.5000000000004926	4.917177776064818e−13	0.500009801976889	9.801976888157427e−06

RKF45 global err: 3.572059315004594e-11
Euler global err: 0.0012829516288189735

Рис. 12: Точные значения при h = 0.0125

После вычисления данных значений можно сделать несколько выводов:

1. Погрешность функции RKF45 значительно меньше погрешности усовершенствованного метода ломаных Эйлера
2. Глобальная погрешность уменьшается с уменьшением шага
3. Локальная погрешность увеличивается к концу промежутка

4 Вывод

В ходе данной работы мною было решено линейное дифференциальное уравнение второй степени путем приведения этого уравнения к системе из двух дифференциальных уравнений первого порядка и решения этой системы при заданных начальных условиях при помощи программ RKF45, а так же собственной программы, работающей на основе усовершенствованного метода ломаных Эйлера. Была найдена зависимость шага интегрирования h и величин глобальной и локальной погрешностей, а так же были проанализированы различия в точности вычисления для представленных методов.

5 Приложение

```
import numpy as np
from scipy.integrate import ode
import matplotlib.pyplot as plt
from tabulate import tabulate

# Required absolute tolerance for solution
EPS = 0.00001

#Global values for functions

def func(t, X):
    dX = np.zeros(X.shape)
    dX[0] = X[1]
    dX[1] = -t * X[1] - (np.power(t, 2) - 2) / np.power(t, 2) * X[0]
    return dX

def func_exact(t):
    return 1 / t

def RKF45(f, T, x0):
    rk_integ = ode(f).set_integrator("dopri5", atol=EPS).
    ↪ set_initial_value(x0, T[0])

    X = np.array([x0, *[rk_integ.integrate(T[i]) for i in range(1, len(T)
    ↪ )]])

    # Split the array to values of Y and values of Y derivative
    return X[:, 0], X[:, 1]

def eulers_method(f, T, x0):
    X = np.zeros((len(T), len(x0)))
    X[0] = x0
    h = T[1] - T[0]

    for i in range(len(T) - 1):
        x_star = X[i] + h/2 * func(T[i], X[i])
        X[i + 1] = X[i] + h * f(T[i] + h/2, x_star)

    return X[:, 0]

def evaluate(h, rang):
    global T
    global Y_EXACT
    global Y_RKF45
    global Y_DER_RKF45
    global Y_RKF45_ERR
    global Y_EULER
```

```
46          global Y_EULER_ERR
47
48          x0 = np.array([1, -1])
49          T = np.arange(rang[0], rang[1] + h, h)
50          Y_EXACT = func_exact(T)
51
52          Y_RKF45, Y_DER_RKF45 = RKF45(func, T, x0)
53          Y_RKF45_ERR = Y_RKF45 - Y_EXACT
54
55          Y_EULER = eulers_method(func, T, x0)
56          Y_EULER_ERR = Y_EULER - Y_EXACT
57
58      def draw_graphs(values, titles, output_filename):
59          fig, *ax = plt.subplots(nrows=1, ncols=len(values))
60
61          figsizes = [0, 4, 12, 12]
62          fig.set_figwidth(figsizes[len(values)])
63
64          for i in range(len(values)):
65              ax[0][i].title.set_text(titles[i])
66              ax[0][i].grid()
67              ax[0][i].set(xlabel='t', ylabel='y')
68
69              if len(values[i][0]) < 25:
70                  ax[0][i].plot(values[i][0], values[i][1], marker='o')
71              else:
72                  ax[0][i].plot(values[i][0], values[i][1], marker=',')
73
74          fig.savefig(output_filename, bbox_inches='tight')
75          plt.close(fig)
76
77      def print_table():
78          column_titles = ["t", "Exact value", "RKF45", "RKF45 err", "Euler's
        ↪ method", "Euler's method err"]
79          table = []
80          sample_values = [1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9,
        ↪ 2.0]
81
82          it = 0
83          for i in range(len(sample_values)):
84              while not np.isclose(sample_values[i], T[it], atol=1e-05):
85                  it += 1
86              table.append([round(T[it], 5), Y_EXACT[it], Y_RKF45[it],
        ↪ Y_RKF45_ERR[it], Y_EULER[it], Y_EULER_ERR[it]])
87
88          print(tabulate(table, column_titles, 'pretty'))
89          print("RKF45 global err:", np.sum(Y_RKF45_ERR))
90          print("Euler global err:", np.sum(Y_EULER_ERR))
91
```

```
92          print("\n\n")
93
94      def main():
95          evaluate(0.1, [1, 2])
96          draw_graphs(
97                  np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
98                  ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
99                  "plots\\functions-01"
100         )
101         draw_graphs(
102                 np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
103                 ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
104                 "plots\\errors-01"
105         )
106         print("h = 0.1")
107         print_table()
108
109         evaluate(0.05, [1, 2])
110         draw_graphs(
111                 np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
112                 ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
113                 "plots\\functions-005"
114         )
115         draw_graphs(
116                 np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
117                 ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
118                 "plots\\errors-005"
119         )
120         print("h = 0.05")
121         print_table()
122
123         evaluate(0.025, [1, 2])
124         draw_graphs(
125                 np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
126                 ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
127                 "plots\\functions-0025"
128         )
129         draw_graphs(
130                 np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
131                 ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
132                 "plots\\errors-0025"
133         )
134         print("h = 0.025")
135         print_table()
136
137         evaluate(0.0125, [1, 2])
138         draw_graphs(
139                 np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
140                 ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
```

```
141                 "plots\\functions-00125"
142         )
143         draw_graphs(
144                 np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
145                 ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
146                 "plots\\errors-00125"
147         )
148         print("h = 0.0125")
149         print_table()
150 
151 
152     if __name__ == "__main__":
153         main()
```

Рис. 1: Полный код программы main.py