

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	2
2	Инструменты	2
3	Ход выполнения работы	2
3.1	Порядок действий:	2
3.2	Задача №1	3
3.3	Задача №2	3
3.4	Задача №3	4
3.5	Отрисовка графиков результата	4
4	Вывод	11
5	Приложение	12

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

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

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

Далее создадим аналог функции RKF45, используя библиотеку SciPy:

```
1 def RKF45(f, T, x0):
2     rk_integ = ode(f).set_integrator("dopri5", atol=EPS).
3     ↪ 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
8     # Split the array to values of Y and values of Y derivative
9     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:

```
1 def eulers_method(f, T, x0):
2     X = np.zeros((len(T), len(x0)))
3     X[0] = x0
4     h = T[1] - T[0]
5
6     for i in range(len(T) - 1):
7         x_star = X[i] + h/2 * func(T[i], X[i])
8         X[i + 1] = X[i] + h * f(T[i] + h/2, x_star)
9
10    return X[:, 0]
```

3.5 Отрисовка графиков результата

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

```
1 def evaluate(h, rang):
2     global T
3     global Y_EXACT
4     global Y_RKF45
5     global Y_DER_RKF45
6     global Y_RKF45_ERR
7     global Y_EULER
8     global Y_EULER_ERR
9
10    global RKF45_FIRST_STEP_ERR
11    global EULER_FIRST_STEP_ERR
12    global RKF45_GLOBAL_ERR
13    global EULER_GLOBAL_ERR
14
15    x0 = np.array([1, -1])
16    T = np.arange(rang[0], rang[1] + h, h)
17    Y_EXACT = func_exact(T)
18
19    Y_RKF45, Y_DER_RKF45 = RKF45(func, T, x0)
20    Y_RKF45_ERR = Y_RKF45 - Y_EXACT
21    RKF45_FIRST_STEP_ERR.append(Y_RKF45_ERR[1])
22    RKF45_GLOBAL_ERR.append(np.sum(Y_RKF45_ERR))
23
24    Y_EULER = eulers_method(func, T, x0)
25    Y_EULER_ERR = Y_EULER - Y_EXACT
26    EULER_FIRST_STEP_ERR.append(Y_EULER_ERR[1])
27    EULER_GLOBAL_ERR.append(np.sum(Y_EULER_ERR))
```

Для удобства отрисовки графиков создадим отдельную функцию:

```

1  def draw_graphs(values, titles, output_filename):
2      fig, *ax = plt.subplots(nrows=1, ncols=len(values))
3
4      figsizes = [0, 4, 12, 12]
5      fig.set_figwidth(figsizes[len(values)])
6
7      for i in range(len(values)):
8          ax[0][i].title.set_text(titles[i])
9          ax[0][i].grid()
10         ax[0][i].set(xlabel='t', ylabel='y')
11
12         if len(values[i][0]) < 25:
13             ax[0][i].plot(values[i][0], values[i][1], marker='o')
14         else:
15             ax[0][i].plot(values[i][0], values[i][1], marker=',')
16
17     fig.savefig(output_filename, bbox_inches='tight')
18     plt.close(fig)

```

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

```

1  def print_table():
2      column_titles = ["t", "Exact value", "RK45", "RK45 err", "Euler's
↪ method", "Euler's method err"]
3      table = []
4      sample_values = [1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9,
↪ 2.0]
5
6      it = 0
7      for i in range(len(sample_values)):
8          while not np.isclose(sample_values[i], T[it], atol=1e-05):
9              it += 1
10         table.append([round(T[it], 5), Y_EXACT[it], Y_RKF45[it],
↪ Y_RKF45_ERR[it], Y_EULER[it], Y_EULER_ERR[it]])
11
12     print(tabulate(table, column_titles, 'pretty'))

```

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

```

1  def main():
2      evaluate(0.1, [1, 2])
3      draw_graphs(
4          np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
5          ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
6          "plots\\functions-01"
7      )
8      draw_graphs(
9          np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
10         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
11         "plots\\errors-01"
12     )
13     print("h = 0.1")
14     print_table()
15     print("\n\n")
16
17     evaluate(0.05, [1, 2])
18     draw_graphs(
19         np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
20         ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
21         "plots\\functions-005"
22     )
23     draw_graphs(
24         np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
25         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
26         "plots\\errors-005"
27     )
28     print("h = 0.05")
29     print_table()
30     print("\n\n")

```

Аналогичным образом выводятся результаты для остальных значений h_{int}

Результат:

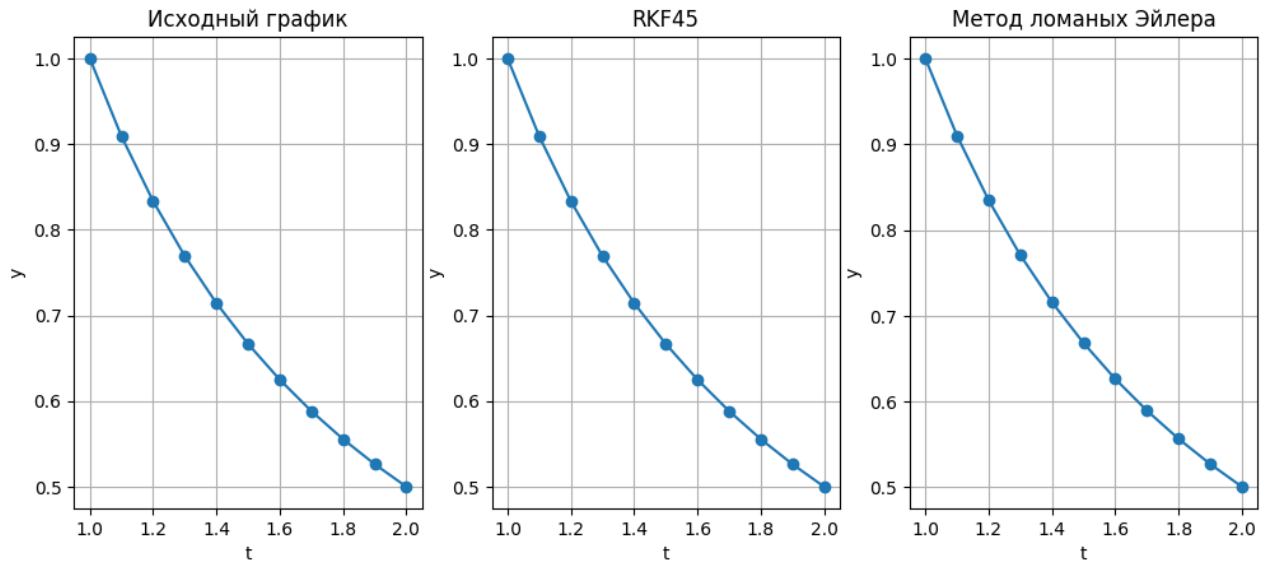


Рис. 1: Графики функций при $h = 0.1$

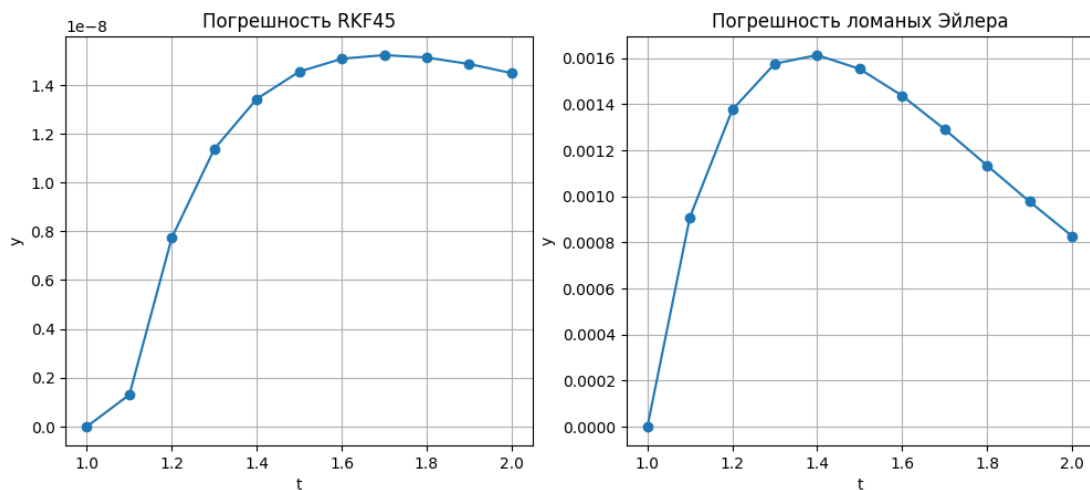


Рис. 2: Графики погрешности для $h = 0.1$

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	0.001375792109405749
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

Рис. 3: Точные значения при $h = 0.1$

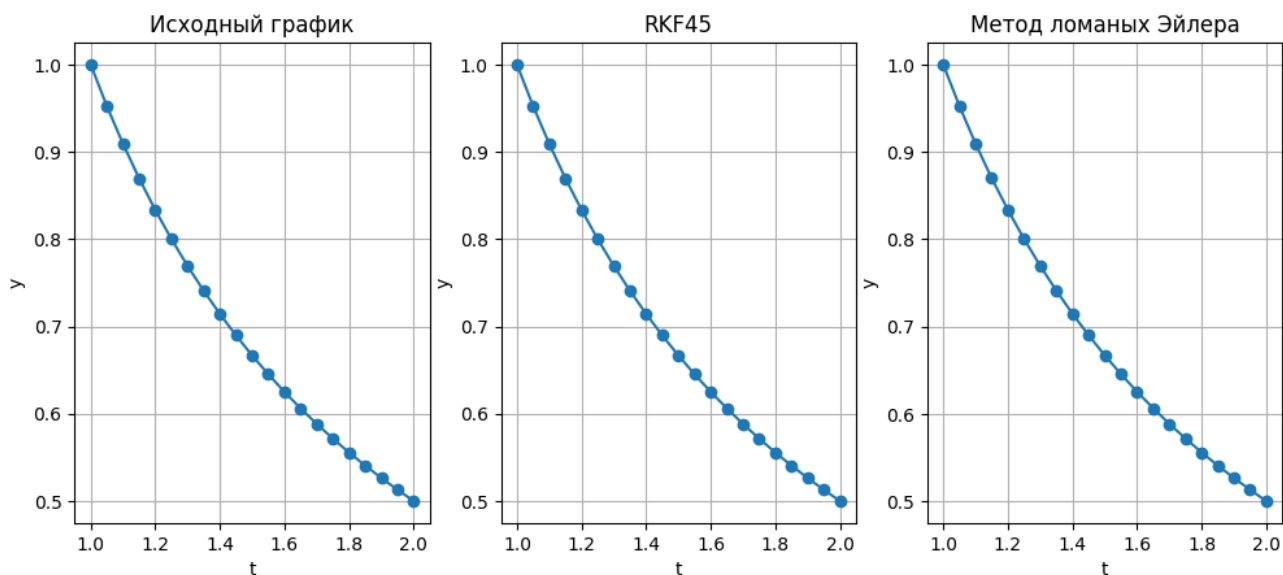


Рис. 4: Графики функций при $h = 0.05$

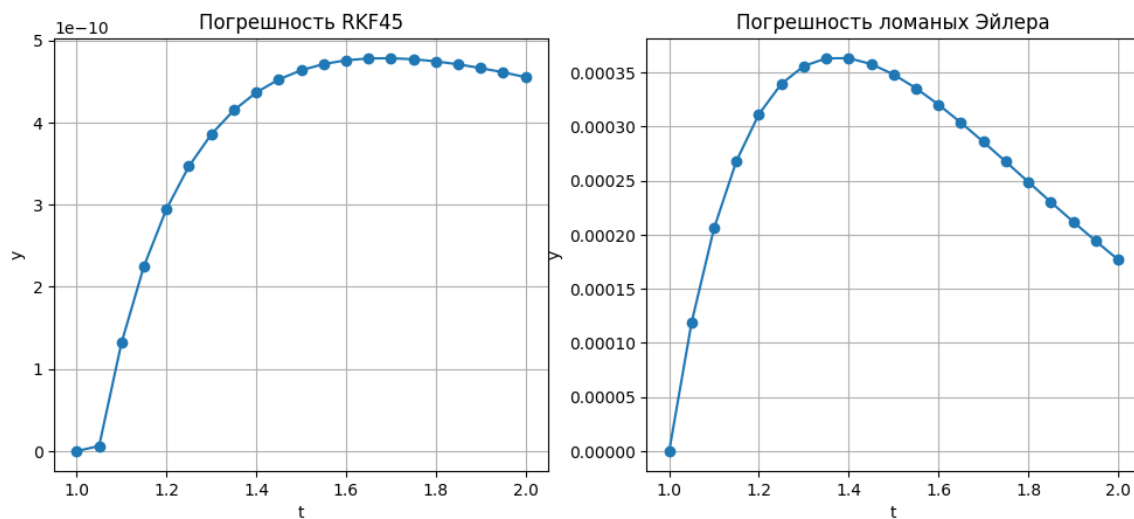


Рис. 5: Графики погрешности для $h = 0.05$

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	0.00020607956884588496
1.2	0.8333333333333333	0.8333333336279406	2.9460733852459953e-10	0.8336447823724864	0.00031144903915314437
1.3	0.769230769230769	0.7692307696163557	3.8558667370125477e-10	0.7695866514474522	0.00035588221668314546
1.4	0.7142857142857141	0.7142857147221717	4.3645764780109175e-10	0.7146488902413343	0.00036317595562018745
1.5	0.6666666666666665	0.6666666671301543	4.634878036924306e-10	0.667014811881098	0.0003481452144314945
1.6	0.6249999999999998	0.6250000004756532	4.756534055516681e-10	0.6253202922042894	0.00032029220428964056
1.7	0.5882352941176469	0.5882352945958439	4.781970375233868e-10	0.5885211233717113	0.00028582925406439585
1.8	0.5555555555555554	0.5555555560299045	4.743491155423385e-10	0.555804402009977	0.00024884645442169173
1.9	0.526315789473684	0.526315789398864	4.662024100099416e-10	0.5265278056812706	0.00021201620758659612
2.0	0.4999999999999998	0.5000000004551748	4.5517500879554973e-10	0.5001770374899122	0.00017703748991237944

Рис. 6: Точные значения при $h = 0.05$

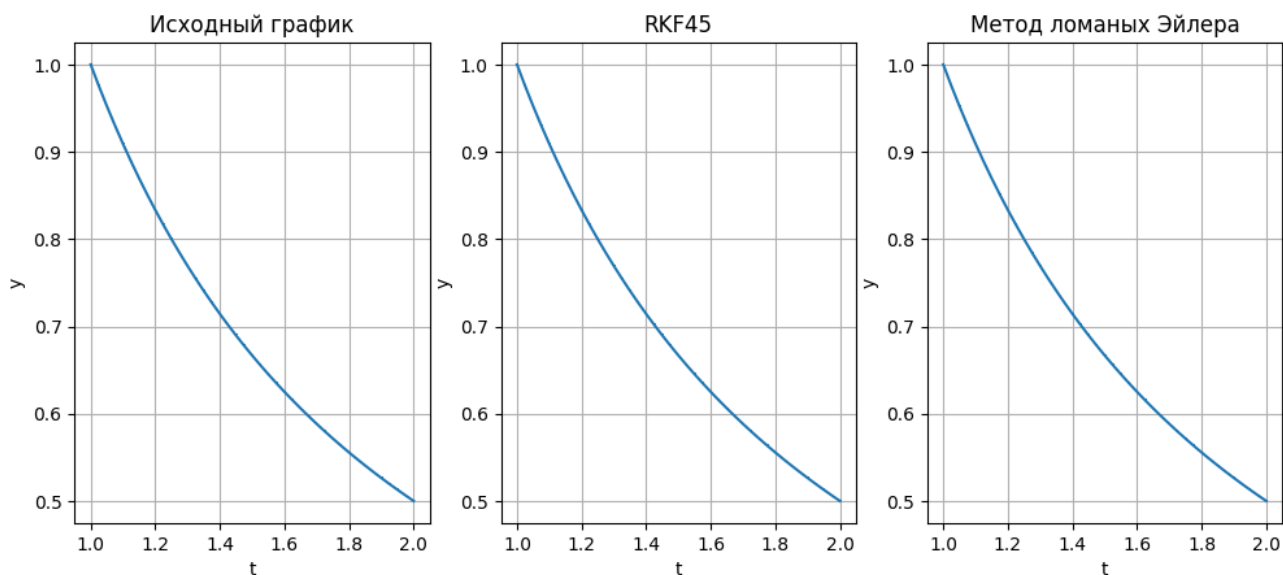


Рис. 7: Графики функций при $h = 0.025$

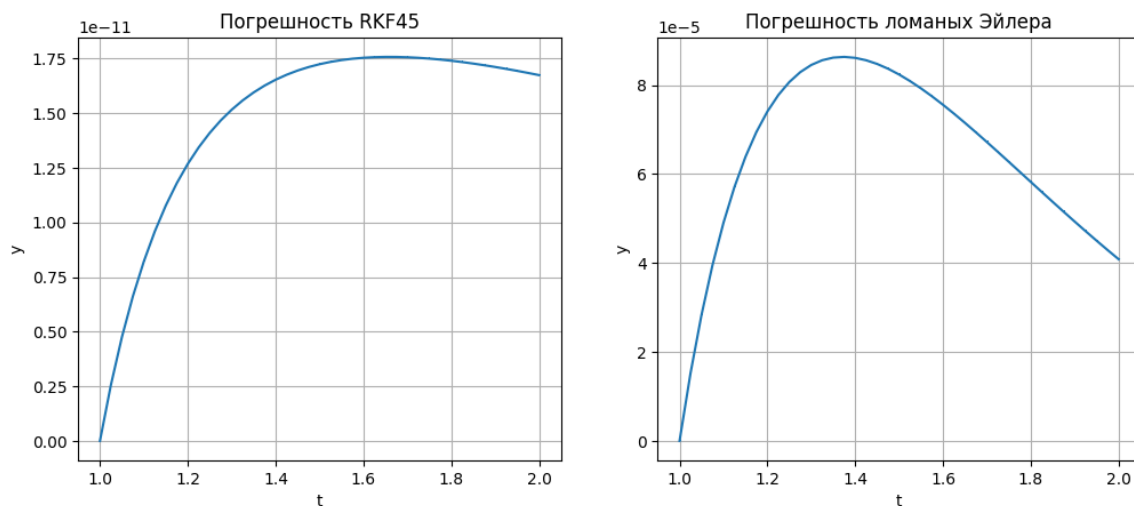


Рис. 8: Графики погрешности для $h = 0.025$

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	0.83333333333460126	1.2678746941219288e-11	0.8334073857939172	7.405246058334036e-05
1.3	0.7692307692307698	0.7692307692459308	1.5160983579676213e-11	0.7693152943395679	8.452510879808361e-05
1.4	0.714285714285715	0.7142857143022479	1.653288617120552e-11	0.7143718331415433	8.611885582832102e-05
1.5	0.6666666666666674	0.66666666666839106	1.7243206862360694e-11	0.6667490410301061	8.237436343871973e-05
1.6	0.6250000000000009	0.6250000000175392	1.753830414230606e-11	0.6250755673192498	7.556731924895921e-05
1.7	0.588235294117648	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	4.93007764168496e-05
2.0	0.5000000000000009	0.5000000000167453	1.674438365739661e-11	0.5000408433144072	4.084331440634692e-05

Рис. 9: Точные значения при $h = 0.025$

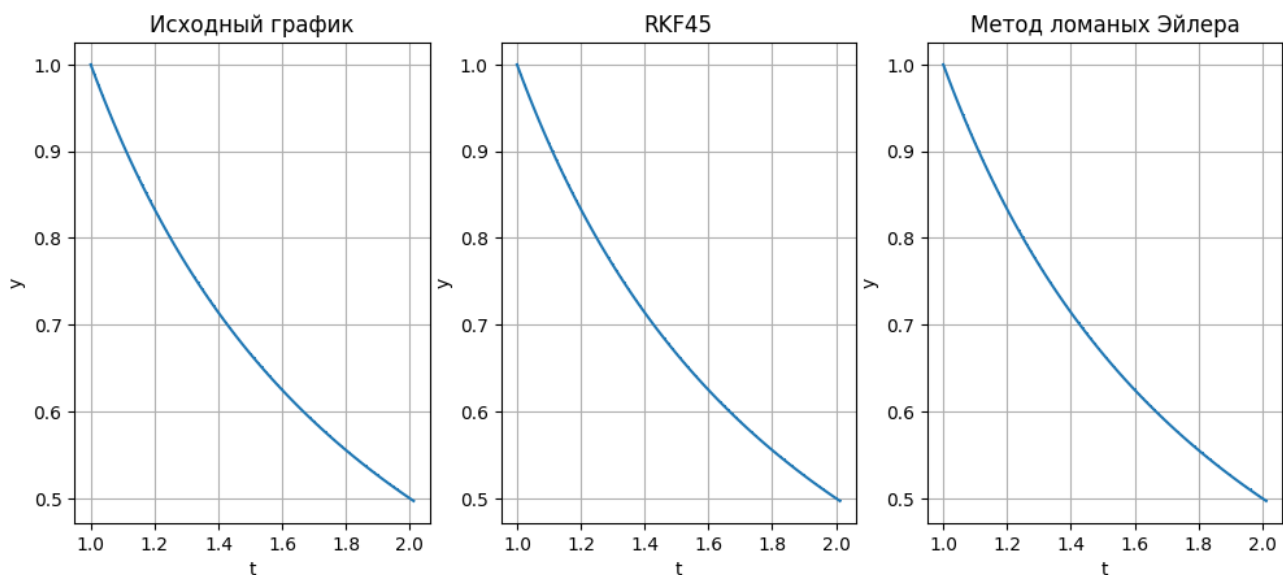


Рис. 10: Графики функций при $h = 0.0125$

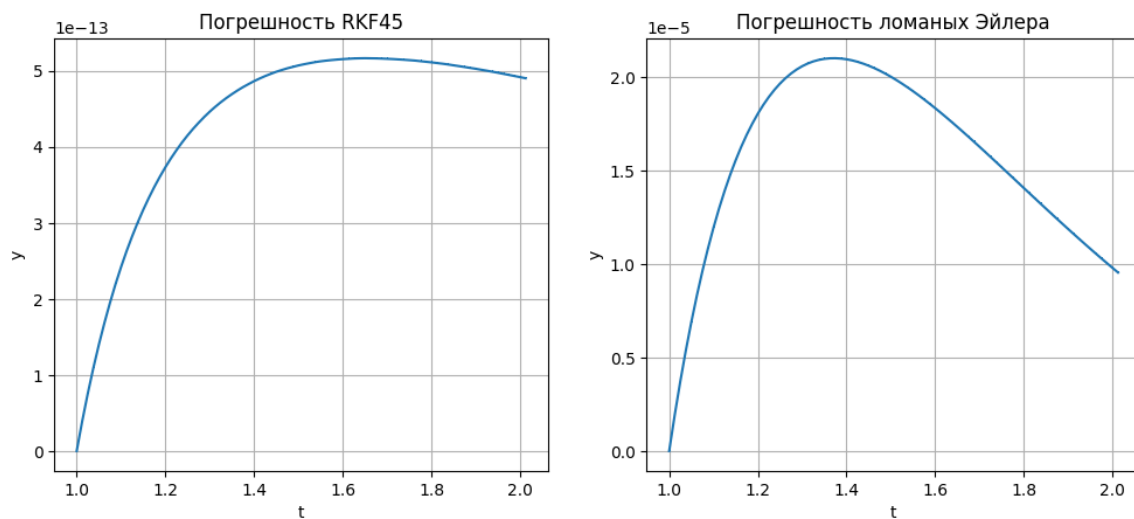


Рис. 11: Графики погрешности для $h = 0.0125$

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	4.91717776064818e-13	0.500009801976889	9.801976888157427e-06

Рис. 12: Точные значения при $h = 0.0125$

h	RKF45 local err	Euler local err	RKF45 global err	Euler global err
0.1	1.3045200475403362e-09	0.0009090909090908594	1.2312466568520364e-07	0.01269065102828748
0.05	6.143863195973154e-12	0.00011904761904757422	7.864453088757273e-09	0.005607481902541678
0.025	2.5673907444456745e-12	1.5243902439010526e-05	6.032587762661024e-10	0.0026305584190569054
0.0125	3.930189507173054e-14	1.9290123456361385e-06	3.572059315004594e-11	0.0012829516288189735

Рис. 13: Таблица погрешностей методов

Усовершенствованный метод ломаных Эйлера является методом второй степени точности, а его погрешность пропорциональна кубу шага интегрирования. Именно такую зависимость мы и можем наблюдать в значениях локальной погрешности: погрешность первого шага уменьшается примерно в восемь раз с уменьшением шага в 2 раза.

Можно так же заметить, что глобальная погрешность этого метода изменяется пропорционально первой степени шага интегрирования, что явно намного хуже, чем уменьшение локальной погрешности.

В это же время программа RKF45 показывает намного более хорошие результаты, где глобальная погрешность изменяется пропорционально четвертой степени шага интегрирования.

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

```
1  import numpy as np
2  from scipy.integrate import ode
3  import matplotlib.pyplot as plt
4  from tabulate import tabulate
5
6  # Required absolute tolerance for solution
7  EPS = 0.00001
8
9  RKF45_FIRST_STEP_ERR = []
10 EULER_FIRST_STEP_ERR = []
11 RKF45_GLOBAL_ERR = []
12 EULER_GLOBAL_ERR = []
13
14
15 def func(t, X):
16     dX = np.zeros(X.shape)
17     dX[0] = X[1]
18     dX[1] = -t * X[1] - (np.power(t, 2) - 2) / np.power(t, 2) * X[0]
19     return dX
20
21 def func_exact(t):
22     return 1 / t
23
24 def RKF45(f, T, x0):
25     rk_integ = ode(f).set_integrator("dopri5", atol=EPS).
26     ↪ set_initial_value(x0, T[0])
27
28     X = np.array([x0, *[rk_integ.integrate(T[i]) for i in range(1, len(T)
29     ↪ )]])
30
31     # Split the array to values of Y and values of Y derivative
32     return X[:, 0], X[:, 1]
33
34 def eulers_method(f, T, x0):
35     X = np.zeros((len(T), len(x0)))
36     X[0] = x0
37     h = T[1] - T[0]
38
39     for i in range(len(T) - 1):
40         x_star = X[i] + h/2 * func(T[i], X[i])
41         X[i + 1] = X[i] + h * f(T[i] + h/2, x_star)
42
43     return X[:, 0]
44
45 def evaluate(h, rang):
46     global T
47     global Y_EXACT
```

```

46     global Y_RKF45
47     global Y_DER_RKF45
48     global Y_RKF45_ERR
49     global Y_EULER
50     global Y_EULER_ERR
51
52     global RKF45_FIRST_STEP_ERR
53     global EULER_FIRST_STEP_ERR
54     global RKF45_GLOBAL_ERR
55     global EULER_GLOBAL_ERR
56
57     x0 = np.array([1, -1])
58     T = np.arange(rang[0], rang[1] + h, h)
59     Y_EXACT = func_exact(T)
60
61     Y_RKF45, Y_DER_RKF45 = RKF45(func, T, x0)
62     Y_RKF45_ERR = Y_RKF45 - Y_EXACT
63     RKF45_FIRST_STEP_ERR.append(Y_RKF45_ERR[1])
64     RKF45_GLOBAL_ERR.append(np.sum(Y_RKF45_ERR))
65
66     Y_EULER = eulers_method(func, T, x0)
67     Y_EULER_ERR = Y_EULER - Y_EXACT
68     EULER_FIRST_STEP_ERR.append(Y_EULER_ERR[1])
69     EULER_GLOBAL_ERR.append(np.sum(Y_EULER_ERR))
70
71
72 def draw_graphs(values, titles, output_filename):
73     fig, *ax = plt.subplots(nrows=1, ncols=len(values))
74
75     figsizes = [0, 4, 12, 12]
76     fig.set_figwidth(figsizes[len(values)])
77
78     for i in range(len(values)):
79         ax[0][i].title.set_text(titles[i])
80         ax[0][i].grid()
81         ax[0][i].set(xlabel='t', ylabel='y')
82
83         if len(values[i][0]) < 25:
84             ax[0][i].plot(values[i][0], values[i][1], marker='o')
85         else:
86             ax[0][i].plot(values[i][0], values[i][1], marker=',')
87
88     fig.savefig(output_filename, bbox_inches='tight')
89     plt.close(fig)
90
91 def print_table():
92     column_titles = ["t", "Exact value", "RKF45", "RKF45 err", "Euler's
93     ↪ method", "Euler's method err"]
94     table = []

```

```

94     sample_values = [1.0, 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9,
↪ 2.0]

95
96     it = 0
97     for i in range(len(sample_values)):
98         while not np.isclose(sample_values[i], T[it], atol=1e-05):
99             it += 1
100         table.append([round(T[it], 5), Y_EXACT[it], Y_RKF45[it],
↪ Y_RKF45_ERR[it], Y_EULER[it], Y_EULER_ERR[it]])
101
102     print(tabulate(table, column_titles, 'pretty'))
103
104 def print_table_errors():
105     column_titles = ["h", "RKF45 local err", "Euler local err", "RKF45
↪ global err", "Euler global err"]
106     h_values = [0.1, 0.05, 0.025, 0.0125]
107     table = []
108
109     for i in range(len(h_values)):
110         table.append((h_values[i], RKF45_FIRST_STEP_ERR[i],
↪ EULER_FIRST_STEP_ERR[i],
111                        RKF45_GLOBAL_ERR[i], EULER_GLOBAL_ERR[i]))
112     print(tabulate(table, column_titles, 'pretty'))
113
114
115 def print_table_exact():
116     table = []
117
118     for i in range(len(sample_values)):
119         table.append([round(T[it], 5), Y_EXACT[it], Y_RKF45[it],
↪ Y_RKF45_ERR[it], Y_EULER[it], Y_EULER_ERR[it]])
120
121     print(tabulate(table, column_titles, 'pretty'))
122
123 def main():
124     evaluate(0.1, [1, 2])
125     draw_graphs(
126         np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
127         ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
128         "plots\\functions-01"
129     )
130     draw_graphs(
131         np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
132         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
133         "plots\\errors-01"
134     )
135     print("h = 0.1")
136     print_table()
137     print("\n\n")

```

```

138
139     evaluate(0.05, [1, 2])
140     draw_graphs(
141         np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
142         ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
143         "plots\\functions-005"
144     )
145     draw_graphs(
146         np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
147         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
148         "plots\\errors-005"
149     )
150     print("h = 0.05")
151     print_table()
152     print("\n\n")
153
154     evaluate(0.025, [1, 2])
155     draw_graphs(
156         np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
157         ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
158         "plots\\functions-0025"
159     )
160     draw_graphs(
161         np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
162         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
163         "plots\\errors-0025"
164     )
165     print("h = 0.025")
166     print_table()
167     print("\n\n")
168
169     evaluate(0.0125, [1, 2])
170     draw_graphs(
171         np.array(([T, Y_EXACT], [T, Y_RKF45], [T, Y_EULER])),
172         ["Исходный график", "RKF45", "Метод ломаных Эйлера"],
173         "plots\\functions-00125"
174     )
175     draw_graphs(
176         np.array(([T, Y_RKF45_ERR], [T, Y_EULER_ERR])),
177         ["Погрешность RKF45", "Погрешность ломаных Эйлера"],
178         "plots\\errors-00125"
179     )
180     print("h = 0.0125")
181     print_table()
182     print("\n\n")
183
184     print_table_errors()
185
186

```



```
187     if __name__ == "__main__":  
188         main()
```

Рис. 1: Полный код программы main.py