Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н. Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

Отчёт по лабораторной работе №1 по курсу "Моделирование"

Тема Решение задачи Коши методами Пикара, Эйлера и Рунге-Кутта

Студент Ковалец К. Э.

Группа ИУ7-63Б

Преподаватель Градов В. М.

Москва — 2022 г.

1 Задание

1.1 Тема работы

Программная реализация приближенного аналитического метода и численных алгоритмов первого и второго порядков точности при решении задачи Коши для ОДУ.

1.2 Цель работы

Получение навыков решения задачи Коши для ОДУ методами Пикара и явными методами первого порядка точности (Эйлера) и второго порядка точности (Рунге-Кутта).

1.3 Исходные данные

ОДУ, не имеющее аналитического решения

$$\begin{cases} u'(x) = x^2 + u^2 \\ u(0) = 0 \end{cases} \tag{1.1}$$

1.4 Результат работы программы

- Таблица, содержащая значения аргумента с заданным шагом в интервале $[0, x_{max}]$ и результаты расчета функции $u(x)$ в приближениях Пикара (от 1-го до 4-го), а также численными методами. Границу интервала x_{max} выбирать максимально возможной из условия, чтобы численные методы обеспечивали точность вычисления решения уравнения $u(x)$ до второго знака после запятой.
- График функции в диапазоне $[-x_{max}, x_{max}]$.

2 Теоретические сведение

Имеем ОДУ, у которого отсутствует аналитическое решение:

$$\begin{cases} u'(x) = f(x,u) \\ u(\xi) = \eta \end{cases} \tag{2.1}$$

Для решения данного ОДУ были использованы 3 алгоритма.

2.1 Метод Пикара

Имеем:

$$u(x) = \eta + \int_{\xi}^{x} f(t, u(t))\, dt \tag{2.2}$$

Строим ряд функций:

$$y^{(s)} = \eta + \int_{\xi}^{x} f(t, y^{(s-1)}(t))\, dt, \qquad y^{(0)} = \eta \tag{2.3}$$

Построим 4 приближения для уравнения (2.2):

$$y^{(1)}(x) = 0 + \int_{0}^{x} t^2\, dt = \frac{x^3}{3} \tag{2.4}$$

$$y^{(2)}(x) = 0 + \int_{0}^{x} (t^2 + \left(\frac{t^3}{3}\right)^2)\, dt = \frac{x^3}{3} + \frac{x^7}{63} \tag{2.5}$$

$$y^{(3)}(x) = 0 + \int_{0}^{x} (t^2 + \left(\frac{t^3}{3} + \frac{t^7}{63}\right)^2)\, dt = \frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^{11}}{2079} + \frac{x^{15}}{59535} \tag{2.6}$$

$$y^{(4)}(x) = 0 + \int_{0}^{x} (t^2 + \left(\frac{t^3}{3} + \frac{t^7}{63} + \frac{2t^{11}}{2079} + \frac{t^{15}}{59535}\right)^2)\, dt = \frac{x^3}{3} + \frac{x^7}{63} + \frac{2x^{11}}{2079} + \frac{13x^{15}}{218295} + \frac{82x^{19}}{37328445} + \frac{662x^{23}}{10438212015} + \frac{4x^{27}}{3341878155} + \frac{x^{31}}{109876903975} \tag{2.7}$$

2.2 Метод Эйлера

$$y^{(n+1)}(x) = y^{(n)}(x) + h \cdot f(x_n, y^{(n)}) \tag{2.8}$$

Порядок точности: $O(h)$.

2.3 Метод Рунге-Кутта

$$y^{n+1}(x) = y^n(x) + h((1-\alpha)R_1 + \alpha R_2) \tag{2.9}$$

где $R1 = f(x_n, y^n)$, $R2 = f(x_n + \frac{h}{2\alpha}, y^n + \frac{h}{2\alpha}R_1)$, $\alpha = \frac{1}{2}$ или 1

Порядок точности: $O(h^2)$.

3 Исходный код алгоритмов

Листинг 3.1 – Исходный код алгоритмов

```
import matplotlib.pyplot as plt
from color import *


MAX_X = 1
STEP  = 1e-4


def f(x, y):
    return pow(x, 2) + pow(y, 2)


def PicarApprox1(x):
    return pow(x, 3) / 3


def PicarApprox2(x):
    return PicarApprox1(x) + \
        pow(x, 7) / 63


def PicarApprox3(x):
    return PicarApprox2(x)    + \
        2 * pow(x, 11) / 2079 + \
            pow(x, 15) / 59535


def PicarApprox4(x):
    return PicarApprox2(x)             + \
        2   * pow(x, 11) / 2079        + \
        13  * pow(x, 15) / 218295      + \
        82  * pow(x, 19) / 37328445    + \
        662 * pow(x, 23) / 10438212015 + \
        4   * pow(x, 27) / 3341878155  + \
                pow(x, 31) / 109876903975


def Picar(x_max, h, PicarApprox):
    result = []
    x, y = 0, 0

    while abs(x) < abs(x_max):
        result.append(y)
        x += h
```

```
45          y = PicarApprox(x)
46
47      return result
48
49
50  def Euler(x_max, h):
51      result = []
52      x, y = 0, 0
53
54      while abs(x) < abs(x_max):
55          result.append(y)
56          y = y + h * f(x, y)
57          x += h
58
59      return result
60
61
62  def RungeKutta(x_max, h):
63      result = []
64      coeff = h / 2
65      x, y = 0, 0
66
67      while abs(x) < abs(x_max):
68          result.append(y)
69          y = y + h * f(x + coeff, y + coeff * f(x, y))
70          x += h
71
72      return result
73
74
75  def generate_x(x_max, step):
76      result = []
77      x = 0
78
79      while abs(x) < abs(x_max):
80          result.append(round(x, 3))
81          x += step
82
83      return result
84
85
86  def print_res_table(x_arr, picar_approx1_arr, picar_approx2_arr,
87                      picar_approx3_arr, picar_approx4_arr,
88                      euler_arr, runge_kutta):
89
90      print("\n%s  X    | PicarApprox1 | PicarApprox2 | PicarApprox3 |
        PicarApprox4 |     Euler     | RungeKutta \n"
```

```
91                 "-------------------------------------------------
                      ---------------------------------------------%s"
92     %(PURPLE, BASE))
93
94
95     for i in range(len(x_arr)):
96         if i % 500 == 0:
97             print("%5.2f %s|%s%12.5f  %s|%s%12.5f  %s|%s%12.5f  %s|%s%12.5f
                   %s|%s%12.5f  %s|%s%12.5f  " \
98             %(x_arr[i],              PURPLE, BASE,
99                 picar_approx1_arr[i], PURPLE, BASE,
100                picar_approx2_arr[i], PURPLE, BASE,
101                picar_approx3_arr[i], PURPLE, BASE,
102                picar_approx4_arr[i], PURPLE, BASE,
103                euler_arr[i],         PURPLE, BASE,
104                runge_kutta[i]
105            ))
106
107    print()
108
109
110 def build_graph(x_arr, picar_approx1_arr, picar_approx2_arr,
111                 picar_approx3_arr, picar_approx4_arr,
112                 euler_arr, runge_kutta):
113
114    fig1 = plt.figure(figsize = (10, 7))
115    plot = fig1.add_subplot()
116    plot.plot(x_arr, picar_approx1_arr,  label = "PicarApprox1")
117    plot.plot(x_arr, picar_approx2_arr,  label = "PicarApprox2")
118    plot.plot(x_arr, picar_approx3_arr,  label = "PicarApprox3")
119    plot.plot(x_arr, picar_approx4_arr,  label = "PicarApprox4")
120    plot.plot(x_arr, euler_arr,          label = "Euler")
121    plot.plot(x_arr, runge_kutta,        label = "RungeKutta")
122
123    plt.legend()
124    plt.grid()
125    plt.title("Сравнение алгоритмом")
126
127    plt.show()
128
129
130 def main():
131
132    x_arr             = generate_x(MAX_X, STEP)
133    picar_approx1_arr = Picar(MAX_X, STEP, PicarApprox1)
134    picar_approx2_arr = Picar(MAX_X, STEP, PicarApprox2)
135    picar_approx3_arr = Picar(MAX_X, STEP, PicarApprox3)
136    picar_approx4_arr = Picar(MAX_X, STEP, PicarApprox4)
```

```
137     euler_arr          = Euler(MAX_X, STEP)
138     runge_kutta        = RungeKutta(MAX_X, STEP)
139 
140     print_res_table(x_arr, picar_approx1_arr, picar_approx2_arr,
141                     picar_approx3_arr, picar_approx4_arr,
142                     euler_arr, runge_kutta)
143 
144     x_arr = generate_x(-MAX_X, -STEP)
145     x_arr.reverse()
146     x_arr.extend(generate_x(MAX_X, STEP))
147 
148     picar_approx1_arr = Picar(-MAX_X, -STEP, PicarApprox1)
149     picar_approx1_arr.reverse()
150     picar_approx1_arr.extend(Picar(MAX_X, STEP, PicarApprox1))
151 
152     picar_approx2_arr = Picar(-MAX_X, -STEP, PicarApprox2)
153     picar_approx2_arr.reverse()
154     picar_approx2_arr.extend(Picar(MAX_X, STEP, PicarApprox2))
155 
156     picar_approx3_arr = Picar(-MAX_X, -STEP, PicarApprox3)
157     picar_approx3_arr.reverse()
158     picar_approx3_arr.extend(Picar(MAX_X, STEP, PicarApprox3))
159 
160     picar_approx4_arr = Picar(-MAX_X, -STEP, PicarApprox4)
161     picar_approx4_arr.reverse()
162     picar_approx4_arr.extend(Picar(MAX_X, STEP, PicarApprox4))
163 
164     euler_arr = Euler(-MAX_X, -STEP)
165     euler_arr.reverse()
166     euler_arr.extend(Euler(MAX_X, STEP))
167 
168     runge_kutta = RungeKutta(-MAX_X, -STEP)
169     runge_kutta.reverse()
170     runge_kutta.extend(RungeKutta(MAX_X, STEP))
171 
172     build_graph(x_arr, picar_approx1_arr, picar_approx2_arr,
173                 picar_approx3_arr, picar_approx4_arr,
174                 euler_arr, runge_kutta)
175 
176 
177 if __name__ == "__main__":
178     main()
```

4 Результаты работы программы

X	PicarApprox1	PicarApprox2	PicarApprox3	PicarApprox4	Euler	RungeKutta
0.00	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000
0.05	0.00004	0.00004	0.00004	0.00004	0.00004	0.00004
0.10	0.00033	0.00033	0.00033	0.00033	0.00033	0.00033
0.15	0.00112	0.00113	0.00113	0.00113	0.00112	0.00113
0.20	0.00267	0.00267	0.00267	0.00267	0.00266	0.00267
0.25	0.00521	0.00521	0.00521	0.00521	0.00521	0.00521
0.30	0.00900	0.00900	0.00900	0.00900	0.00900	0.00900
0.35	0.01429	0.01430	0.01430	0.01430	0.01430	0.01430
0.40	0.02133	0.02136	0.02136	0.02136	0.02135	0.02136
0.45	0.03037	0.03043	0.03043	0.03043	0.03042	0.03043
0.50	0.04167	0.04179	0.04179	0.04179	0.04178	0.04179
0.55	0.05546	0.05570	0.05570	0.05570	0.05569	0.05570
0.60	0.07200	0.07244	0.07245	0.07245	0.07243	0.07245
0.65	0.09154	0.09232	0.09233	0.09233	0.09231	0.09233
0.70	0.11433	0.11564	0.11566	0.11566	0.11563	0.11566
0.75	0.14062	0.14274	0.14278	0.14279	0.14276	0.14279
0.80	0.17067	0.17400	0.17408	0.17408	0.17405	0.17408
0.85	0.20471	0.20980	0.20996	0.20996	0.20992	0.20996
0.90	0.24300	0.25059	0.25090	0.25091	0.25086	0.25091
0.95	0.28579	0.29688	0.29743	0.29745	0.29740	0.29745
1.00	0.33333	0.34921	0.35019	0.35023	0.35017	0.35023

Рисунок 4.1 – Демонстрация работы программы

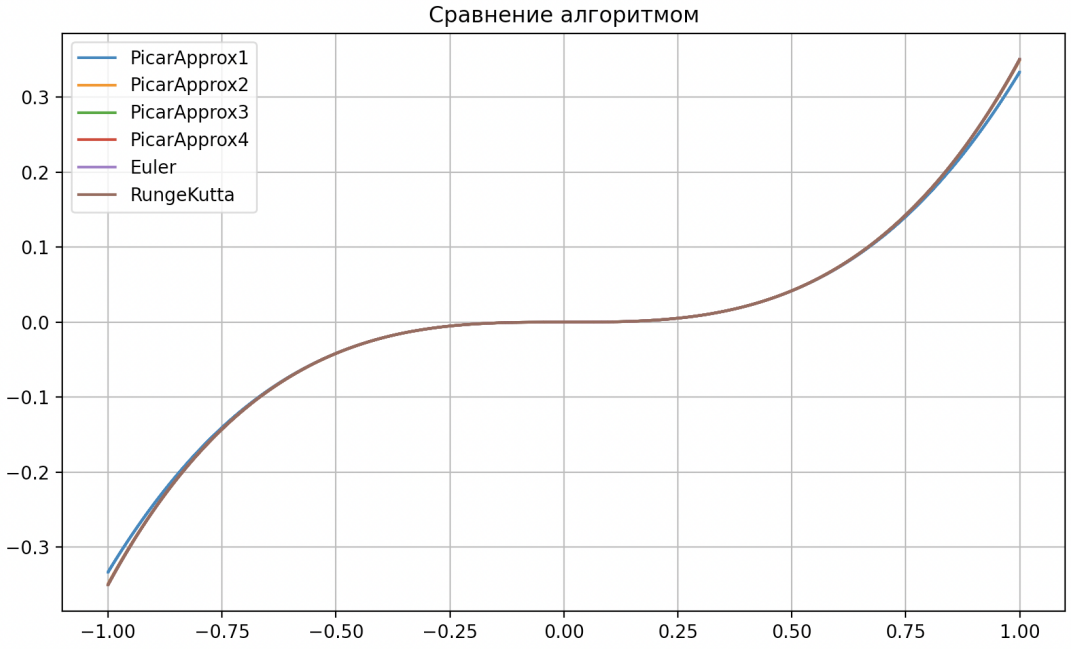

Рисунок 4.2 – График функции

5 Ответы на контрольные вопросы

5.1 Вопрос 1

5.1.1 Задание

Укажите интервалы значений аргумента, в которых можно считать решением заданного уравнения каждое из первых 4-х приближений Пикара, т.е. для каждого приближения указать свои границы применимости. Точность результата оценивать до второй цифры после запятой. Объяснить свой ответ.

5.1.2 Ответ

Для того, чтобы указать интервалы значений аргумента, в которых можно считать решением заданного уравнения каждое из первых 4-х приближений проанализируем полученные значения. Так как нам дано начальное приближение, то левой границей будет 0. Для определения правой границы границы мы будем анализировать полученные решения методом Пикара для конкретного приближения и сравнивать со значениями более высоких порядков приближения и с результатами численных методов при определенном шаге.

- Для 1-го приближения искомым интервалом будет [0, 0.89].
- Для 2-го приближения искомым интервалом будет [0, 1.12].
- Для 3-го приближения искомым интервалом будет [0, 1.34].
- Для 4-го приближения искомым интервалом будет [0, 1.4].

5.2 Вопрос 2

5.2.1 Задание

Пояснить, каким образом можно доказать правильность полученного результата при фиксированном значении аргумента в численных методах.

5.2.2 Ответ

В численных методах правильность полученного результата, при фикси рованном значении аргумента, доказывается путем уменьшения шага. Правильно полученный результат – это когда при уменьшение шага значение аргумента незна чительно (или вообще) не меняется.

5.3 Вопрос 3

5.3.1 Задание

Каково значение решения уравнения в точке x = 2, т.е. привести значение u(2).

5.3.2 Ответ

Примерно 317.490

5.4 Вопрос 4

5.4.1 Задание

Дайте оценку точки разрыва решения уравнения.

5.5 Вопрос 5

5.5.1 Задание

Покажите, что метод Пикара сходится к точному аналитическому решению уравнения

$$\begin{cases} u'(x) = x^2 + u \\ u(0) = 0 \end{cases} \tag{5.1}$$