**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ФАКУЛЬТЕТ ПРИКЛАДНОЙ МАТЕМАТИКИ И ИНФОРМАТИКИ**

**Кафедра вычислительной математики**

**Отчёт**

**Лабораторная работа**

«Методы решения ОДУ»

Вариант №12

Снежко Льва Владимировича

студента 3 курса, 3 группы

специальности «Информатика»

дисциплина «Численные методы»

Минск, 2025

**Постановка задачи**

Дана задача Коши вида:

Необходимо:

* Найти приближенное решение задачи Коши на сетке узлов при 10-ти разбиениях отрезка интегрирования, применяя
* Явный метод Эйлера;
* Явный метод Рунге-Кутта 2-го порядка ;
* Явный метод Адамса 2-го порядка.
* Используя таблицу результатов, получить погрешности методов, сравнивая приближенное решение с точным.
* Оценивая величину истинной погрешности, сделать вывод о точности каждого используемого метода.

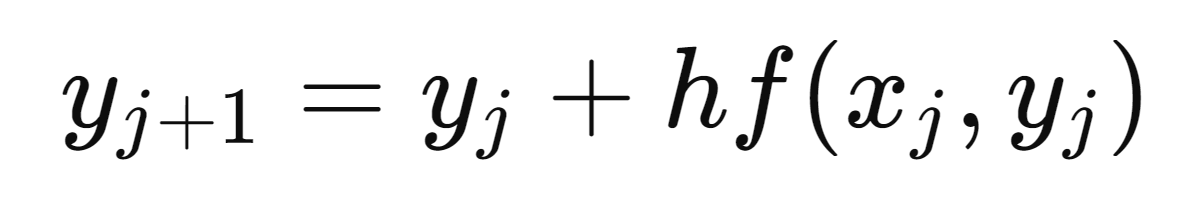
**Алгоритм решения**

**Явный метод Эйлера.**

Пусть:

* , где j=0,1,2,…,N , ;
* h = = = 0.1— шаг;
* ​ — приближённое значение функции y().

Тогда приближённое решение вычисляется по формуле:



j=

**Явный метод Рунге-Кутта 2-го порядка ().**

Пусть:

* , где j=0,1,2,…,N , ;
* h = = = 0.1— шаг;
* ​ — приближённое значение функции y().

При приближённое решение вычисляется по следующей формуле:

j=

**Явный метод Адамса 2-го порядка.**

Пусть:

* , где j=0,1,2,…,N , ;
* h = = = 0.1— шаг;
* ​ — приближённое значение функции y().

Тогда приближённое решение вычисляется по формуле:

,

j=

Причём, находим с помощью м. Р-К.

**Листинг кода**

import numpy as np

import matplotlib.pyplot as plt

# Правая часть дифференциального уравнения

def f(x, y):

return y \* x + x \* (0.7 \* np.exp(x) + 0.3 \* np.cos(x))

# Точное решение

def exact\_solution(x):

return 0.7 \* np.exp(x) + 0.3 \* np.sin(x)

# Начальные условия

x0 = 0.7

x\_end = 1.7

y0 = exact\_solution(x0) # по условию

# Сетка

n = 10

h = (x\_end - x0) / n

x = np.linspace(x0, x\_end, n+1)

# Метод Эйлера

y\_euler = np.zeros(n+1)

y\_euler[0] = y0

for i in range(n):

y\_euler[i+1] = y\_euler[i] + h \* f(x[i], y\_euler[i])

# Метод Рунге-Кутты 2-го порядка (β = 3/4)

beta = 3/4

alpha = 1 / (2 \* beta)

y\_rk2 = np.zeros(n+1)

y\_rk2[0] = y0

for i in range(n):

k1 = f(x[i], y\_rk2[i])

k2 = f(x[i] + beta\*h, y\_rk2[i] + beta\*h\*k1)

y\_rk2[i+1] = y\_rk2[i] + h \* (alpha \* k1 + (1 - alpha) \* k2)

# Метод Адамса–Башфорта 2-го порядка (нужны первые два значения, используем РК2)

y\_adams = np.zeros(n+1)

y\_adams[0] = y0

# Второе значение из РК2

k1 = f(x[0], y\_adams[0])

k2 = f(x[0] + beta\*h, y\_adams[0] + beta\*h\*k1)

y\_adams[1] = y\_adams[0] + h \* (alpha \* k1 + (1 - alpha) \* k2)

for i in range(1, n):

y\_adams[i+1] = y\_adams[i] + h/2 \* (3 \* f(x[i], y\_adams[i]) - f(x[i-1], y\_adams[i-1]))

# Точное решение

y\_exact = exact\_solution(x)

# Погрешности

err\_euler = np.abs(y\_exact - y\_euler)

err\_rk2 = np.abs(y\_exact - y\_rk2)

err\_adams = np.abs(y\_exact - y\_adams)

# Таблица результатов

print(f"{'x':>5} {'Точное':>12} {'Эйлер':>12} {'Погреш.':>10} {'РК2':>12} {'Погреш.':>10} {'Адамс':>12} {'Погреш.':>10}")

for i in range(n+1):

print(f"{x[i]:5.2f} {y\_exact[i]:12.6f} {y\_euler[i]:12.6f} {err\_euler[i]:10.2e} {y\_rk2[i]:12.6f} {err\_rk2[i]:10.2e} {y\_adams[i]:12.6f} {err\_adams[i]:10.2e}")

**Сравнительный анализ полученных результатов**



Метод Эйлера является самым простым из рассмотренных и относится к первому порядку точности. Его глобальная погрешность пропорциональна шагу h, то есть линейно возрастает при увеличении длины интервала. На первых шагах метод даёт приемлемую точность, однако с ростом x ошибка значительно увеличивается. Это связано с тем, что метод опирается только на значение производной в начале шага и не учитывает изменения производной в течение шага, что особенно важно при экспоненциальном росте решения.

Метод Рунге–Кутта второго порядка обладает более высоким порядком точности: его глобальная ошибка уменьшается пропорционально квадрату шага. Этот метод учитывает значение производной не только в начале, но и внутри шага, что делает его более точным на изогнутых участках решения. Несмотря на это, при фиксированном шаге 0.1 и достаточно быстро растущем решении даже метод Рунге–Кутта даёт заметную погрешность, которая к концу интервала достигает более 6 единиц.

Метод Адамса–Башфорта второго порядка также имеет второй порядок точности, но в отличие от Рунге–Кутты, является многошаговым. Это означает, что он использует значения производной на двух предыдущих шагах для предсказания следующего значения. Такой подход требует точных начальных значений, обычно получаемых с помощью одношагового метода. В данной задаче в качестве стартового значения использовано приближение методом Рунге–Кутты. Ошибка, накопленная на первом шаге, передаётся на следующие, и это приводит к наибольшей ошибке среди всех трёх методов на последних узлах сетки. Таким образом, несмотря на формально одинаковый порядок точности с методом Рунге–Кутты, метод Адамса в данной реализации показал худший результат.

Таким образом, поведение ошибок согласуется с теоретическими оценками точности методов. Это подчёркивает важность выбора метода и шага интегрирования в зависимости от поведения решения и требований к точности.