**Лабораторная работа №4**

**Численное решение систем дифференциальных уравнений первого порядка**

**Вариант 11**

**Чеботаревский Никита**

**3 курс, 8 группа**

**Постановка задачи**



Решить задачу Коши для дифференциального уравнения второго порядка

методом Рунге-Кутта на равномерной сетке отрезка [*a*, *b*]. Вычисления произвести дважды с шагами *h* и *h*/2, полагая *h =* 0.2. Найти численное решение дифференциального уравнения и оценить его погрешность с помощью правила Рунге. Сравнить численное решение с известным

аналитическим решением.

По результатам лабораторной работы оформляется отчёт, в котором должна быть приведена следующая информация:

1) постановку задачи c учетом предложенного варианта;

2) краткие теоретические сведения;

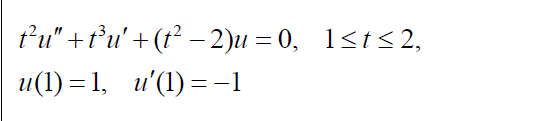
3) результаты решения поставленной задачи;

4) выводы;

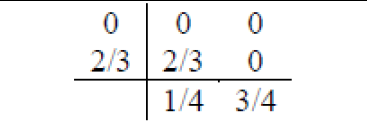
5) листинг программы с комментариями

**Теоретические сведения:**

У нас есть следующая задача Коши:



Её нужно решить методом Рунге-Кутта, для этого нам дана таблица Батчера:



Для решения задачи введём следующую замену:

Откуда наша исходная задача Коши примет вид:

По итогу нам надо будет решить следующую систему:

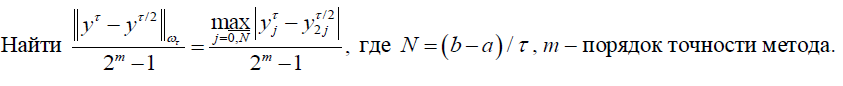
Где

.

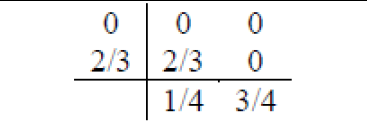
Для сравнения численного решения с аналитическим воспользуемся формулой:



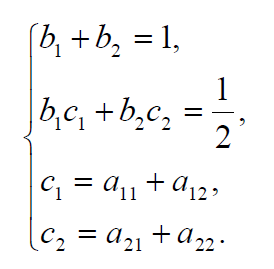
А для оценки погрешности по правилу Рунге воспользуемся следующей формулой:



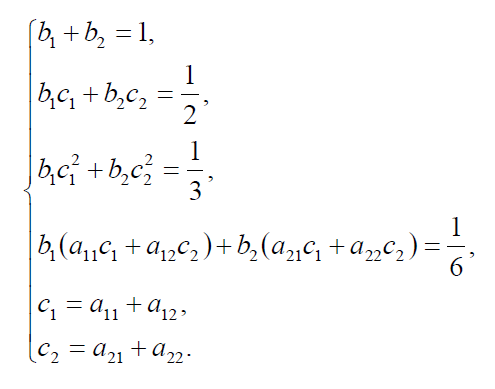
Теперь определим порядок точности нашего метода.



Учитывая, что мы имеем неявный двухстадийный метод, то для второго порядка для него должны выполняться условия:



Теперь проверим условия третьего порядка:



Первые два и последние два условия выполняются проверим третье и четвёртое:

Т.е получается мы имеем метод Рунге-Кутта второго порядка.

**Листинг программы**

an = [[0, 0], [2 / 3, 0]]  
bn = [1 / 4, 3 / 4]  
cn = [0, 2 / 3]  
  
  
*class* RungeKutta:  
  
 *def* \_\_init\_\_(*self*, a, b, h):  
 *self*.a = a  
 *self*.b = b  
 *self*.h = h  
  
 *def* t(*self*):  
 n = int((*self*.b - *self*.a) / *self*.h)  
 result = [*self*.a + i \* *self*.h *for* i *in* range(n + 1)]  
 *return* result  
  
 @staticmethod  
 *def* u(t):  
 *return* 1 / t  
  
 @staticmethod  
 *def* f1(y2):  
 *return* y2  
  
 @staticmethod  
 *def* f2(tj, y1, y2):  
 *return* -tj \* y2 - ((tj \*\* 2 - 2) / tj \*\* 2) \* y1  
  
 *def* k(*self*, tj, y1, y2, a, c):  
 k = [[*self*.f1(y2), *self*.f2(tj, y1, y2)], []]  
 y2\_21 = y2 + *self*.h \* a[1][0] \* k[0][1]  
 k[1].append(*self*.f1(y2\_21))  
 y1\_22 = y1 + *self*.h \* a[1][0] \* k[0][0]  
 k[1].append(*self*.f2(tj + c[1] \* *self*.h, y1\_22, y2\_21))  
 *return* k  
  
 *def* runge\_kutta(*self*, a, b, c):  
 t\_j = *self*.t()  
 n = int((*self*.b - *self*.a) / *self*.h)  
 u1 = [1]  
 u2 = [-1]  
 *for* i *in* range(n):  
 summary\_1 = 0  
 summary\_2 = 0  
 ks = *self*.k(t\_j[i], u1[i], u2[i], a, c)  
 *for* j *in* range(len(b)):  
 summary\_1 += (b[j] \* ks[j][0])  
 summary\_2 += (b[j] \* ks[j][1])  
 summary\_1 \*= *self*.h  
 summary\_2 \*= *self*.h  
  
 u1.append(u1[i] + summary\_1)  
 u2.append(u2[i] + summary\_2)  
  
 *return* u1  
  
 @staticmethod  
 *def* count\_accuracy(u, y):  
 result = []  
 *for* i *in* range(len(u)):  
 result.append(abs(u[i] - y[i]))  
 *return* max(result)  
  
 @staticmethod  
 *def* count\_runge\_accuracy(y1, y2, m):  
 result = []  
 *for* i *in* range(len(y1)):  
 result.append(np.abs(y1[i] - y2[2 \* i]))  
 *return* max(result) / (2 \*\* m - 1)  
  
 @classmethod  
 *def* show\_result(*cls*, left, right, step, a, b, c):  
 rg = RungeKutta(left, right, step)  
 u1 = rg.runge\_kutta(a, b, c)  
 ts = rg.t()  
 result\_function = [rg.u(t) *for* t *in* ts]  
 print(f"Численное решение задачи Коши: {u1}")  
 print(f"Точное решение: {result\_function}")  
 print(f"Погрешность: {*cls*.count\_accuracy(result\_function, u1)}", end="\n" \* 2)  
  
  
RungeKutta.show\_result(1, 2, 0.2, an, bn, cn)  
RungeKutta.show\_result(1, 2, 0.1, an, bn, cn)  
  
rg1 = RungeKutta(1, 2, 0.2)  
u1 = rg1.runge\_kutta(an, bn, cn)  
rg2 = RungeKutta(1, 2, 0.1)  
u2 = rg2.runge\_kutta(an, bn, cn)  
  
print(f"Погрешность по правилу Рунге: {RungeKutta.count\_runge\_accuracy(u1, u2, m=2)}")

**Результаты**

Численное решение задачи Коши: [1, 0.84, 0.7228209642445214, 0.6334984044328947, 0.5632844677037105, 0.5067506840764575]

Точное решение: [1.0, 0.8333333333333334, 0.7142857142857143, 0.625, 0.5555555555555556, 0.5]

Погрешность: 0.008535249958807056

Численное решение задачи Коши: [1, 0.91, 0.8347605908445248, 0.7709340556176868, 0.7161120089942984, 0.6685186416594087, 0.6268163449595352, 0.5899786042948943, 0.5572045652161047, 0.5278602074232394, 0.5014369673790298]

Точное решение: [1.0, 0.9090909090909091, 0.8333333333333334, 0.7692307692307692, 0.7142857142857143, 0.6666666666666666, 0.625, 0.588235294117647, 0.5555555555555556, 0.5263157894736842, 0.5]

Погрешность: 0.0018519749927420337

Погрешность по правилу Рунге: 0.00223631841674099

**Вывод**

С помощью метод Рунге-Кутта можно также решать задачи Коши и для уравнений высших порядков, в нашем случае второго порядка. Наш метод также имеет порядок точности 2, и благодаря ему мы находим решение задачи Коши с точностью