МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

**Компьютерный практикум по курсу**

**«ВВЕДЕНИЕ В ЧИСЛЕННЫЕ МЕТОДЫ»**

**ЗАДАНИЕ № 2 (1)**

**«РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ ПЕРВОГО ПОРЯДКА И СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА»**

**ОТЧЕТ**

**о выполненном задании**

студента ***206*** учебной группы факультета ВМК МГУ

***Щеголихина Максима Сергеевича***

Москва, 2015 г.

**0.Вариант задания**

Подвариант 1: приложение 1-6, приложение 2-17.

**1.Цель работы**

Освоить методы Рунге-Кутта второго и четвёртого порядка точности, применимые для численного решения задачи Коши для дифференциального уравнения (или системы диыыеренциальных уравнений) первого порядка.

**2.Постановка задачи**

Рассматривается обыкновенное дифференциальное уравнение первого порядка, разрешенное относительно производной и имеющее вид:



с дополнительным начальным условием, заданным в точке *х =* *х0*:



Предполагается, что правая часть уравнения (1) функция *f = f(x,у)* такова, что гарантирует существование и единственность решения задачи Коши (1)-(2).

В том случае, если рассматривается не одно дифференциальное уравнение вида (1), а система обыкновенных дифференциальных уравнений первого порядка, разрешенных относительно производных неизвестных функций, то соответствующая задача Коши имеет вид (на примере двух дифференциальных уравнений):



Дополнительные (начальные) условия задаются в точке х = х0:



Также предполагается, что правые части уравнений из (3) заданы так, что это гарантирует существование и единственность решения задачи Коши (3)-(4), но уже для системы обыкновенных дифференциальных уравнений первого порядка в форме, разрешенной относительно производных неизвестных функций.

## Необходимо:

1. Решить задачу Коши (1)-(2) (или (3)-(4)) наиболее известными и широко используемыми на практике методами Рунге-Кутта второго и четвертого порядка точности, аппроксимировав дифференциальную задачу соответствующей разностной схемой(на равномерной сетке); полученное конечно-разностное уравнение (или уравнения в случае системы), представляющие фактически некоторую рекуррентную формулу, просчитать численно;
2. Найти численное решение задачи и построить его график;
3. Найденное численное решение сравнить с точным решением дифференциального уравнения.

**3.Метод(алгоритм) решения**

Метод Рунге-Кутты является наиболее распространенным методом решения обыкновенных дифференциальных уравнений. Существуют разностные схемы разного порядка точности, построенные на основе этого метода. Приведем метод Рунге-Кутты четвертого порядка:

*yi+1 = yi + (k0 + 2k1 + 2k2 + k3)/6,*

*k0 = hf(xi , yi), k1 = hf(xi + h/2, yi + k0/2),*

*k2 = hf(xi + h/2, yi + k1/2),*

*k3 = hf(xi + h, yi + k2).*

Метод явный, одношаговый. Имеет погрешность на шаге O (h\*\*5), а значит, глобальную погрешность O(h\*\*4). Метод Рунге-Кутты требует на каждом шаге четырехкратного вычисления правой части уравнения. Но это окупается повышенной точностью, что дает возможность проводить расчет с более крупным шагом.

**4.Код программы(для одного уравнения)**

from math import \*

def f(x, y):

return (x - x\*\*2) \* y

class CF:

def \_\_init\_\_(self, f):

self.f = f

def step(self, x, y, h):

k0 = h\*self.f(x, y)

k1 = h\*self.f(x + h / 2, y + k0 / 2)

k2 = h\*self.f(x + h / 2, y + k1 / 2)

k3 = h\*self.f(x + h, y + k2)

return (k0 + 2 \* k1 + 2 \* k2 + k3) / 6

b = float(input("input b: "))

n = 100

h = (b - a) / n

x, y = 0, 1

Y = CF(f)

while x <= b:

y += Y.step(x, y, h)

x += h

print(x, y)

**5.Код программы(для системы уравнений)**

from math import \*

def f1(x, u, v):

return sin(1.4 \* u\*\*2) - x + v

def f2(x, u, v):

return x + u - 2.2 \* v\*\*2 + 1

class CF:

def \_\_init\_\_(self, f1, f2):

self.f1 = f1

self.f2 = f2

def step(self, x, u, v, h):

k0 = h \* self.f1(x, u, v)

m0 = h \* self.f2(x, u, v)

k1 = h\*self.f1(x + h / 2, u + k0 / 2, v + k0 / 2)

m1 = h\*self.f2(x + h / 2, u + k0 / 2, v + k0 / 2)

k2 = h\*self.f1(x + h / 2, u + k1 / 2, v + k1 / 2)

m2 = h\*self.f2(x + h / 2, u + k1 / 2, v + k1 / 2)

k3 = h\*self.f1(x + h, u + k2, v + k2)

m3 = h\*self.f2(x + h, u + k2, v + k2)

return [(k0 + 2 \* k1 + 2 \* k2 + k3) / 6, (m0 + 2 \* m1 + 2 \* m2 + m3) / 6]

a, b =0, float(input("input b: "))

n = 100

h = (b - a) /n

x, u, v = 0, 1, 0.5

U1 = CF(f1, f2)

V2 = CF(f1, f2)

while x <= b:

u += U1.step(x, u, v, h)[0]

v += V2.step(x, u, v, h)[1]

x += h

print(x, u, v)

**6.Тесты**

|  |  |  |
| --- | --- | --- |
| ***x*** | ***вывод программы*** | ***e\*\*(x\*\*2 \* (3 - 2\*x) / 6)*** |
| 0.1 | 1.0046775690052083 | 1.004677572513607 |
| 0.2 | 1.0174844200939863 | 1.0174844272803685 |
| 0.3 | 1.0366558354113649 | 1.0366558464909237 |
| 0.4 | 1.0604216927173233 | 1.060421707918656 |
| 0.5 | 1.0869040299683457 | 1.086904049521229 |
| 0.6 | 1.1140477212741684 | 1.1140477453864677 |
| 0.7 | 1.1395878265544555 | 1.13958785537606 |
| 0.8 | 1.1610599111211029 | 1.1610599446944865 |
| 0.9 | 1.1758602031208547 | 1.1758602413209998 |
| 1.0 | 1.1813603703614308 | 1.1813604128656459 |
| 1.1 | 1.1750765493394613 | 1.1750765957376708 |

**7.Литература**

Костомаров Д.П., Фаворский А.П.. Вводные лекции по численным методам.

Костомаров Д.П. Введение в численные методы. Методическое пособие для 2 курса.