МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

имени М.В.Ломоносова

Факультет вычислительной математики и кибернетики

Компьютерный практикум по учебному курсу

«ВВЕДЕНИЕ В ЧИСЛЕННЫЕ МЕТОДЫ»

ЗАДАНИЕ № 2.

Подвариант № 1.

РЕШЕНИЕ ЗАДАЧИ КОШИ ДЛЯ ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ ПЕРВОГО ПОРЯДКА ИЛИ СИСТЕМЫ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ ПЕРВОГО ПОРЯДКА

ОТЧЕТ

о выполненном задании

студента 202 учебной группы факультета ВМК МГУ

Плеханова Антона Дмитриевича

гор. Москва

2020 г.

Содержание

Цель работы

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

Постановка задачи

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

$$\frac{dy}{dx} = f(x, y),\ x_0 < x \tag{1}$$

с дополнительным начальным услвоием, заданным в точке $x = x_0$:

$$y(x_0) = y_0 \tag{2}$$

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

Также рассматривается система обыкновенных дифференциальных уравнений первого порядка, разрешенных относительно производных неизвестных функций:

$$\begin{cases} \frac{dy_1}{dx} = f_1(x, y_1, y_2), \\ \frac{dy_2}{dx} = f_2(x, y_1, y_2), \quad x > x_0. \end{cases} \tag{3}$$

Дополнительные (начальные) условия задаются в точке $x = x_0$:

$$y_1(x_0) = y_1^{(0)},\ \ y_2(x_0) = y_2^{(0)} \tag{4}$$

Также предполагается, что правые части урванения 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 неизвезстных функций. Требуется решить задачи Коши (1)-(2) и (3)-(4).

Цели практической работы

1. Решить задачи Коши (1)-(2) и (3)-(4) метом Рунге-Кутта 2 и 4 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 (подобрать специальные тесты, где аналитические решения находятся в классе элементарных функций)

Алгоритм

Пусть $u(x)$ - решение задачи Коши (1)-(2) на некотором отрезке $[x_0, x_0 + l]$. Рассмотрим сеточную функцию $y_i = u(x_i)$ на равномерной сетке $x_i = x_0 + ih$, $h = \frac{l}{n}$, где n - количество шагов. Разложим функцию $u(x)$ по формуле Тейлора до второго порядка:

$$u_{i+1} = u(x_i + h) = u_i + u'(x_i)h + \frac{1}{2}u''(x_i)h^2 + O(h^2)$$

Поскольку

$$u''(x) = \frac{\partial f}{\partial x}(x, y) + \frac{\partial f}{\partial x}(x, y)f(x, y),$$

то при подстановке этого равенства в предыдущее с учетом равенства $y_i = u(x_i)$ получим:

$$\frac{y_{i+1} - y_i}{h} = f(x_i, y_i) + \frac{1}{2}\left(\frac{\partial f}{\partial x}(x, y) + \frac{\partial f}{\partial x}(x, y)f(x, y)\right)h$$

Идея метода Рунге-Кутта состоит в том, чтобы приближенно заменить правую часть формулы на сумму значений функции f в двух разных точках с точностью до членов порядка h^2.
Проведя соответствующие действия, получим однопараметрическое семейство разностных схем Рунге-Кутта:

$$\frac{y_{i+1} - y_i}{h} = (1 - \alpha)f(x_i, y_i) + \alpha f(x_i + \frac{h}{2\alpha}, y_i + \frac{h}{2\alpha}f(x_i, y_i))$$

Или, в виде рекуррентного соотношения:

$$y_{i+1} = y_i + \left[(1 - \alpha)f(x_i, y_i) + \alpha f(x_i + \frac{h}{2\alpha}, y_i + \frac{h}{2\alpha}f(x_i, y_i))\right]h$$

Удобная для расчета разностная схема получается при $\alpha = 0.5$.

$$y_{i+1} = y_i + \frac{h}{2}\left[f(x_i, y_i) + f(x_i + h, y_i + hf(x_i, y_i))\right]$$

Для метода Рунге-Кутта 4 порядка точности используется следующая формула:

$$\frac{y_{i+1} - y_i}{h} = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4),$$

где

$$k_1 = f(x_i, y_i),\ \ k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right),$$

$$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right),\ \ k_4 = f(x_i + h, y_i + hk_3).$$

Приведенные выше формулы также верны и для векторного представления $y = (y^{(1)}, y^{(2)},\ ..., y^{(k)})$

Описание программы

Программа написана на языке Python3. Для ее работы необходимо наличие установленной библиотеки numpy.
При запуске программы пользователь должен выбрать вариант задания. Всего 4 варианта: 1-2 - решение обыкновенного дифференциального уравнения, 3-4 - решение системы обыкновенных дифференциальных уравнений).
Затем пользователю будет предложено ввести порядок точности метода Рунге-Кутта (2 или 4), размер шага сетки и их количество.
Программа выведет в зависимости от варианта (одно уравнение или система) 2 или 3 столбца чисел. Первый столбец - значения точек сетки, второй (третий) - значения найденной сеточной функции.
В программе реализованы следующие функции (реализацию функций с комментариями можно посмотреть в листинге программы ниже):

- $RungeKutt2Method(f, x\_start, y\_start, h, step\_cnt)$ - в этой функции реализован метод Рунге-Кутта второго порядка точности. Функция принимает на вход функцию f из правой части уравнения (в случае системы - столбец-функцию), начальную точку x_start, значение функции в данной точке y_start, размер шага сетки h и количество шагов $step\_cnt$;

- $RungeKutt4Method(f, x\_start, y\_start, h, step\_cnt)$ - в этой функции реализован метод Рунге-Кутта четвертого порядка точности. Аргументы этой функции аналогичны аргументам предыдущей;

- $f(x, y)$ - правая часть уравнения из таблицы 1, варианта 2

- $f1(x, u, v)$ - правая часть первого уравнения таблицы 2, варианта 8

- $f2(x, u, v)$ - правая часть второго уравенининия таблицы 2, варианта 8

- $F(x, y)$ - столбец-функция с компонентами $f1(x, u, v)$, $f2(x, u, v)$, причем $y = (u, v)$.

- $g(x, y)$ - правая часть уравнения из таблицы 1, варианта 4

- $g1(x, u, v)$ - правая часть первого уравнения таблицы 2, варианта 2

- $g2(x, u, v)$ - правая часть второго уравенининия таблицы 2, варианта 2

- $G(x, y)$ - столбец-функция с компонентами $g1(x, u, v)$, $g2(x, u, v)$, причем $y = (u, v)$.

Код программы

```python
# coding: utf-8

import math
import numpy as np

# ## Метод РунгеКутта- 2 и 4 порядков точности

def RungeKutt2Method(f, x_start, y_start, h, step_cnt):
    # сетка
    x = np.arange(x_start, x_start + h * (step_cnt + 1), h)

    # размер сетки
    n = x.shape[0]

    # сеточная функция
    y = np.zeros((n, y_start.shape[0]))
    y[0] = y_start

    # вычисление компонент сеточной функции
    for i in range(1, n):
        # предсказание""
        y[i] = y[i - 1] +  f(x[i - 1], y[i - 1]) * h

        # корректировка""
        y[i] = y[i - 1] + h / 2 * (f(x[i - 1], y[i - 1]) + f(x[i], y[i]))

    return (x, y)


def RungeKutt4Method(f, x_start, y_start, h, step_cnt):
    # сетка
    x = np.arange(x_start, x_start + h * (step_cnt + 1), h)

    # размер сетки
    n = x.shape[0]

    # сеточная функция
    y = np.zeros((n, y_start.shape[0]))
    y[0] = y_start

    # вычисление компонент сеточной функции
    for i in range(1, n):
        # вычисляем вспомогательные значения функции f
        k1 = f(x[i-1], y[i-1])
        k2 = f(x[i-1] + h/2, y[i-1] + h/2 * k1)
        k3 = f(x[i-1] + h/2, y[i-1] + h/2 * k2)
        k4 = f(x[i-1] + h, y[i-1] + h * k3)

        # вычисляем саму компоненту сеточной функции
        y[i] = y[i-1] + h/6 * (k1 + 2*k2 +2*k3 + k4)

    return (x, y)
```

```


# ## Функции из варианта задания:

# ### Основные тесты

# Таблица 1, вариант 2
def f(x, y):
    return math.sin(x) - y

# Таблица 2, вариант 8
def f1(x, u, v):
    return math.cos(x + 1.1*v) + u

def f2(x, u, v):
    return -v**2 + 2.1*u + 1.1

def F(x, y):
    return np.array([f1(x, y[0], y[1]), f2(x, y[0], y[1])])


# ### Дополнительные тесты

# Таблица 1, вариант 4
def g(x, y):
    return y - y*x

# Таблица 2, вариант 2
def g1(x, u, v):
    return x * u - v

def g2(x, u, v):
    return u - v

def G(x, y):
    return np.array([g1(x, y[0], y[1]), g2(x, y[0], y[1])])


def main():
    print("Варианты 1, 2 - задача Коши для обыкновенного дифференциального уравнения")
    print("Варианты 3, 4 - задача Коши для системы обыкновенных дифференциальных уравнений")
    print("Выберите вариант задачи (1, 2, 3, 4)")

    var = int(input())

    if var == 1:
        print("Уравнение: y' = sin(x) - y\начальноеn условие y(0) = 10")
        x_start = 0
        y_start = np.array([10])
        func = f
    elif var == 2:
        print("Уравнение: y' = y - yx\начальноеn условие: y(0) = 5")
        x_start = 0
```

```
        y_start = np.array([5])
        func = g
    elif var == 3:
        print("Система уравнений:")
        print("y1' = cos(x + 1.1 * (y2)) + y1")
        print("y2' = -(y2)^2 + 2.1 * y1 + 1.1")
        print("Начальные условия: y1(0) = 0.25, y2(0) = 1")
        x_start = 0
        y_start = np.array([0.25, 1])
        func = F
    elif var == 4:
        print("Система уравнений:")
        print("y1' = x*y1 + y2")
        print("y2' = y1 - y2")
        print("Начальные условия: y1(0) = 0, y2(0) = 1")
        x_start = 0
        y_start = np.array([0, 1])
        func = G
    else:
        print("Такого варианта нет")
        return

    print("Введите порядок точности метода РунгеКутта-:")
    accur_order = int(input())

    if accur_order != 2 and accur_order != 4:
        print("Доступные порядки точности: 2, 4")
        return

    RungeKuttMethod = {2: RungeKutt2Method, 4: RungeKutt4Method}

    print("Введите размер шага:")
    h = float(input())
    print("Введите количество шагов сетки:")
    step_cnt = int(input())

    x, y = (RungeKuttMethod[accur_order])(func, x_start, y_start, h, step_cnt)

    if (var < 3):
        print("Первый столбец - сетка, второй столбец - значения сеточной функции")
        for x_coord, y_coord in zip(x, y):
            print(x_coord, y_coord[0])
    else:
        print("Первый столбец - сетка, второй и третий столбцы - значения сеточных функций y1 и y2")
        for x_coord, y_coord in zip(x, y):
            print(x_coord, y_coord[0], y_coord[1])

    return

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

Тестирование программы

Тесты из условия заданаия

Таблица 1, вариант 2

Задача Коши:

$$y' = \sin x - y, \;\; y(0) = 10$$

Аналитическое решение:

$$y = -\frac{1}{2}\cos x + \frac{1}{2}\sin x + 10.5e^{-x}$$

В результате работы программы были получены сеточные функции, отображенные на графиках (h - размер шага сетки, n - количество шагов сетки):

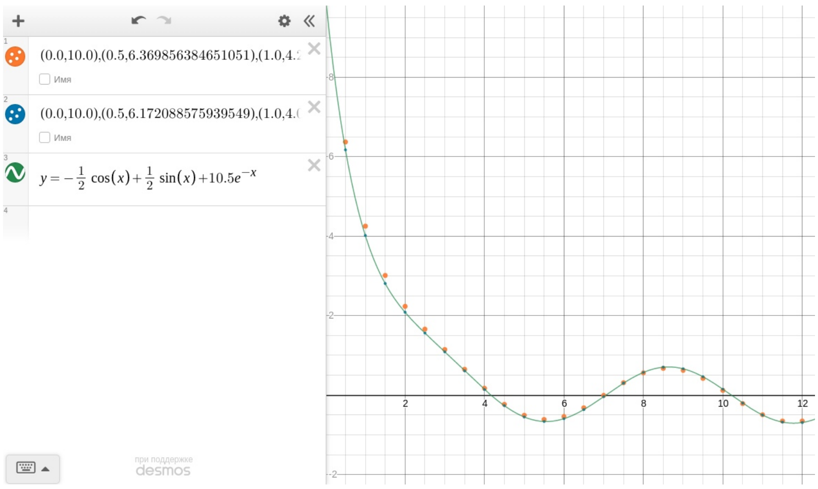

Рис. 1: $h = 0.5, n = 100$

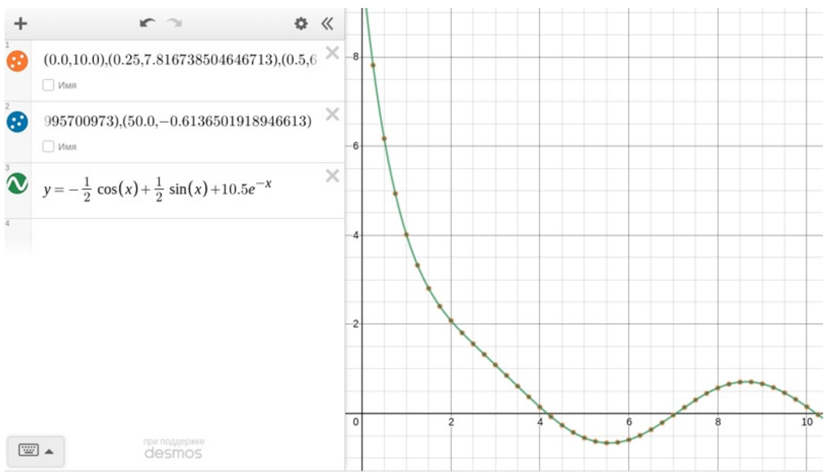

Рис. 2: $h = 0.25, n = 200$

Зеленая линия - аналитическое решение, синие точки - решение 2 порядка точности, оранжевые - 4 порядка точности.

Таблица 2, вариант 8

Задача Коши:

$$\begin{cases} y_1' = \cos\left(x + 1.1y_2\right) + y_1, \\ y_2' = -y_2^2 + 2.1y_1 + 1.1 \end{cases} \quad y_1(0) = 0.25,\ y_2(0) = 1$$

В результате работы программы были получены сеточные функции, отображенные на графиках (размер шага сетки - 0.25, количество шагов сетки - 15): Синие точки - решение 2 порядка точности, оранжевые - 4 порядка.

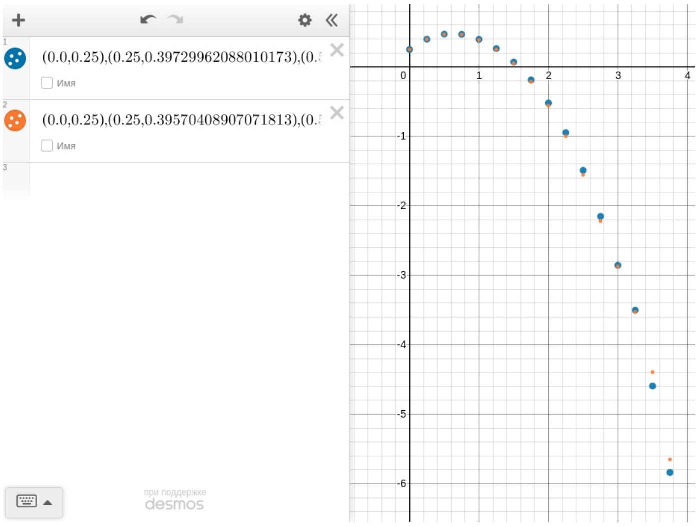

Рис. 3: $y = y_1(x)$

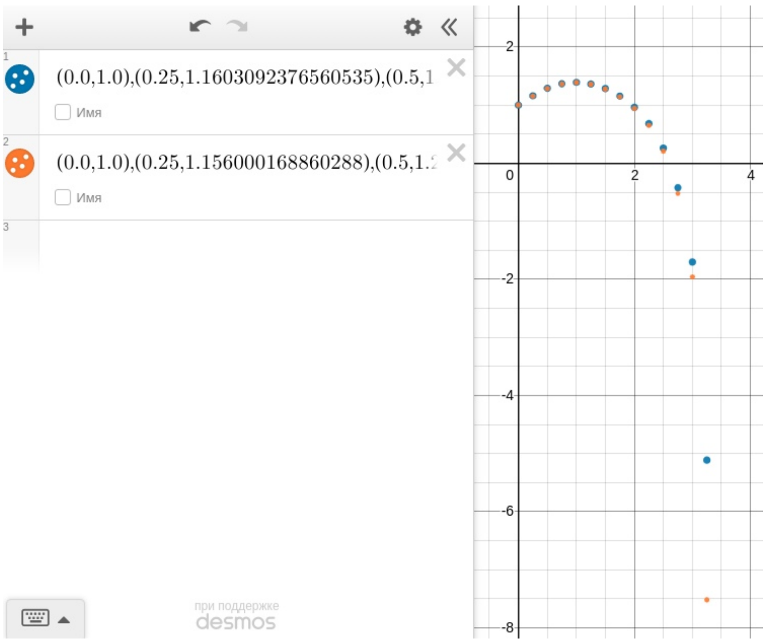

Рис. 4: $y = y_2(x)$

Дополнительные тесты

Таблица 1, вариант 4

Задача Коши:

$$y' = y - yx, \ \ y(0) = 5$$

Аналитическое решение:

$$y = 5e^{-\frac{1}{2}x(x-2)}$$

В результате работы программы были получены сеточные функции, отображенные на графиках (h - размер шага сетки, n - количество шагов сетки):

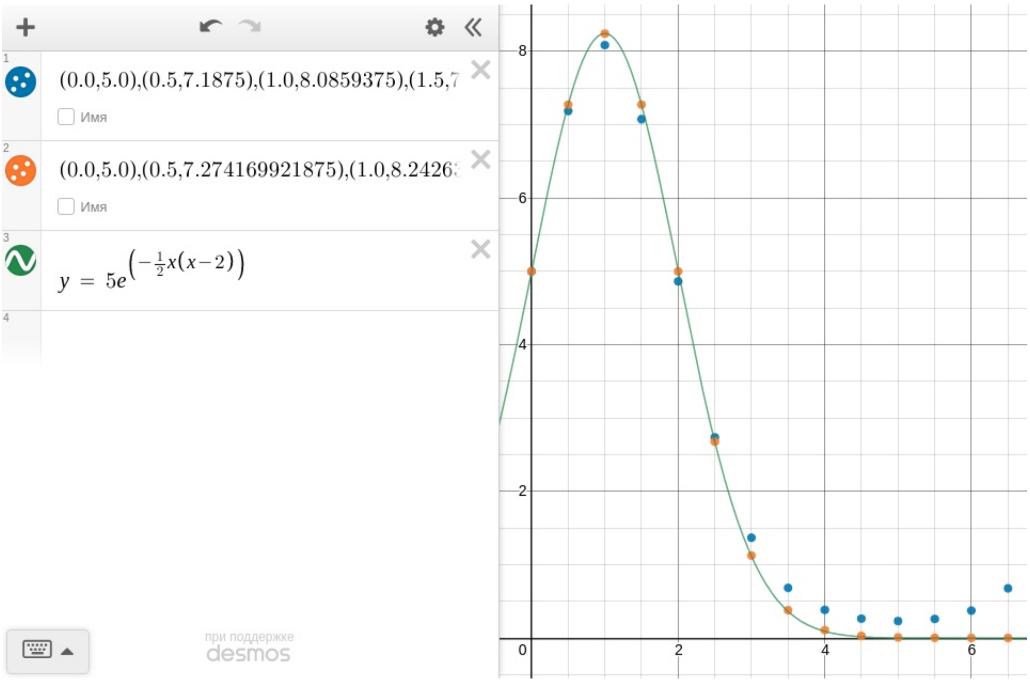

Рис. 5: $h = 0.5, n = 20$

Зеленая линия - аналитическое решение, синие точки - решение 2 порядка точности, оранжевые - 4 порядка точности.
На этом примере заметна серьезная разница в точности метода Рунге-Кутта 2 и 4 порядков точности

Таблица 2, вариант 2

Задача Коши: $\begin{cases} y_1' = xy_1 - y_2, \\ y_2' = y_1 - y_2 \end{cases} \quad y_1(0) = 0,\ y_2(0) = 1$

В результате работы программы были получены сеточные функции, отображенные на графиках (размер шага сетки - 0.25, количество шагов сетки - 20): Синие точки - решение 2 порядка точности, оранжевые - 4 порядка.

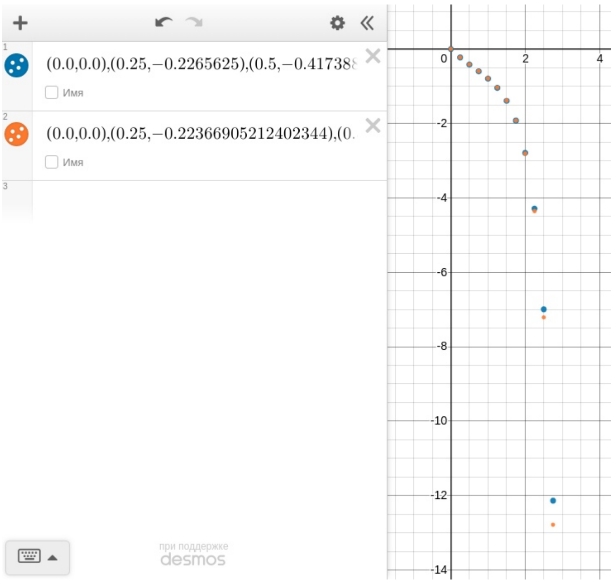

Рис. 6: $y = y_1(x)$

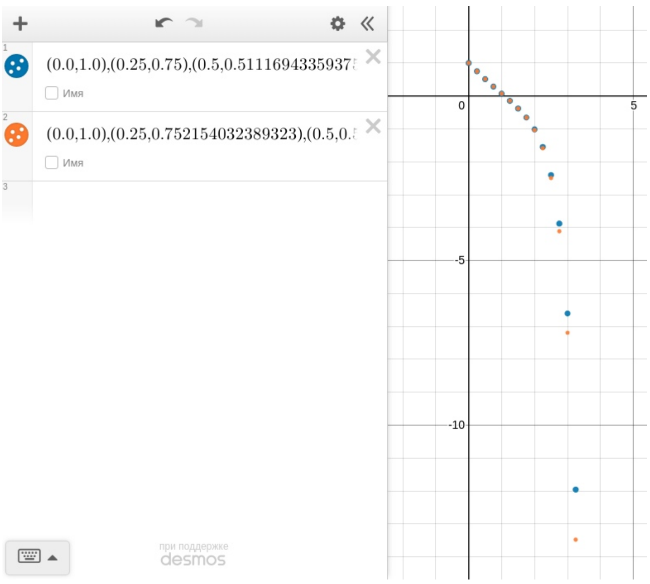

Рис. 7: $y = y_2(x)$

Выводы

В ходе работы был изучены и запрограммированы методы Рунге-Кутта второго и четвертого порядка точности, применяемые для численного решения задачи Коши для дифференциального уравнения первого порядка и систем дифференциальных уравнений первого порядка. На приведенных тестах показана ощутимая разница в точности решения методом вторым и четвертым порядком (см. тесты из таблицы 1).