Московский Государственный Университет
им. М. В. Ломоносова
Факультет Вычислительной математики и кибернетики

КОМПЬЮТЕРНЫЙ ПРАКТИКУМ ПО УЧЕБНОМУ КУРСУ
«ВВЕДЕНИЕ В ЧИСЛЕННЫЕ МЕТОДЫ»

ЗАДАНИЕ №2
«ЧИСЛЕННЫЕ МЕТОДЫ РЕШЕНИЯ ДИФФЕРЕНЦИАЛЬНЫХ УРАВНЕНИЙ»

ОТЧЕТ

о выполненном задании

студента 202 учебной группы факультета ВМК МГУ
Мартьянова Артема Олеговича

гор. Москва
2022 год

ПОДВАРИАНТ № 1

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

Цель работы

Освоить методы Рунге-Кутта второго и четвёртого 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 имеющее вид:

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

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

$$y(x_0) = y_0 \quad (2)$$

Предполагается, что правая часть уравнения (1) функция $f = f(x, y)$ 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):

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

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

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

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.

Цели и задачи практической работы

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.

Описание метода решения

Разложим решение дифференциального уравнения по формуле Тейлора и получим разностное уравнение:

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

Главная идея метода Рунге-Кутта заключается в том, чтобы приближенно заменить правую часть на сумму значений функции f в двух разных точках с точностью до членов порядка h^2. Таким образом, получим однопараметрическое семейство разностных схем Рунге-Кутта:

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

Наиболее удобной для вычисления разностной схемой этого семейства соответствует значение параметра $\alpha = \frac{1}{2}$. Тогда формула принимает рекуррентный вид:

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

В случае решения задачи Коши для системы дифференциальных уравнений, эта формула также справедлива для любого $y^{(i)}$ из вектора решений.

Иногда второго порядка точности недостаточно. Поэтому часто используется схема Рунге-Кутта четвертого порядка точности следующего вида:

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

где

$$k_1 = f(x_i,y_i)$$
$$k_2 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_1\right)$$
$$k_3 = f\left(x_i + \frac{h}{2}, y_i + \frac{h}{2}k_2\right)$$
$$k_4 = f(x_i + h, y_i + hk_3)$$

Тесты

Таблица 1-5

$$y' = (y - y^2) * x, \quad y(0) = 3$$

Аналитическое решение: $y = \frac{1}{1-\frac{2}{3}e^{-\frac{1}{2}x^2}}$

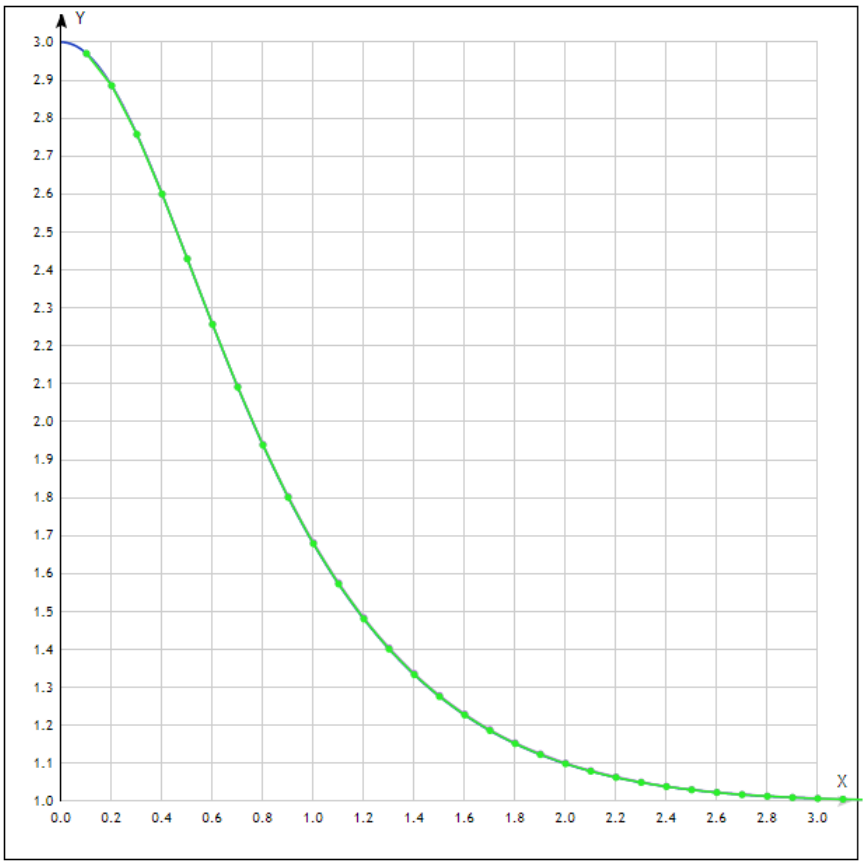
Метод Рунге-Кутта II и IV порядков

h = 0.25		h = 0.1		Аналитическое решение	
II	IV	II	IV	y	x
2.2417	2.2707	2.2577	2.2566	2.2566	0.6
1.6978	1.6790	1.6821	1.6789	1.6788	1.0
1.2498	1.2292	1.2303	1.2275	1.2275	1.6
1.1137	1.0994	1.1010	1.0992	1.0992	2.0
1.0306	1.0235	1.0240	1.0232	1.0232	2.6
1.0114	1.0075	1.0079	1.0075	1.0075	3.0

По результатам программы построен график, где видно аналитическое решение и решение методом Рунге-Кутта двух и четырех порядков при разном шаге h.

(синий – аналитическое решение; фиолетовый – II порядка; зеленый – IV порядка)

h = 0.1


Y
X
3.0
2.9
2.8
2.7
2.6
2.5
2.4
2.3
2.2
2.1
2.0
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8
3.0

h = 0.25

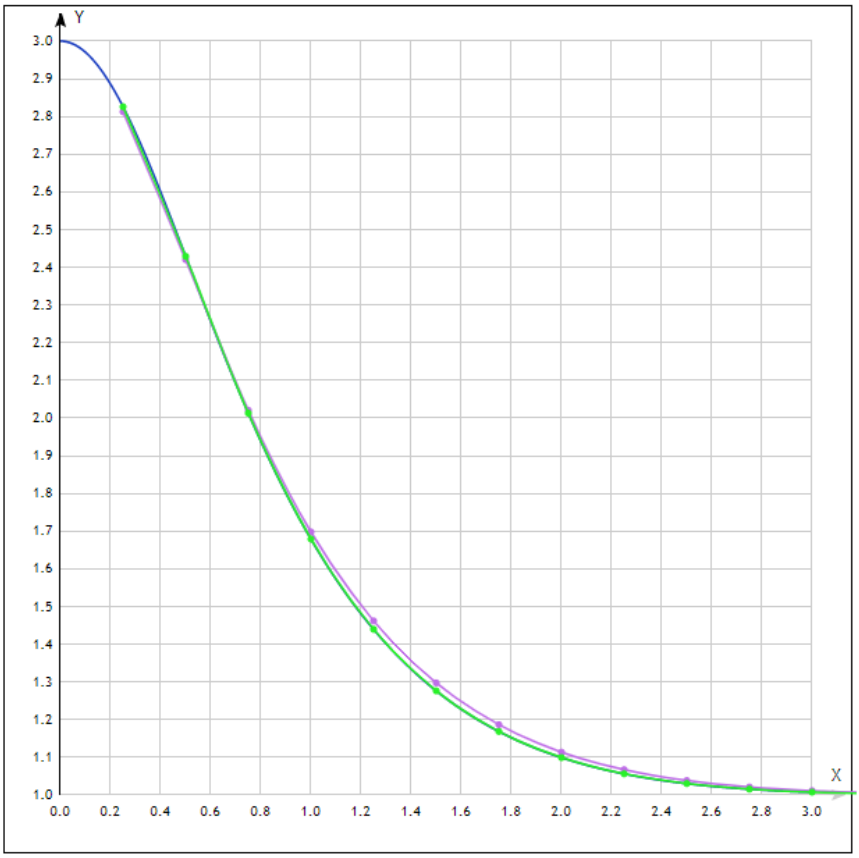

Y
X
3.0
2.9
2.8
2.7
2.6
2.5
2.4
2.3
2.2
2.1
2.0
1.9
1.8
1.7
1.6
1.5
1.4
1.3
1.2
1.1
1.0
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8
3.0

Таблица 2-4

$$\begin{cases} y_1' = \sqrt{x^2 + 1.1 \cdot {y_1}^2} + y_2 \\ y_2' = \cos(2.1 * y_2) + y_1 \end{cases}$$

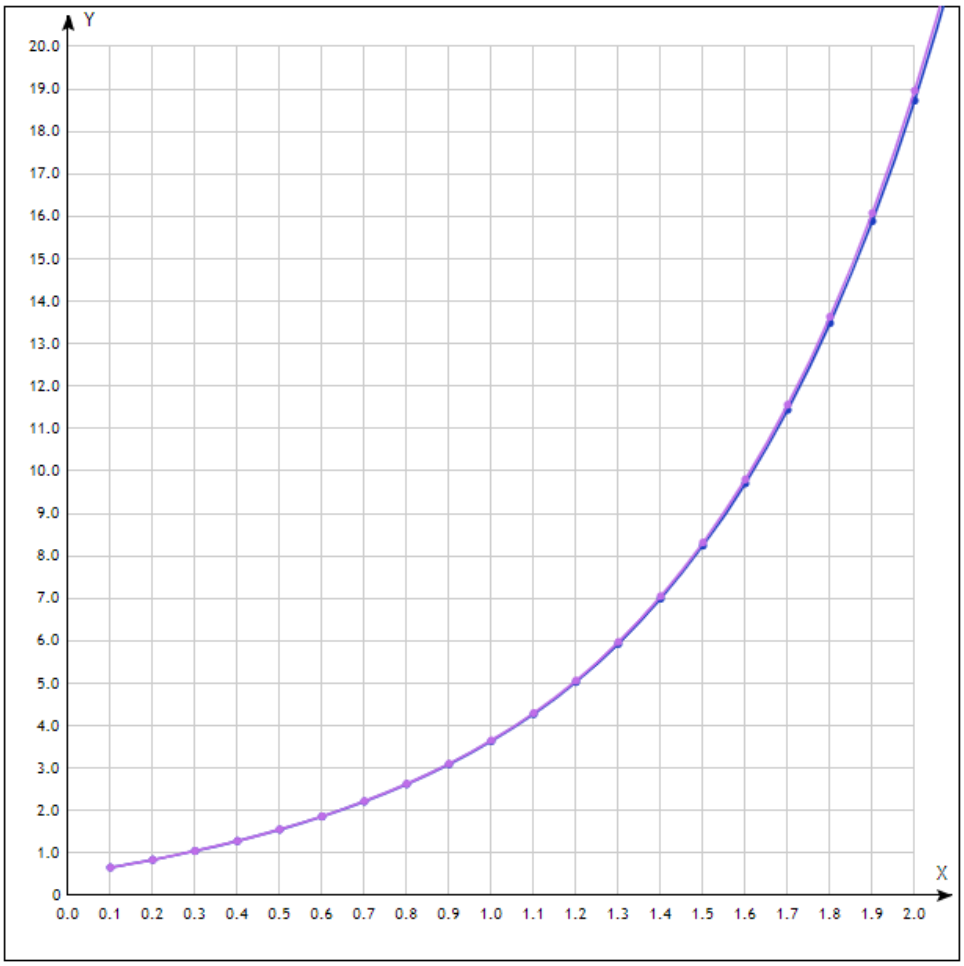
$$x_0 = 0, \;\; y_1(x_0) = 0.5, \;\; y_2(x_0) = 1.0$$

Метод Рунге-Кутта II и IV порядков для h = 0.1

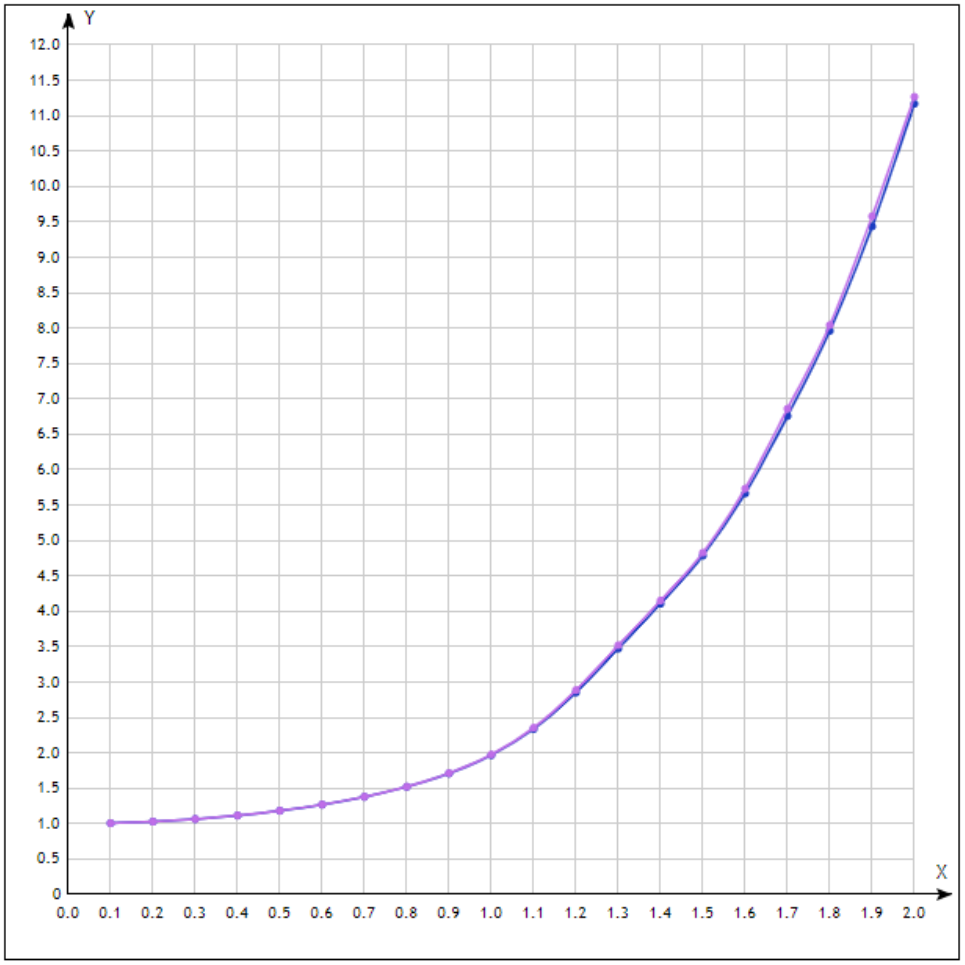
$y_1(x)$		$y_2(x)$	
II	IV	II	IV
(0.1; 0.660774)	(0.1; 0.661218)	(0.1; 1.00718)	(0.1; 1.00701)
(0.2; 0.841954)	(0.2; 0.842998)	(0.2; 1.02874)	(0.2; 1.02858)
(0.3; 1.04786)	(0.3; 1.04966)	(0.3; 1.06456)	(0.3; 1.06454)
(0.4; 1.28288)	(0.4; 1.28564)	(0.4; 1.11525)	(0.4; 1.11547)
(0.5; 1.55179)	(0.5; 1.55575)	(0.5; 1.18229)	(0.5; 1.18287)
(0.6; 1.86002)	(0.6; 1.86549)	(0.6; 1.26846)	(0.6; 1.26952)
(0.7; 2.21388)	(0.7; 2.22128)	(0.7; 1.37847)	(0.7; 1.38022)
(0.8; 2.62101)	(0.8; 2.63092)	(0.8; 1.52039)	(0.8; 1.52322)
(0.9; 3.09094)	(0.9; 3.10421)	(0.9; 1.70827)	(0.9; 1.71306)
(1; 3.63613)	(1; 3.6541)	(1; 1.96674)	(1; 1.97576)
(1.1; 4.27375)	(1.1; 4.29857)	(1.1; 2.3357)	(1.1; 2.35397)
(1.2; 5.02781)	(1.2; 5.06236)	(1.2; 2.85263)	(1.2; 2.88626)
(1.3; 5.92718)	(1.3; 5.97309)	(1.3; 3.47309)	(1.3; 3.52071)
(1.4; 6.99337)	(1.4; 7.05155)	(1.4; 4.10578)	(1.4; 4.15162)
(1.5; 8.24276)	(1.5; 8.31653)	(1.5; 4.78486)	(1.5; 4.82627)
(1.6; 9.70678)	(1.6; 9.80248)	(1.6; 5.6651)	(1.6; 5.73241)
(1.7; 11.4399)	(1.7; 11.5615)	(1.7; 6.7567)	(1.7; 6.8612)
(1.8; 13.483)	(1.8; 13.635)	(1.8; 7.96033)	(1.8; 8.04045)
(1.9; 15.8856)	(1.9; 16.079)	(1.9; 9.4347)	(1.9; 9.57948)
(2; 18.7254	(2; 18.963)	(2; 11.1684)	(2; 11.2672)

По результатам программы построен график, где видно аналитическое решение и решение методом Рунге-Кутта двух и четырех порядков.

(синий – II порядка; фиолетовый – IV порядка)


Y
X
20.0
19.0
18.0
17.0
16.0
15.0
14.0
13.0
12.0
11.0
10.0
9.0
8.0
7.0
6.0
5.0
4.0
3.0
2.0
1.0
0
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0

$y_1(x)$


Y
X
12.0
11.5
11.0
10.5
10.0
9.5
9.0
8.5
8.0
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0
0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9 2.0

$y_2(x)$

Вывод

В ходе работы был реализован метод Рунге-Кутта II и IV порядка точности для решения задачи Коши дифференциального уравнения первого порядка и системы дифференциальных уравнений первого порядка.

По результатам численных просчетов конечно-разностных уравнений метода Рунге-Кутта, полученным представленной программой, построены графики. В результате метод Рунге-Кутта IV порядка точности дает меньшее отклонение от точного решения уравнения, чем метод II порядка точности.

Таким образом, можно сделать вывод, что метод Рунге-Кутта IV порядка является более точным для решения дифференциальных уравнений первого порядка. Также при меньшем шаге методы сходятся лучше.

ПОДВАРИАНТ № 2

«РЕШЕНИЕ КРАЕВОЙ ЗАДАЧИ ДЛЯ ОБЫКНОВЕННОГО ДИФФЕРЕНЦИАЛЬНОГО УРАВНЕНИЯ ВТОРОГО ПОРЯДКА, РАЗРЕШЕННОГО ОТНОСИТЕЛЬНО СТАРШЕЙ ПРОИЗВОДНОЙ»

Цель работы

Освоить метод прогонки решения краевой задачи для дифференциального уравнения второго порядка.

Постановка задачи

Рассматривается линейное дифференциальное уравнение второго порядка вида:

$$y'' + p(x) * y' + q(x) * y = -f(x), \;\; 1 < x < 0, \qquad (1)$$

с дополнительными условиями в граничных точках

$$\begin{cases} \sigma_1 y(0) + \gamma_1 y'(0) = \delta_1 \\ \sigma_2 y(1) + \gamma_2 y'(1) = \delta_2 \end{cases} \qquad (2)$$

Цели и задачи практической работы

1) Реши Решить краевую задачу (1)-(2) методом конечных разностей, аппроксимировав ее разностной схемой второго порядка точности (на равномерной сетке); полученную систему конечно-разностных уравнений решить методом прогонки;
2) Найти разностное решение задачи и построить его график;
3) Найденное разностное решение сравнить с точным решением дифференциального уравнения.

Описание метода решения

Пусть дано ОДУ второго порядка вида (1) и дополнительные условия (2). Здесь, $y(x)$ - искомая функция, δ_1, δ_2 – краевые условия, $p(x), \; q(x), \; f(x)$. – заданные функции.

Введем на отрезке $[a; b]$ разностную сетку $(x_0, x_1, \dots, x_n)$ $x_i = a + ih, \; i = 0, 1, \dots, n$, $h = \frac{b-a}{n}$, n – параметр задачи (число шагов сетки).

Заменим уравнение разностной схемой:

$$\frac{y_{i+1} - 2 * y_i + y_{i-1}}{h^2} + p(x_i)\frac{y_{i+1} - y_{i-1}}{2 * h} + q(x_i) * y_i = f(x_i),$$

$$\sigma_1 y_0 + \gamma_1 \frac{y_1 - y_0}{h} = \delta_1,$$

$$\sigma_2 y_n + \gamma_2 \frac{y_n - y_{n-1}}{h} = \delta_2$$

Собирая коэффициенты при y_i, y_{i+1}, y_{i-1}, получим следующую СЛАУ на вектор неизвестных $(y_0, y_1, \dots, y_n)$:

$$\begin{cases} y_0\left(\sigma_1 - \frac{\gamma_1}{h}\right) + y_1\frac{\gamma_1}{h} = \delta_1, \\ A_i y_{i-1} - C_i y_i + B_i y_{i+1} = F_i, i = \overline{1, n-1}\,, \\ y_n\left(\sigma_2 + \frac{\gamma_2}{h}\right) - y_{n-1}\frac{\gamma_2}{h} = \delta_2\,, \end{cases} \quad (1)$$

где

$$C_i = 2 - q(x_i)h^2,\ A_i = 1 - p(x_i)\frac{h}{2},\ B_i = 1 + p(x_i)\frac{h}{2},\ F_i = h^2 f(x_i) \quad (2)$$

Система решается методом прогонки, в котором решение отыскивается в виде:

$$y_i = \xi_{i+1} y_{i+1} + \eta_{i+1}\,, i = \overline{0, n-1}$$

Алгоритм решения состоит из следующих шагов:

1. Поскольку $y_0 = \xi_1 y_1 + \eta_1$, и из первого уравнения $y_0 = \frac{A - \frac{\gamma_1 y_1}{h}}{\sigma_1 - \frac{\gamma_1}{h}}$, то

$$\xi_1 = -\frac{\gamma_1}{h\left(\sigma_1 - \frac{\gamma_1}{h}\right)}\,,\ \eta_1 = \frac{A}{\sigma_1 - \frac{\gamma_1}{h}}$$

2. Следующие прогоночные коэффициенты определяются по рекуррентным формулам (прямым ходом метода прогонки с трёхдиагональной матрицей):

$$\xi_{i+1} = \frac{B_i}{C_i - \xi_i A_i}\,, \qquad \eta_{i+1} = \frac{A_i\eta_i - F_i}{C_i - \xi_i A_i}\,, \qquad i = \overline{1, n-1};$$

3. Для определения решения на правой границе отрезка воспользуемся последним уравнением системы (1) и соотношением $y_{n-1} = \xi_n y_n + \eta_n$, где y_{n-1} определяется как решение системы двух линейных алгебраических уравнений.

4. Обратный ход метода прогонки для определения решения:

$$y_i = \xi_{i+1} y_{i+1} + \eta_{i+1}\,, i = \overline{n-1, 0}$$

Тесты

Вариант 15

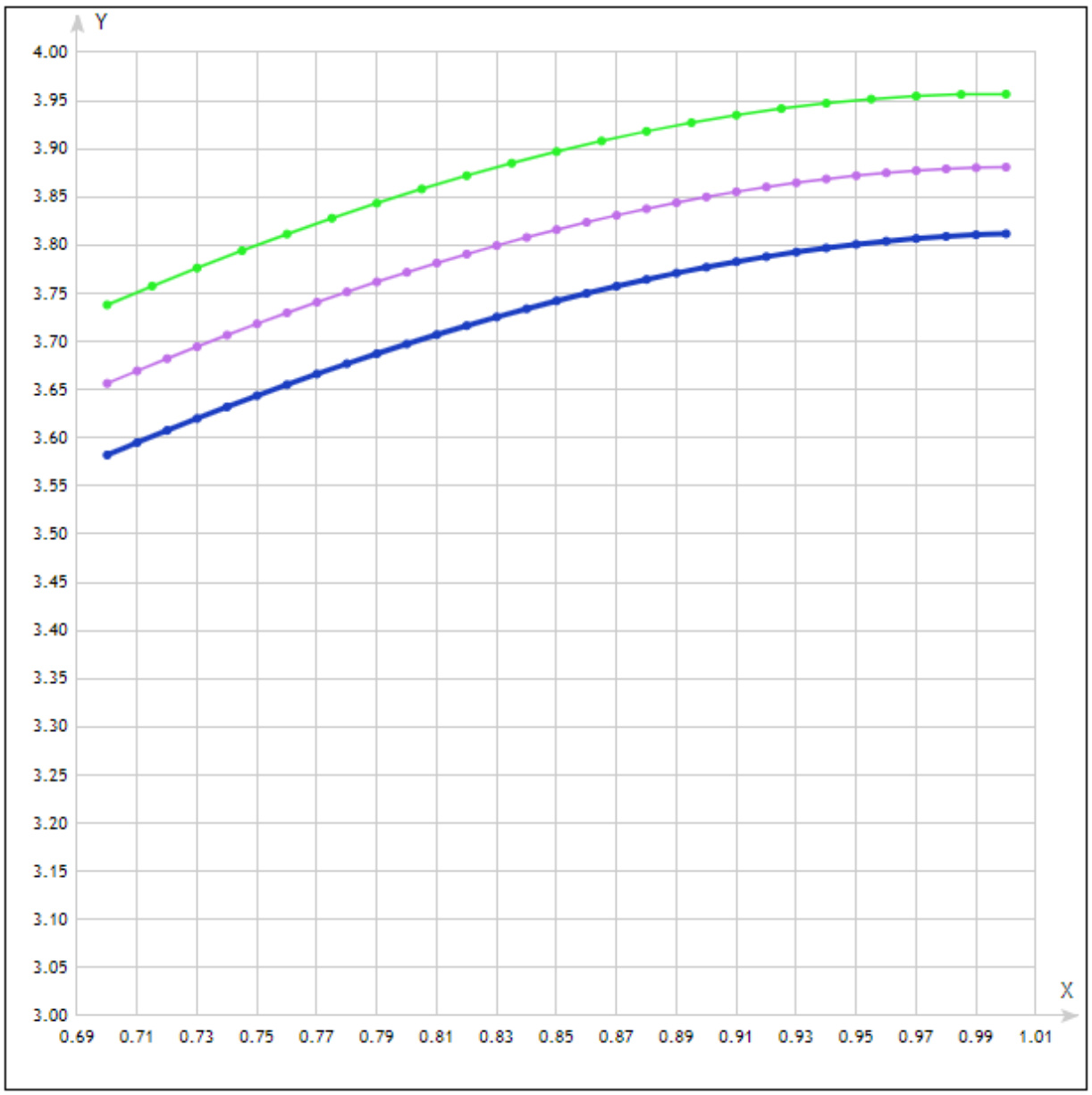
$$y'' - 3xy' + 2y = 1.5\,, \qquad y'(0.7) = 1.3, \qquad 0.5y(1) + y'(1) = 2$$

Результат работы программы:

0.015	0.01	0.005
(0.7; 3.737992)	(0.7; 3.6564974)	(0.7; 3.5821638)
(0.715; 3.757492)	(0.71; 3.6694974)	(0.71; 3.5950902)
(0.73; 3.7762542)	(0.72; 3.6821871)	(0.72; 3.6077198)
(0.745; 3.7942585)	(0.73; 3.694561)	(0.73; 3.6200473)
(0.76; 3.8114833)	(0.74; 3.7066135)	(0.74; 3.6320675)
(0.775; 3.8279059)	(0.75; 3.7183386)	(0.75; 3.6437746)
(0.79; 3.8435017)	(0.76; 3.7297301)	(0.76; 3.6551628)
(0.805; 3.8582447)	(0.77; 3.7407815)	(0.77; 3.6662259)
(0.82; 3.8721071)	(0.78; 3.7514861)	(0.78; 3.6769575)
(0.835; 3.8850594)	(0.79; 3.7618366)	(0.79; 3.6873508)
(0.85; 3.8970698)	(0.8; 3.7718259)	(0.8; 3.6973989)
(0.865; 3.9081047)	(0.81; 3.7814461)	(0.81; 3.7070945)
(0.88; 3.9181284)	(0.82; 3.7906892)	(0.82; 3.7164299)
(0.895; 3.9271026)	(0.83; 3.7995468)	(0.83; 3.7253971)
(0.91; 3.9349866)	(0.84; 3.8080101)	(0.84; 3.7339878)
(0.925; 3.941737)	(0.85; 3.81607)	(0.85; 3.7421934)
(0.94; 3.9473075)	(0.86; 3.823717)	(0.86; 3.7500048)
(0.955; 3.9516488)	(0.87; 3.830941)	(0.87; 3.7574124)
(0.97; 3.9547085)	(0.88; 3.8377318)	(0.88; 3.7644066)
(0.985; 3.9564304)	(0.89; 3.8440784)	(0.89; 3.7709769)
(1; 3.9567547)	(0.9; 3.8499696)	(0.9; 3.7771127)
	(0.91; 3.8553936)	(0.91; 3.7828027)
	(0.92; 3.860338)	(0.92; 3.7880353)
	(0.93; 3.86479)	(0.93; 3.7927981)
	(0.94; 3.8687362)	(0.94; 3.7970786)
	(0.95; 3.8721626)	(0.95; 3.8008634)
	(0.96; 3.8750546)	(0.96; 3.8041387)
	(0.97; 3.8773969)	(0.97; 3.8068899)
	(0.98; 3.8791738)	(0.98; 3.809102)
	(0.99; 3.8803685)	(0.99; 3.8107591)
	(1; 3.8809636)	(1; 3.8118449)

По результатам работы программы построен график, где отмечено полученное решение с различным шагом.

(синий – h = 0.005; фиолетовый – шаг h = 0.01; зеленый – шаг h = 0.015)



Однако определение аналитического решения данного задания имеет свои вычислительные сложности, с чем не справился ни один интернет ресурс. Поэтому для определения точности вычислений представленной в отчете программы, протестируем задачи, у которых известно аналитическое решение.

Тест №1

$$y'' - y = -1\,, \qquad y(-1) = y(1) = 0$$

Аналитическое решение:

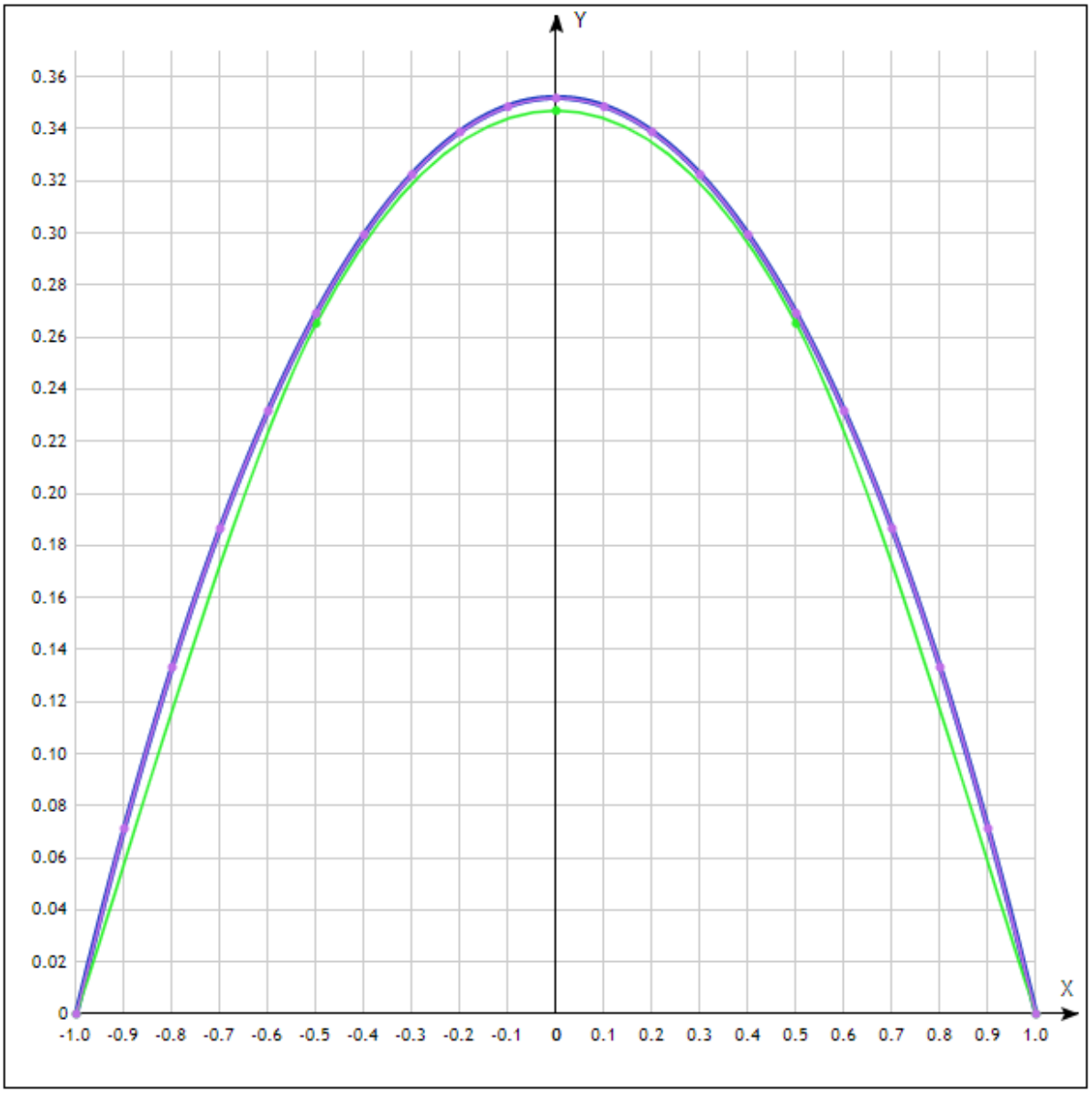
$$y(x) = 1 - \frac{chx}{ch1}$$

Результат работы программы:

h= 0.5	h = 0.1
(-1; 0) (-0.5; 0.26530612) (0; 0.34693878) (0.5; 0.26530612) (1; 0)	(-1; 0) (-0.9; 0.071237049) (-0.8; 0.13318647) (-0.7; 0.18646775) (-0.6; 0.23161371) (-0.5; 0.26907581) (-0.4; 0.29922867) (-0.3; 0.32237381) (-0.2; 0.33874269) (-0.1; 0.348499) (0; 0.3517403) (0.1; 0.348499) (0.2; 0.33874269) (0.3; 0.32237381) (0.4; 0.29922867) (0.5; 0.26907581) (0.6; 0.23161371) (0.7; 0.18646775) (0.8; 0.13318647) (0.9; 0.071237049) (1; 0)

По результатам работы программы и аналитическому решению построен график, где отмечено полученное решение с различным шагом. На нем видно, как с уменьшением шага найденное решение приближается к правильному.

(синий – аналитическое решение; фиолетовый – шаг h = 0.1; зеленый – шаг h = 0.5)



Вывод

В ходе практической работы был рассмотрен метод прогонки, применяемый для разностного решения краевой задачи для ОДУ второго порядка, разрешенного относительно старшей производной. При тестировании было выявлено, что при увеличении числа разбиений (уменьшении шага) точность решения задачи увеличивается.

Код программы на C++

```
1. #include <iostream>
2. #include <cmath>
3. #include <vector>
4.
5. using namespace std;
6.
7. double answer(double x) {
8.     return -0.5 * cos(x) + 0.5 * sin(x) + 21. / 2 * exp(-x);
9. }
10.
11.   double f(double x, double y) {
12.       return (y - y * y) * x;
13.   }
14.
15.   double f1(double x, double u, double v) {
16.       return sqrt(x * x + 1.1 * u * u) + v;
17.   }
18.
19.   double f2(double x, double u, double v) {
20.       return cos(2.1 * v) + u;
21.   }
22.
23.   void rk2(double x, double y, double h, double end) {
24.       while (x + h <= end) {
25.           y += h / 2 * (f(x, y) + f(x + h, y + h * f(x, y)));
26.           x += h;
27.           printf("(%.5lg; %.10lg)\n", x, y);
28.       }
29.   }
30.
31.   void rk4(double x, double y, double h, double n) {
32.       double k1, k2, k3, k4;
33.       while (x + h <= n) {
34.           k1 = h * f(x, y);
35.           k2 = h * f(x + h / 2, y + k1 / 2);
36.           k3 = h * f(x + h / 2, y + k2 / 2);
37.           k4 = h * f(x + h, y + k3);
38.           y += (k1 + 2 * (k2 + k3) + k4) / 6;
39.           x += h;
40.           printf("(%.5lg; %.10lg)\n", x, y);
41.       }
42.   }
43.
44.   void rk2Sys(double x, double u, double v, double h, double end) {
45.       double u_new, v_new;
46.       while (x + h <= end) {
47.           u_new = u + h / 2 * (f1(x, u, v) + f1(x + h, u + h * f1(x, u, v), v
   + h * f2(x, u, v)));
48.           v_new = v + h / 2 * (f2(x, u, v) + f2(x + h, u + h * f1(x, u, v), v
   + h * f2(x, u, v)));
49.           x += h;
50.           u = u_new;
51.           v = v_new;
52.           printf("(%.2lg; %.6lg)\n", x, u);
53.       }
54.   }
55.
56.   void rk4Sys(double x, double u, double v, double h, double n) {
57.       double ku1, ku2, ku3, ku4;
58.       double kv1, kv2, kv3, kv4;
59.       while (x + h <= n) {
60.           ku1 = h * f1(x, u, v);
```

```
61.             kv1 = h * f2(x, u, v);
62.             ku2 = h * f1(x + h / 2, u + ku1 / 2, v + kv1 / 2);
63.             kv2 = h * f2(x + h / 2, u + ku1 / 2, v + kv1 / 2);
64.             ku3 = h * f1(x + h / 2, u + ku2 / 2, v + kv2 / 2);
65.             kv3 = h * f2(x + h / 2, u + ku2 / 2, v + kv2 / 2);
66.             ku4 = h * f1(x + h, u + ku3, v + kv3);
67.             kv4 = h * f2(x + h, u + ku3, v + kv3);
68.             u += (ku1 + 2 * (ku2 + ku3) + ku4) / 6;
69.             v += (kv1 + 2 * (kv2 + kv3) + kv4) / 6;
70.             x += h;
71.             printf("(%.2lg; %.6lg)\n", x, u);
72.         }
73.     }
74.
75.     double p(double x) {
76.         return -3 * x;
77.     }
78.
79.     double q(double x) {
80.         return 2;
81.     }
82.
83.     double fr(double x) {
84.         return 1.5;
85.     }
86.
87.     void boundary(double h) {
88.         const double x0 = 0.7, x1 = 1.0;
89.         const double k1 = 0, k2 = 1.0;
90.         const double l1 = 0.5, l2 = 1.0;
91.         const double a = 1.3, b = 2;
92.         const double eps = 1e-12;
93.         const int n = (x1 - x0 + eps) / h;
94.         vector<double> aa(n + 1), bb(n + 1), cc(n + 1), ff(n + 1), x(n + 1);
95.         for (int i = 0; i <= n; ++i){
96.             x[i] = x0 + h * i;
97.             aa[i] = 1 - p(x[i]) * h / 2;
98.             bb[i] = 1 + p(x[i]) * h / 2;
99.             cc[i] = 2 - q(x[i]) * h * h;
100.            ff[i] = h * h * fr(x[i]);
101.        }
102.        vector<double> al(n + 1), bet(n + 1), y(n + 1); // al, bet - прогоночные
    коэффициенты, y - вектор ответ
103.        al[1] = k2 / (k2 - k1 * h);
104.        bet[1] = - (a * h) / (k2 - k1 * h);
105.        for (int i = 1; i < n; ++i){
106.            al[i + 1] = bb[i] / (cc[i] - al[i] * aa[i]);
107.            bet[i + 1] = (aa[i] * bet[i] - ff[i]) / (cc[i] - al[i] * aa[i]);
108.        }
109.        y[n] = (l2 * bet[n] + b * h) / (l2 + h * l1 - l2 * al[n]);
110.        for (int i = n - 1; i >= 0; --i){
111.            y[i] = al[i + 1] * y[i + 1] + bet[i + 1];
112.        }
113.        for (int i = 0; i <= n; ++i){
114.            printf("(%.5lg; %.8lg)\n", x[i], y[i]);
115.        }
116.    }
117.
118.    int main(int argc, const char * argv[]) {
119.        cout << "Выберите какую задачу требуется решить:\n1) Задача Коши\n2)
    Краевая задача\n";
120.        int problem;
121.        cin >> problem;
122.        if (problem == 1) {
```

```
123.          cout << "Выберите, что требуется решить:\n1) Дифференциальное
   уравнение\n2) Система дифференциальных уравнений\n";
124.          cin >> problem;
125.          cout << "Введите вторую границу отрезка: ";
126.          double end;
127.          cin >> end;
128.          double h;
129.          cout << "Введите требуемый шаг: ";
130.          cin >> h;
131.          if (problem == 1) {
132.              cout << "Какого порядка точности использовать метод (2/4)?" <<
   endl;
133.              cin >> problem;
134.              const double x = 0;
135.              const double y = 3;
136.              if (problem == 2) {
137.                  rk2(x, y, h, end);
138.              }
139.              else if(problem == 4) {
140.                  rk4(x, y, h, end);
141.              }
142.          }
143.          else if (problem == 2) {
144.              cout << "Какого порядка точности использовать метод (2/4)?" <<
   endl;
145.              cin >> problem;
146.              const double x = 0;
147.              const double u = 0.5;
148.              const double v = 1;
149.              if (problem == 2) {
150.                  rk2Sys(x, u, v, h, end);
151.              }
152.              else if(problem == 4) {
153.                  rk4Sys(x, u, v, h, end);
154.              }
155.          }
156.      }
157.      else if (problem == 2) {
158.          double h;
159.          cout << "Введите требуемый шаг: ";
160.          cin >> h;
161.          boundary(h);
162.      }
163.      return 0;
164.  }
```