МИНОБРНАУКИ РОССИИ

федеральное государственное бюджетное образовательное учреждение

высшего образования

«Нижегородский государственный технический университет

им. Р.Е. Алексеева» (НГТУ)

Кафедра: «Цифровая экономика»

Дисциплина: «Численные методы»

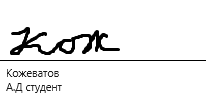
**Лабораторная работа №3**

**«Численные методы решения обыкновенных дифференциальных уравнений Вариант 7»**

Выполнил:

студент 3-го курса группы 21-САИ

Кожеватов Алексей Дмитриевич



Проверил:

д.ф.м.н., проф. Катаева Лилия Юрьевна

11.10.2023

Подпись преподавателя:\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Нижний Новгород, 2023

Оглавление

[Постановка задачи 3](#_Toc157043275)

[Задание №1 3](#_Toc157043276)

[Задание №2 3](#_Toc157043277)

[Задание №3 3](#_Toc157043278)

[Методы решений и инструменты 4](#_Toc157043279)

[Инструменты 4](#_Toc157043280)

[Методы решений 4](#_Toc157043281)

[Метод Рунге-Кутта 4](#_Toc157043282)

[Правило Рунге 5](#_Toc157043283)

[Метод Адамса 5](#_Toc157043284)

[Метод прогонки 6](#_Toc157043285)

[Проекционный метод 7](#_Toc157043286)

[Таблица идентификаторов 8](#_Toc157043287)

[Метод Рунге-Кутта 8](#_Toc157043288)

[Метод прогонки 8](#_Toc157043289)

[Проекционный метод 9](#_Toc157043290)

[Ручной счёт 11](#_Toc157043291)

[Метод Рунге-Кутта 11](#_Toc157043292)

[Метод прогонки 13](#_Toc157043293)

[Проекционный метод 16](#_Toc157043294)

[Реализация задачи в excel 20](#_Toc157043295)

[Метод Рунге-Кутта 20](#_Toc157043296)

[Метод прогонки 21](#_Toc157043297)

[Проекционный метод 22](#_Toc157043298)

[Реализация задачи в Mathcad 15 23](#_Toc157043299)

[Метод Рунге-Кутта 23](#_Toc157043300)

[Метод прогонки 25](#_Toc157043301)

[Реализация задачи в C++ 26](#_Toc157043302)

[Основа и выбор пользователя 26](#_Toc157043303)

[Метод Рунге-Кутта 27](#_Toc157043304)

[Метод прогонки 28](#_Toc157043305)

[Проекционный метод 30](#_Toc157043306)

[Реализация задачи в Java 32](#_Toc157043307)

[Основа и выбор пользователя 32](#_Toc157043308)

[Метод Рунге-Кутта 33](#_Toc157043309)

[Метод прогонки 35](#_Toc157043310)

[Проекционный метод 36](#_Toc157043311)

[Результат и анализ всех реализаций 39](#_Toc157043312)

[Список литературы 40](#_Toc157043313)

# Постановка задачи

Вариант 10

## Задание №1

1. Найти решение задачи Коши на отрезке методом Рунге-Кутта с шагом Решение в точках 0.4 и 0.5 найти методом Адамса. Оценить погрешность полученного решения, применяя правило Рунге.

;

## Задание №2

2. Решить краевую задачу

Методом прогонки по равномерной сетке

Для аппроксимации первой производной воспользоваться формулой

## Задание №3

3. Решить краевую задачу

Одним из проекционных методов. Данные взять из задачи 2.

# Методы решений и инструменты

## Инструменты

При решении лабораторной работы были использованы следующие инструменты:

1. Mathcad 15 (версия: M050 [], разрядность: x64);
2. Eclipse IDE (версия: 2022-09 (4.25.0), разрядность: x64);
3. Visual Studio IDE (версия: 17.7.3, разрядность: x64).
4. Microsoft Excel (версия: 2023, разрядность: x64).

Mathcad 15 — это программное обеспечение для математического моделирования и анализа технических и научных данных. Оно предоставляет удобную среду для создания и решения математических выражений и уравнений, а также выполнения численных и символьных вычислений.

Eclipse IDE — это интегрированная среда разработки (IDE), используемая в компьютерном программировании. Оно содержит базовое рабочее пространство и расширяемую систему подключаемых модулей для настройки среды. Это вторая по популярности среда IDE для разработки на Java, и до 2016 года она была самой популярной. Eclipse написан в основном на Java и его основное применение заключается в разработке приложений Java, но он также может быть использован для разработки приложений на других языках программирования с помощью плагинов.

Visual Studio IDE— это интегрированная среда разработки (IDE) от компании Microsoft, предназначенная для разработки программного обеспечения. В Visual Studio предоставляются различные инструменты и функциональные возможности, упрощающие процесс разработки, отладки и тестирования приложений.

Visual Studio поддерживает множество языков программирования, включая C++, C#, Visual Basic, F#, JavaScript и другие. Для разработки на C++ в Visual Studio используется компилятор Microsoft C++, который обеспечивает мощные возможности компиляции и оптимизации кода.

Visual Studio обеспечивает разработчиков на C++ всеми необходимыми инструментами для создания высококачественного программного обеспечения. Отличительной чертой Visual Studio является его обширная функциональность и поддержка различных платформ и технологий, что делает его одним из популярных выборов для разработки на C++.

## Методы решений

### Метод Рунге-Кутта

Пусть известно решение дифференциального уравнения в точке . Решение в точке будем искать в виде:

Параметры формулы Рунге-Кутта выбирают таким образом, чтобы погрешность приближённого решения, т.е. разность между точным значением и приближённым , была как можно меньшей. Обозначим погрешность на шаге через

Число шагов N, которое необходимо сделать до произвольной точки x, обратно пропорционально шагу h, т.е.

Наиболее распространена в практике следующая формула Рунге-Кутта, имеющая погрешность четвёртого порядка:

### Правило Рунге

Обозначим приближённое решение в точке , найденное с шагом h через , а через - приближённое решение, найденное с шагом 2h.

Справедливы приближённые равенства

Приравнивая правые части (1.5), найдём

Отсюда

### Метод Адамса

Особенности метода Адамса:

1. метод применяется для равномерной сетки т.е.

,

1. для нахождения приближённого решения в точке требуется знание приближённого решения в предыдущих точках, т.е. в

Точки образуют начальный отрезок. Решение на начальном отрезке может быть найдено каким-нибудь другим методом, например, методом Рунге-Кутта.

Пусть известно решение в точках

Проинтегрируем исходное уравнение на отрезке

или

Построим для подынтегральной функции интерполяционный многочлен Если интерполяционный многочлен построен по значениям функции в узлах то получим экстраполяционную формулу Адамса. Если интерполяционный многочлен построен по значениям в узлах , то получим интерполяционную формулу Адамса.

Так как значение интерполяционного полинома вычисляем на отрезке то в качестве интерполяционной формулы выберем формулу Ньютона для интерполирования назад.

Ниже записаны формулы Адамса, имеющие погрешность четвёртого порядка.

Экстраполяционная формула Адамса:

**

Интерполяционная формула Адамса:

, где

Следует заметить, что интерполяционная формула Адамса представляет собой уравнение относительно . Один из методов решения этого уравнения – метод последовательных приближений

, где .

Практическая оценка погрешности приближённого решения может быть получена с помощью правила Рунге.

### Метод прогонки

Пусть дано дифференциальное уравнение

требуется найти его решение, удовлетворяющее дифференциальному уравнению внутри отрезка и краевым условиям

Пусть задана краевая задача в виде

Построим равномерную сетку на отрезке

, , .

Заменив вторую производную в точке разностным отношением

а первую производную

Получим систему разностных уравнений

где – приближённые значения решения в точке .

В методе прогонки решение системы ищем в виде

где – прогоночные коэффициенты.

Сравнивая и краевое условие , получим

Если в

подставить вместо правую часть выражения и полученное уравнение записать, то получим рекуррентные формулы для :

Вычисления по методу прогонки проводят в два этапа. Первый этап называется прямой прогонкой и состоит из вычисления прогоночных коэффициентов . Второй этап называется обратной прогонкой и заключается в вычислении для значений

### Проекционный метод

В проекционных методах приближённое аналитическое решение ищут в виде:

где функции , удовлетворяют следующим условиям :

Эти ограничения необходимы для того, чтобы функция удовлетворяла граничным условиям

Система функций , должна быть линейно-независимой и полной в классе дважды непрерывно дифференцируемых функций. Этим условиям удовлетворяют, например, система степенных функций

Если приближённое решение подставить в правую часть дифференциального уравнения, то в общем случае

Разность называется невязкой.

Метод Коллокаций

Выберем на систему точек

и потребуем, чтобы

Эти соотношения представляют собой систему n линейных алгебраических уравнений для определения неизвестных параметров

# Таблица идентификаторов

## Метод Рунге-Кутта

Таблица 1

Таблица идентификаторов для метода Рунге-Кутта

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название переменной | Excel | Mathcad | C++ | Java | Комментарий |
| i | i | i | i | i | Итерации |
| j |  | j | j | j | Переменные для прохода итераций |
| n | n | n | n | n | n из дано |
| h | h | h | h | h | Шаг |
| y | y | y | y[i] | y[i] | Точки функции |
| x | x | x | x[i] | x[i] | Точки функции |
| f | f | f(x) | f[x] | f[x] | Данная функция f от x |
| k1 | k1 | k1 | k1 | k1 | Формулы Рунге-Кутта |
| k2 | k2 | k2 | k2 | k2 | Формулы Рунге-Кутта |
| k3 | k3 | k3 | k3 | k3 | Формулы Рунге-Кутта |
| k4 | k4 | k4 | k4 | k4 | Формулы Рунге-Кутта |

## Метод прогонки

Таблица 2

Таблица идентификаторов для метода прогонки

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название переменной | Excel | Mathcad | C++ | Java | Комментарий |
| i | i | i | i | i | Итерации |
| j |  | j | j | j | Переменные для прохода итераций |
| a | a | a | a | a | Первый x из дано |
| b | b | b | b | b | Последний x из дано |
| n | n | n | n | n | n из дано |
| h | h | h | h | h | Шаг |
| y | y | y | y[i] | y[i] | Точки функции |
| x | x | x | x[i] | x[i] | Точки функции |
| A | A | A(x) | A[x] | A[x] | Данная функция а |
| B | B | B(x) | B[x] | B[x] | Данная функция b |
| C | C | C(x) | C[x] | C[x] | Данная функция c |
| f | f | f(x) | f[x] | f[x] | Данная функция f |
| ai | ai | ai(x) | ai[x] | ai[x] | Изменённая функция a |
| bi | bi | bi(x) | bi[x] | bi[x] | Изменённая функция b |
| ci | ci | ci(x) | ci[x] | ci[x] | Изменённая функция c |
| L | L | L | L[i] | L[i] | Коэффициент L |
| K | K | K | K[i] | K[i] | Коэффициент K |

## Проекционный метод

Таблица 3

Таблица идентификаторов для проекционного метода

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название переменной | Excel | Mathcad | C++ | Java | Комментарий |
| i | i | i | i | i | Итерации |
| j |  | j | j | j | Переменные для прохода итераций |
| a | a | a | a | a | Первый x из дано |
| b | b | b | b | b | Последний x из дано |
| n | n | n | n | n | n из дано |
| h | h | h | h | h | Шаг |
| y | y | y | y[i] | y[i] | Точки функции |
| x | x | x | x[i] | x[i] | Точки функции |
| A | A | A(x) | A[x] | A[x] | Данная функция а |
| B | B | B(x) | B[x] | B[x] | Данная функция b |
| C | C | C(x) | C[x] | C[x] | Данная функция c |
| f | f | f(x) | f[x] | f[x] | Данная функция f |
| c | c | c | c | c | Коэффициенты для функции u(x) |
| d | d | d | d | d | Коэффициенты для функции u(x) |
| u | u(x) | u(x) | u[x] | u[x] | Функция от x для функции L. Проекция |
| a1 | a1 | a1 | a1 | a1 | Коэффициенты для решения L(u) |
| b1 | b1 | b1 | b1 | b1 | Коэффициенты для решения L(u) |
| c1 | c1 | c1 | c1 | c1 | Коэффициенты для решения L(u) |
| L | L(u(x)) | L(u(x)) | lu[x] | lu[x] | Функция от u(x) |

# Ручной счёт

## Метод Рунге-Кутта

h=0.1

Шаг 0:

x0=0, y0=0, y0’=0

Шаг 1:

x1=0.1

Ищем наклоны точек в интервале [0;0.1]:

0,1

0,105127

0,1

0,1

Тогда приближённое значение на этом этапе будет равно

0,10,10,10,101709

Решение задачи Коши на этом этапе будет равно

1,010223

Шаг 2:

x2=0.2

Ищем наклоны точек в интервале [0.1;0.2]:

0,101022

0,101019

0,101019

0,102048

Тогда приближённое значение на этом этапе будет равно

0,2029

Решение задачи Коши на этом этапе будет равно

1,041415

Шаг 3:

x3=0.3

Ищем наклоны точек в интервале [0.2;0.3]:

0,104141

0,102078

0,102056

0,104155

Тогда приближённое значение на этом этапе будет равно

0,305661

Решение задачи Коши на этом этапе будет равно

1,096034

Метод Адамса

Экстраполяционная формула для x4=0.4

0,3056611,0960341,0414151,0102231

Решение задачи Коши на этом этапе будет равно

1,182494

Экстраполяционная формула для x5=0.5

0,543278

Решение задачи Коши на этом этапе будет равно

1,312113

Интерполяционная формула для x4=0.4

0,419287

1,182599

Решение задачи Коши на этом этапе будет равно

1,182599

Интерполяционная формула для x5=0.5

0,419287

1,3123311,1825990,54361

Решение задачи Коши на этом этапе будет равно

1,312331

Правило Рунге для оценки погрешности

0,010838

0,00838

0,006619

0,005088

0,003709

## Метод прогонки

Прямой ход

Шаг 0:

,

Вычисляем шаг:

0,0625,

Значения функций A(x), B(x), C(x) в точке :

2,9,

1,

-0,95,

Расчёт значений для нахождения прогоночных коэффициентов:

742,4

-1501,75,

758,4

Подсчёт прогоночных коэффициентов:

Решение f(x) в точке :

Шаг 1:

1,0625,

Значения функций A(x), B(x), C(x) в точке :

3,089453,

1,043351,

-1,07246,

Расчёт коэффициентов для нахождения прогоночных коэффициентов

790,9

-1599,57,

807,5936

Подсчёт прогоночных коэффициентов:

0,504883

0,641653

Решение f(x) в точке :

Все шаги представлены в таблице 4

Таблица 4

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i | Xi | A(xi) | B(xi) | C(xi) | A(xi) | B(xi) | C(xi) | L | K | f(x) |
| 0 | 1 | 2,9 | 1 | -0,95 | 742,4 | -1501,75 | 758,4 | 0 | 1,3 | 1,80411 |
| 1 | 1,0625 | 3,089453 | 1,043351 | -1,07246 | 790,9 | -1599,57 | 807,5936 | 0,504883 | 0,641653 | 1,810602 |
| 2 | 1,125 | 3,282813 | 1,085942 | -1,20234 | 840,4 | -1699,38 | 857,7751 | 0,672726 | 0,421493 | 1,814968 |
| 3 | 1,1875 | 3,480078 | 1,12783 | -1,33965 | 890,9 | -1801,18 | 908,9453 | 0,756286 | 0,31093 | 1,81732 |
| 4 | 1,25 | 3,68125 | 1,169061 | -1,48438 | 942,4 | -1904,99 | 961,105 | 0,806117 | 0,244244 | 1,817766 |
| 5 | 1,3125 | 3,886328 | 1,209677 | -1,63652 | 994,9 | -2010,79 | 1014,255 | 0,839069 | 0,199523 | 1,816404 |
| 6 | 1,375 | 4,095313 | 1,249718 | -1,79609 | 1048,4 | -2118,59 | 1068,395 | 0,862366 | 0,167376 | 1,81333 |
| 7 | 1,4375 | 4,308203 | 1,289215 | -1,96309 | 1102,9 | -2228,39 | 1123,527 | 0,87962 | 0,143104 | 1,808637 |
| 8 | 1,5 | 4,525 | 1,328201 | -2,1375 | 1158,4 | -2340,19 | 1179,651 |  |  | 1,802411 |

Обратный ход:

Шаг 0:

1,1

Шаг 1:

1,10,1431041,11068665

Шаг 2:

1,110686650,1673761,125194266

Шаг 3:

1,1251942660,1995231,14363878

Шаг 4:

0,8061171,143638780,2442441,166150384

Шаг 5:

1,1661503840,3109301,192874004

Шаг 6:

1,1928740040,4214931,22396987

Шаг 7:

0,5048831,2596142

Шаг 8:

## Проекционный метод

Прямой ход

Шаг 0:

,

Вычисляем шаг:

0,0625,

Решаем систему уравнений:

Создаём обратную матрицу:

Для нахождения c и d перемножаем матрицы:

Отсюда ,

Шаг 0:

Значения функций A(x), B(x), C(x) в точке :

2,9,

1,

-0,95,

Расчёт коэффициентов для нахождения прогоночных коэффициентов

742,4

-1501,75,

758,4

Подсчёт прогоночных коэффициентов:

Решение f(x) в точке :

Методом подбора находим коэффициенты a1=- 1,33, b1=5.72, c1=0

Делаем проекцию

3,845

1,201601181,

-1,6055

1,816816075

1,759308757

0,044801

Шаг 1:

0,06251,0625,

Значения функций A(x), B(x), C(x) в точке :

3,089453,

1,043350569,

,

Расчёт коэффициентов для нахождения прогоночных коэффициентов

790,9

-1599,57,

807,5936

Подсчёт прогоночных коэффициентов:

Решение f(x) в точке :

1,81743

Делаем проекцию

1,275

3,762813

1,185378728,

-1,54434375

1,81743223

Шаги со 2 по 8 представлены в таблице 5

Таблица 5

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| i | xi | f(xi) | A(xi) | B(xi) | C(xi) | fu(xi) | Lu(xi) | Lu(xi)- f(xi) |
| 0 | 1 | 1,80411 | 2,9 | 1 | -0,95 | 1,816816075 | 1,759308757 | 0,0448011 |
| 1 | 1,0625 | 1,810602 | 3,089453 | 1,043350569 | -1,072460938 | 1,81743223 | 1,775825698 | 0,0347767 |
| 2 | 1,125 | 1,814968 | 3,282813 | 1,085942339 | -1,20234375 | 1,817765504 | 1,790963902 | 0,024004 |
| 3 | 1,1875 | 1,81732 | 3,480078 | 1,127829715 | -1,339648438 | 1,817809719 | 1,80470909 | 0,0126112 |
| 4 | 1,25 | 1,817766 | 3,68125 | 1,16906056 | -1,484375 | 1,817558578 | 1,81704631 | 0,0007192 |
| 5 | 1,3125 | 1,816404 | 3,886328 | 1,209677265 | -1,636523438 | 1,817005648 | 1,827959891 | 0,0115561 |
| 6 | 1,375 | 1,81333 | 4,095313 | 1,249717607 | -1,79609375 | 1,816144349 | 1,837433393 | 0,024103 |
| 7 | 1,4375 | 1,808637 | 4,308203 | 1,289215438 | -1,963085938 | 1,814967938 | 1,845449552 | 0,0368128 |
| 8 | 1,5 | 1,802411 | 4,525 | 1,32820124 | -2,1375 | 1,813469497 | 1,851990217 | 0,0495792 |

# Реализация задачи в excel

## Метод Рунге-Кутта

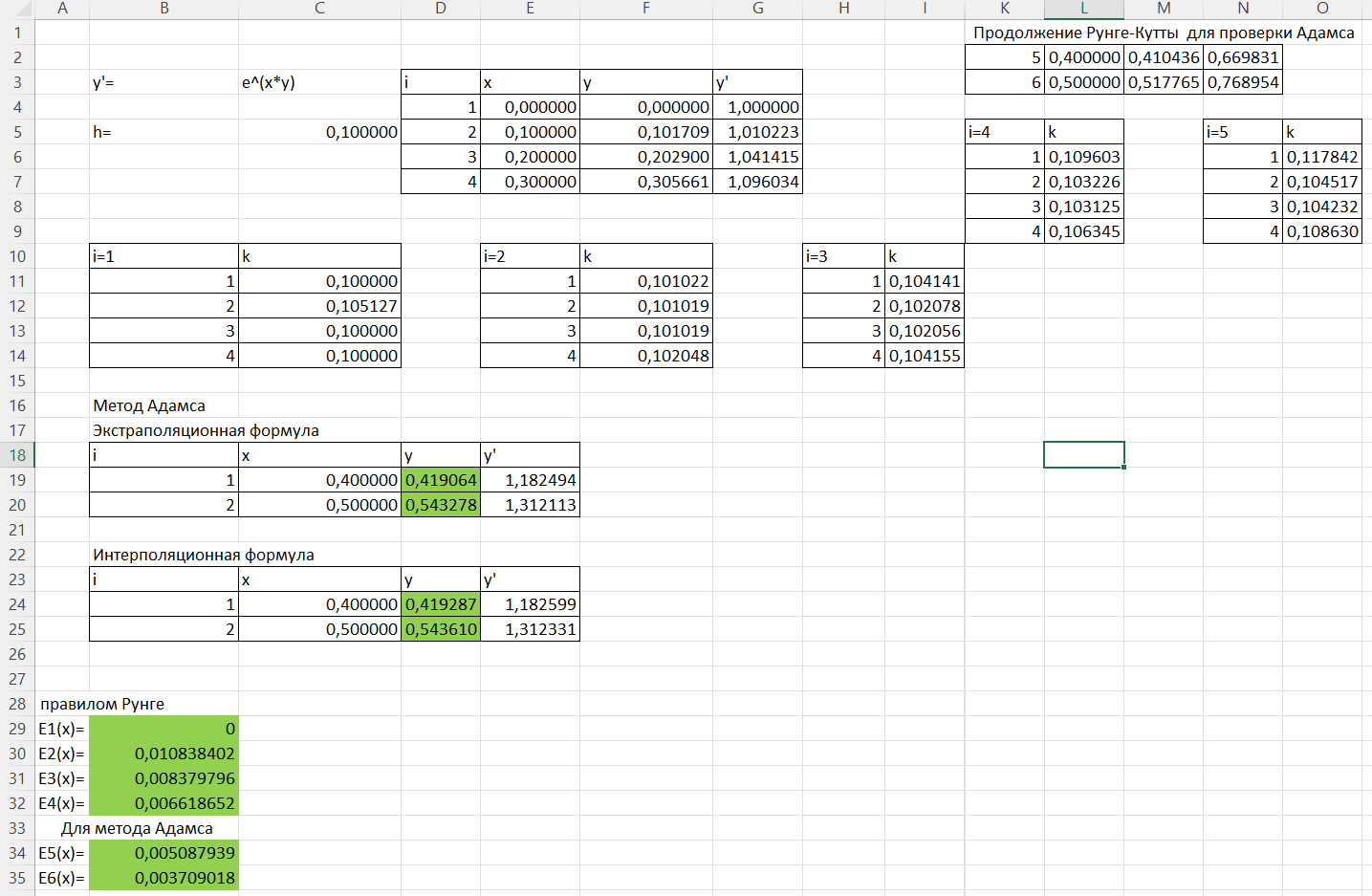


Рисунок 1. - Метод Рунге-Кутта

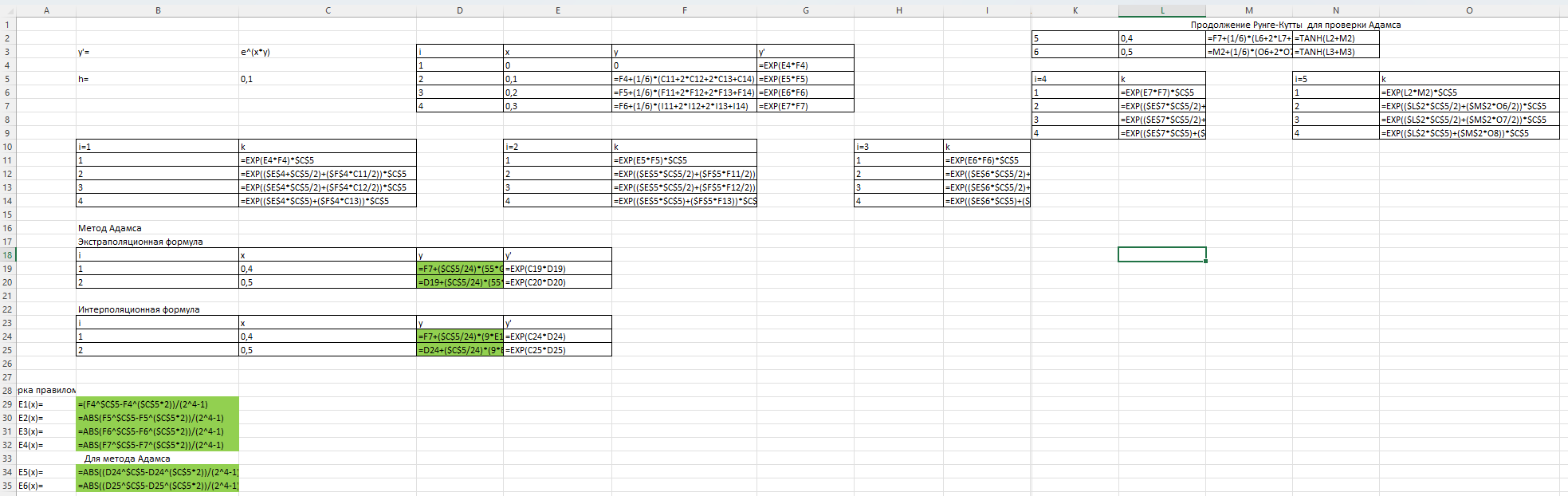


Рисунок 2. - Метод Рунге-Кутта

## Метод прогонки

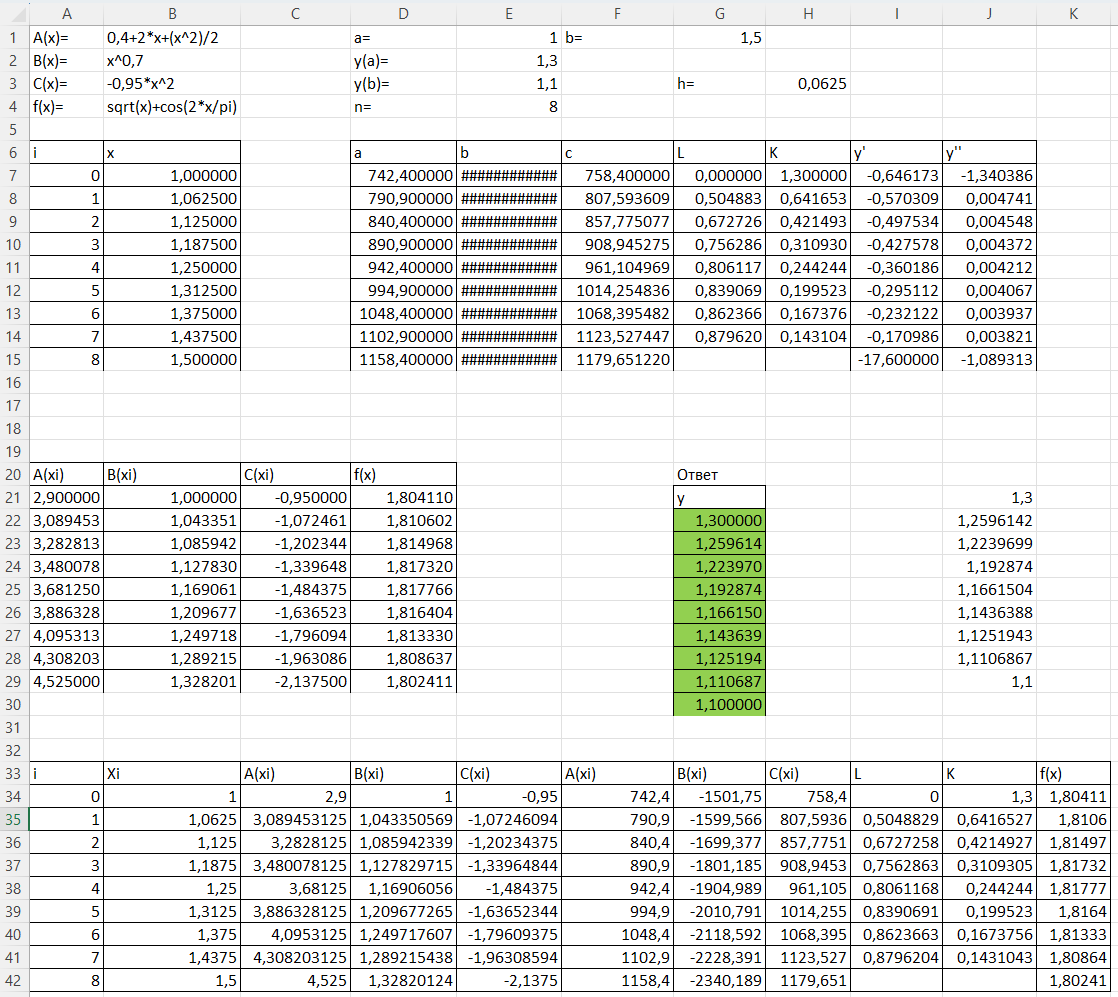


Рисунок 3. - Метод прогонки

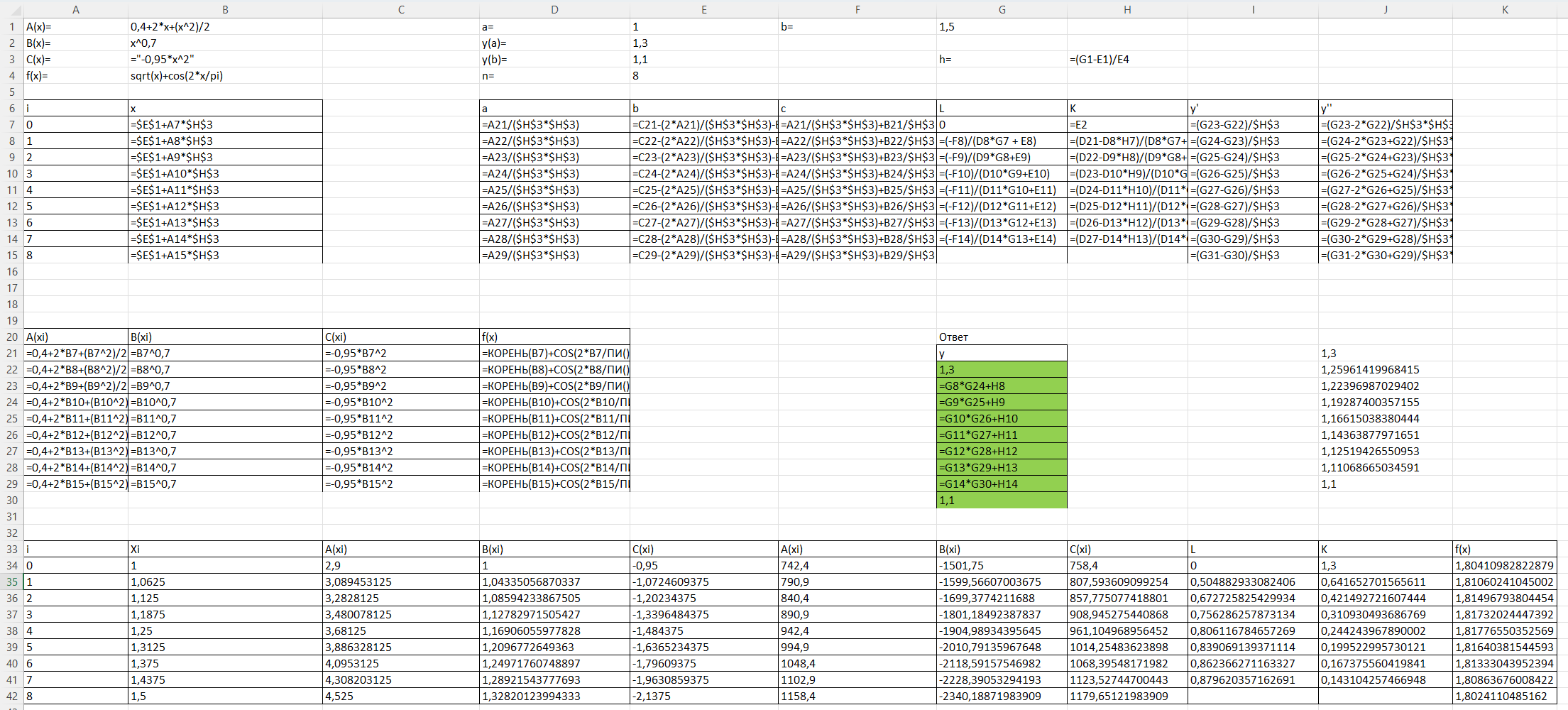


Рисунок 4. - Метод прогонки

## Проекционный метод

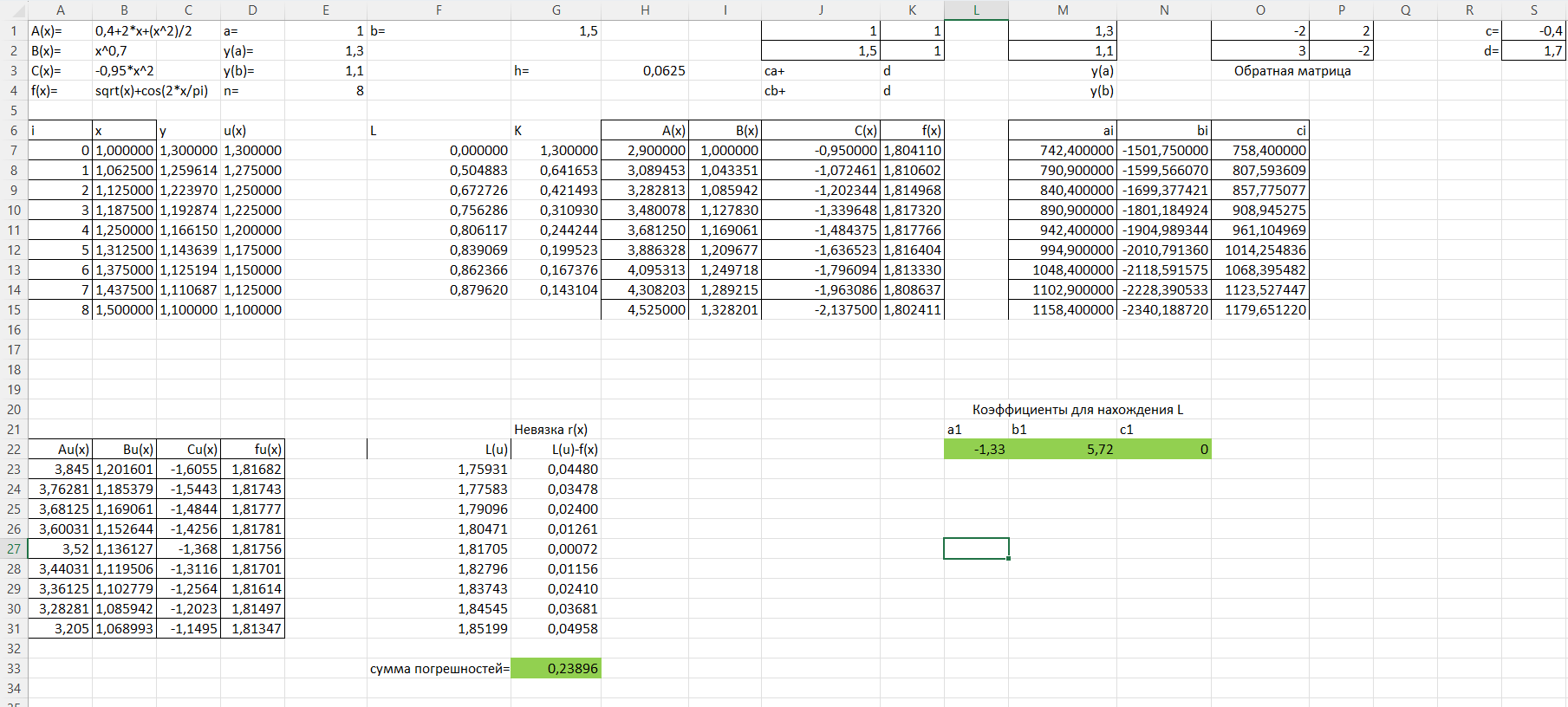


Рисунок 5. - Проекционный метод

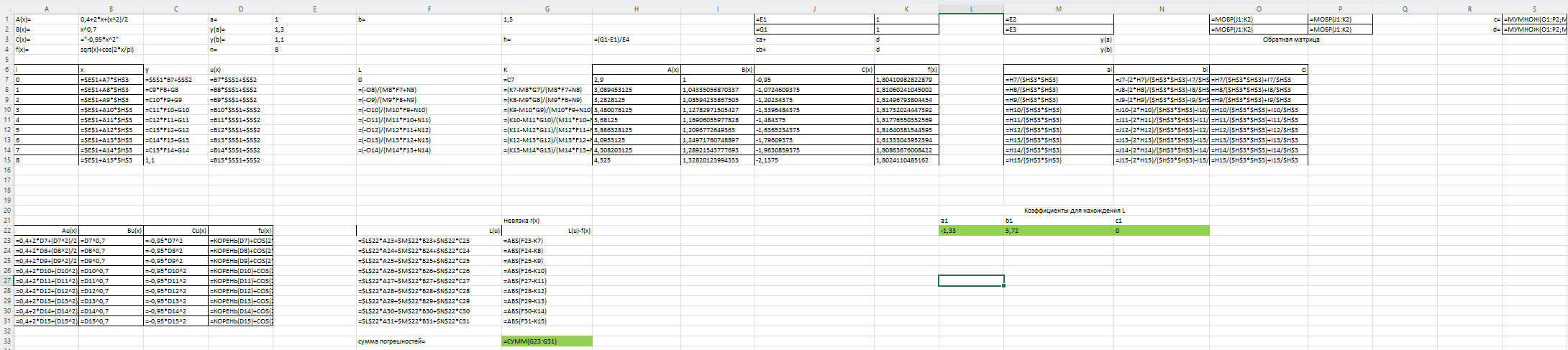


Рисунок 6. - Проекционный метод

# Реализация задачи в Mathcad 15

## Метод Рунге-Кутта

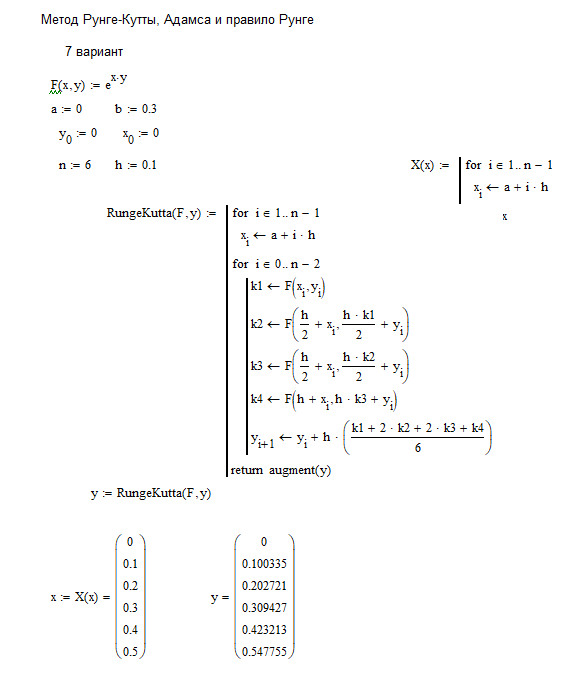


Рисунок 7. - Метод Рунге-Кутта

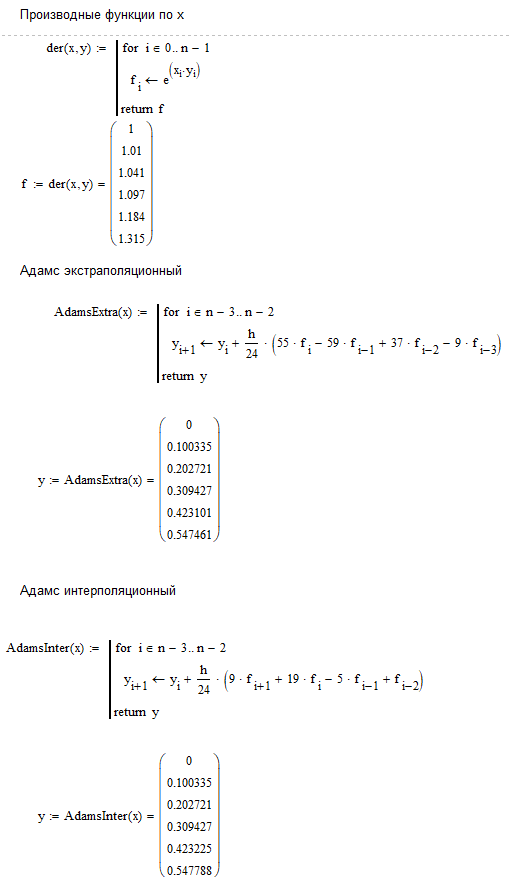


Рисунок 8. – Метод Адамса экстраполяционный и интерполяционный

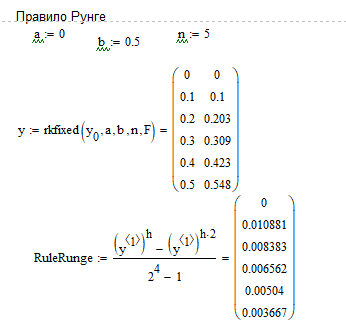


Рисунок 9. – Правило Рунге

## Метод прогонки

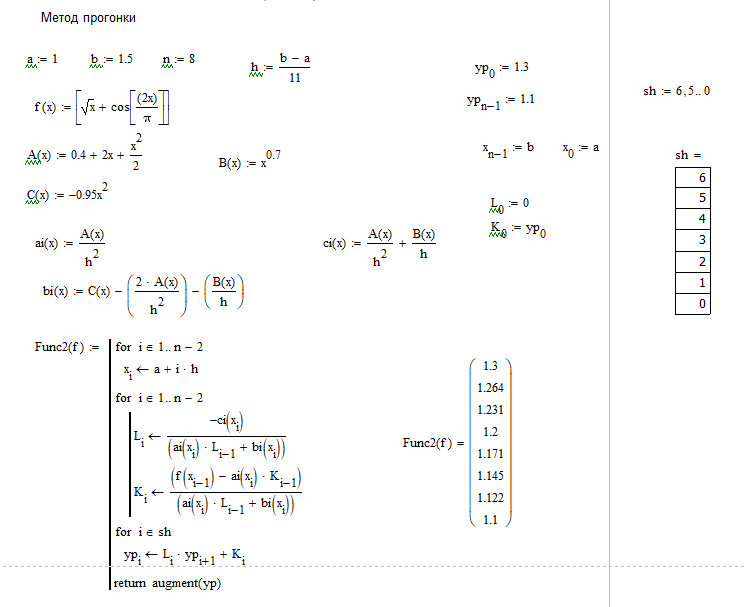


Рисунок 10. - Метод прогонки

# Реализация задачи в C++

## Основа и выбор пользователя

#include <iostream>

#include <math.h>

#include <cmath>

#include <complex>

#include "Runge.h"

#include "RunTh.h"

#include "Proj.h"

using namespace std;

int main() {

setlocale(LC\_ALL, "Russian");

Runge Runge;

RunTh RunTh;

Proj Proj;

int choice;

int exit = 0;

do {

cout << "Программа решает ОДУ. Выберите метод, которым хотите решить (введите номер метода):\n";

cout << "1.Метод Рунге-Кутта.\n";

cout << "2.Метод прогонки.\n";

cout << "3.Проекционный метод.\n";

cout << "4.Завершение работы.\n";

cout << "Введите число:";

cin >> choice;

// Правильность выбора метода

while (!(choice == 1 || choice == 2 || choice == 3 || choice == 4)) {

cout << "Выберите числа от 1 до 4!\n";

cin >> choice;

//Проверка на число

while (!cin)

{

cout << "Введите число!\n";

cin.clear();

while (cin.get() != '\n') continue;

cin >> choice;

}

}

switch (choice) {

case 1:

cout << "Вы выбрали метод Рунге-Кутта.\n";

Runge.main();

break;

case 2:

cout << "Вы выбрали метод прогонки.\n";

RunTh.main();

break;

case 3:

cout << "Вы выбрали проекционный метод.\n";

Proj.main();

break;

case 4:

cout << "Завершение работы.\n\n";

cout << "Обратная связь:\n";

cout << "Студент гр. 21-САИ, Краличев Игорь Евгеньевич, ikralichev@list.ru\n";

exit = 1;

break;

}

} while (exit == 0);

}

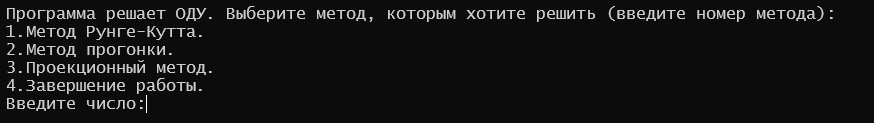


Рисунок 11.-Меню выбора пользователя

## Метод Рунге-Кутта

#include <iostream>

#include <cmath>

using namespace std;

class Runge {

public:

void main() {

int n = 6;

double h = 0.1;

double x[6];

x[0] = 0;

double y[6];

double f[6];

y[0] = 0; //y' = e^(x\*y)

double k1, k2, k3, k4;

cout << "y[" << 0 << "] = " << y[0] << ", x[" << 0 << "] = " << x[0] << endl;

for (int i = 1; i < n; i++) {

x[i] = 0 + i \* h;

}

cout << "Метод Рунге-Кутта:" << endl;

// метод Рунге-Кутта

for (int i = 0; i < n - 2; i++) {

k1 = exp(x[i] \* y[i]) \* h;

k2 = exp((x[i] \* h / 2) + ((k1) / 2 \* y[i])) \* h;

k3 = exp((x[i] \* h / 2) + ((k2) / 2 \* y[i])) \* h;

k4 = exp((x[i] \* h) + (y[i] \* k3)) \* h;

f[i] = exp(x[i] \* y[i]);

y[i + 1] = y[i] + ((k1 + 2 \* k2 + 2 \* k3 + k4) / 6);

}

for (int i = 0; i < n - 3; i++) {

cout << "y[" << (i + 1) << "] = " << y[i + 1] << ", x[" << (i + 1) << "] = " << x[i + 1] << endl;

}

cout << "Метод Аддамса экстраполяционный:" << endl;

// метод Аддамса экстраполяционный

for (int i = n - 3; i < n - 1; i++) {

f[i] = exp(x[i] \* y[i]);

y[i + 1] = y[i] + (h / 24) \* (55 \* f[i] - 59 \* f[i - 1] + 37 \* f[i - 2] - 9 \* f[i - 3]);

cout << "y[" << (i + 1) << "] = " << y[i + 1] << ", x[" << (i + 1) << "] = " << x[i + 1] << endl;

}

f[5] = exp(x[5] \* y[5]);

cout << "Метод Аддамса интерполяционный:" << endl;

// метод Аддамса интерполяционный

for (int i = n - 3; i < n - 1; i++) {

y[i + 1] = y[i] + (h / 24) \* (9 \* f[i + 1] + 19 \* f[i] - 5 \* f[i - 1] + f[i - 2]);

cout << "y[" << (i + 1) << "] = " << y[i + 1] << ", x[" << (i + 1) << "] = " << x[i + 1] << endl;

}

cout << "Правило Рунге:" << endl;

// правило Рунге

for (int i = 0; i < n; i++) {

cout << "Погрешность y[" << i << "] = " << abs((pow(y[i], h) - pow(y[i], h \* 2)) / (15)) << endl;

}

}

};

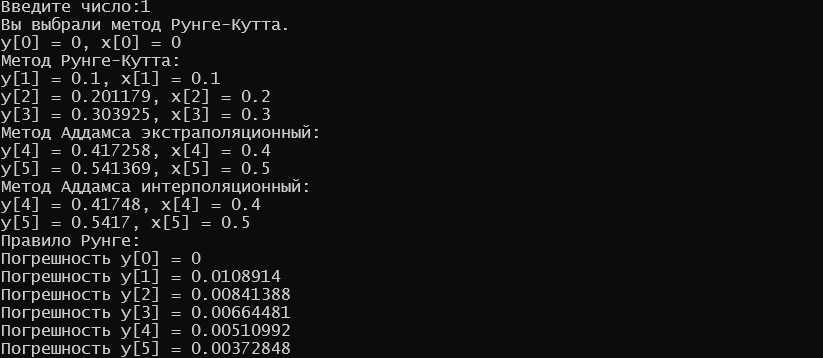


Рисунок 12. - Метод Рунге-Кутта и Адамса

## Метод прогонки

#include <iostream>

#include <cmath>

using namespace std;

class RunTh {

public:

double a = 1;

double b = 1.5;

static const int n = 9;

static const int hn = 8;

double h = (b - a) / hn;

double y[n];

double x[n];

double L[n];

double K[n];

double A(double x) {

return 0.4 + 2 \* x + pow(x, 2) / 2;

}

double B(double x) {

return pow(x, 0.7);

}

double C(double x) {

return -0.95 \* pow(x, 2);

}

double f(double x) {

return sqrt(x) + cos(2 \* x / 3.1415926); // приближенное значение pi = 3.1415926

}

double ai(double x) {

return (A(x)) / (h \* h);

}

double bi(double x) {

return C(x) - ((2 \* A(x)) / (h \* h)) - (B(x) / h);

}

double ci(double x) {

return (A(x) / (h \* h)) + (B(x) / h);

}

void main() {

y[0] = 1.3;

y[n - 1] = 1.1;

L[0] = 0;

K[0] = y[0];

x[0] = 1;

x[n - 1] = 1.5;

for (int i = 1; i < n - 1; i++) {

x[i] = a + i \* h;

}

// Прогоночные коэффициенты

for (int i = 1; i < n - 1; i++) {

L[i] = (-ci(x[i])) / (ai(x[i]) \* L[i - 1] + bi(x[i]));

K[i] = ((f(x[i - 1]) - ai(x[i]) \* K[i - 1]) / (ai(x[i]) \* L[i - 1] + bi(x[i])));

}

// Вычисление решения

for (int i = n - 2; i > -1; i--) {

y[i] = L[i] \* y[i + 1] + K[i];

}

cout << "Решение методом прогонки:" << endl;

for (int i = 0; i < n; i++) {

cout << "y[" << i << "] = " << y[i] << endl;

}

}

};

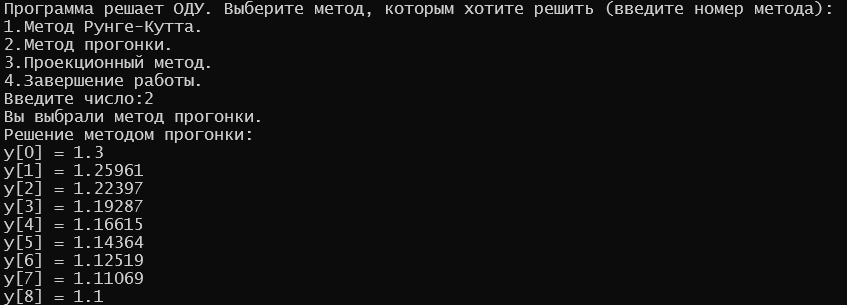


Рисунок 13. - Метод прогонки

## Проекционный метод

#include <iostream>

#include <cmath>

using namespace std;

class Proj {

public:

double A(double x) {

return 0.4 + 2 \* x + pow(x, 2) / 2;

}

double B(double x) {

return pow(x, 0.7);

}

double C(double x) {

return -0.95 \* pow(x, 2);

}

double f(double x) {

return sqrt(x) + cos(2 \* x / 3.1415926); // приближенное значение pi = 3.1415926

}

double L(double u, double a1, double b1, double c1) {

return a1 \* A(u) + b1 \* B(u) + c1 \* C(u);

}

void main() {

double a = 1;

double b = 1.5;

double c, d, a2, b2, c2, sum;

const int n = 8;

double h = (b - a) / n;

double y[12];

double x[12];

double u[12];

double matrix[2][2];

x[0] = a;

x[n] = b;

y[0] = 1.3;

y[n] = 1.1;

for (int i = 1; i < n; i++) {

x[i] = a + i \* h;

}

double det = a - b; // a \* 1 - b \* 1

matrix[0][0] = 1 / det;

matrix[0][1] = -1 / det;

matrix[1][0] = -b / det;

matrix[1][1] = a / det;

for (int i = 0; i < 2; i++) {

cout << matrix[i][0] << endl;

cout << matrix[i][1] << endl;

if (i == 0) c = matrix[i][0] \* y[0] + matrix[i][1] \* y[n];

d = matrix[i][0] \* y[0] + matrix[i][1] \* y[n];

}

cout << "c = " << c << ", d = " << d << endl;

for (int i = 0; i < n + 1; i++) {

u[i] = c \* x[i] + d;

cout << "u[" << i << "] = " << u[i] << endl;

}

double e = 0.24;

double starta = -5, enda = 0;

double startb = 5, endb = 7;

double startc = 0, endc = 2;

double step = 0.01;

bool gotit = false;

for (double a1 = starta; a1 <= enda && !gotit; a1 += step) {

for (double b1 = startb; b1 <= endb && !gotit; b1 += step) {

for (double c1 = startc; c1 <= endc && !gotit; c1 += step) {

sum = 0;

for (int i = 0; i < n + 1; i++) {

sum += abs(L(u[i], a1, b1, c1) - f(x[i]));

}

if (sum < e) {

cout << "Найдено решение для коэффициентов: a1 = " << a1 << ", b1 = " << b1 << ", c1 = " << c1 << endl;

for (int j = 0; j < 3; j++) {

a2 = a1;

b2 = b1;

c2 = c1;

}

gotit = true;

}

}

}

}

for (int i = 0; i < n + 1; i++) {

cout << "L(u[" << i << "]) = " << L(u[i], a2, b2, c2) << ", f(x[" << i << "]) = " << f(x[i]) << ", lu - fx = " << abs(L(u[i], a2, b2, c2) - f(x[i])) << endl;

}

cout << "Сумма погрешностей = " << sum << endl;

}

};

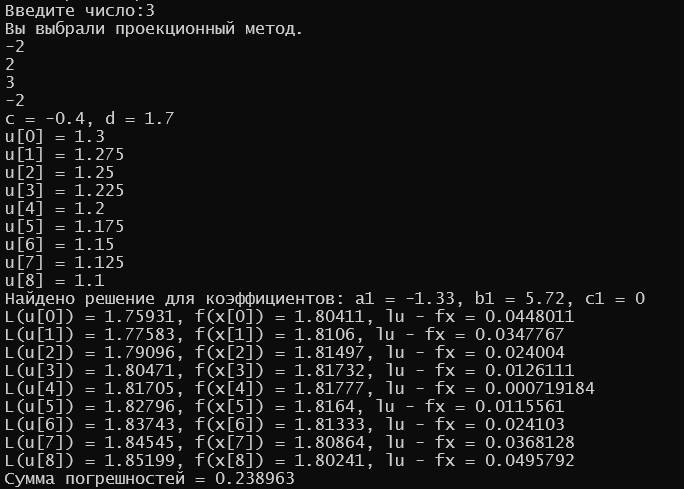


Рисунок 14. - Проекционный метод

# Реализация задачи в Java

## Основа и выбор пользователя

**package** Lab3;

**import** java.util.Scanner;

**public** **class** Menu {

**public** **static** **void** main(String[] args) {

Scanner scanner = **new** Scanner(System.***in***);

**int** choice;

**int** exit = 0;

**do** {

System.***out***.println("Программа решает ОДУ. Выберите метод, которым хотите решить (введите номер метода):");

System.***out***.println("1. Метод Рунге-Кутта.");

System.***out***.println("2. Метод прогонки.");

System.***out***.println("3. Проекционный метод.");

System.***out***.println("4. Завершение работы.");

System.***out***.println("Введите число: ");

choice = scanner.nextInt();

**while**(choice !=1 && choice !=2 && choice !=3 && choice !=4) {

System.***out***.println("Введите число от 1 до 4!");

**while** (**true**) {

**if** (scanner.hasNextInt()) {

choice = scanner.nextInt();

**break**;

} **else** {

System.***out***.println("Введите число!");

scanner.nextLine();

}

}

}

**switch** (choice) {

**case** 1:

System.***out***.println("Вы выбрали метод Рунге-Кутта.");

Runge.*main*(args);

**break**;

**case** 2:

System.***out***.println("Вы выбрали метод прогонки.");

//RunTh.main(args);

**break**;

**case** 3:

System.***out***.println("Вы выбрали проекционный метод.");

// Proj.main(args);

**break**;

**case** 4:

System.***out***.println("Завершение работы.\n");

exit = 1;

**break**;

}

} **while** (exit == 0);

// scanner.close();

}

}

}

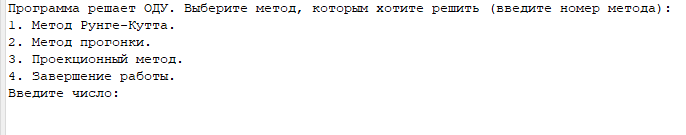


Рисунок 15.-Меню выбора пользователя

## Метод Рунге-Кутта

**package** Lab3;

**public** **class** Runge {

**public** **static** **void** main(String[] args) {

// Ваши значения для сравнения

**int** n = 6;

**double** h = 0.1;

**double**[] x = **new** **double**[n];

x[0] = 0;

**double**[] y = **new** **double**[n];

**double**[] f = **new** **double**[n];

y[0] = 0; // y' = e^(x\*y)

**double** k1, k2, k3, k4;

System.***out***.println("y[" + 0 + "] = " + y[0] + ", x[" + 0 + "] = " + x[0]);

**for** (**int** i = 1; i < n; i++) {

x[i] = 0 + i \* h;

}

System.***out***.println("Метод Рунге-Кутта:");

// метод Рунге-Кутта

**for** (**int** i = 0; i < n - 2; i++) {

k1 = Math.*exp*(x[i] \* y[i]) \* h;

k2 = Math.*exp*((x[i] \* h / 2) + ((k1) / 2 \* y[i])) \* h;

k3 = Math.*exp*((x[i] \* h / 2) + ((k2) / 2 \* y[i])) \* h;

k4 = Math.*exp*((x[i] \* h) + (y[i] \* k3)) \* h;

f[i] = Math.*exp*(x[i] \* y[i]);

y[i + 1] = y[i] + ((k1 + 2 \* k2 + 2 \* k3 + k4) / 6);

}

**for** (**int** i = 0; i < n - 3; i++) {

System.***out***.println("y[" + (i + 1) + "] = " + y[i + 1] + ", x[" + (i + 1) + "] = " + x[i + 1]);

}

System.***out***.println("Метод Аддамса экстраполяционный:");

// метод Аддамса экстраполяционный

**for** (**int** i = n - 3; i < n - 1; i++) {

f[i] = Math.*exp*(x[i] \* y[i]);

y[i + 1] = y[i] + (h / 24) \* (55 \* f[i] - 59 \* f[i - 1] + 37 \* f[i - 2] - 9 \* f[i - 3]);

System.***out***.println("y[" + (i + 1) + "] = " + y[i + 1] + ", x[" + (i + 1) + "] = " + x[i + 1]);

}

f[5] = Math.*exp*(x[5] \* y[5]);

System.***out***.println("Метод Аддамса интерполяционный:");

// метод Аддамса интерполяционный

**for** (**int** i = n - 3; i < n - 1; i++) {

y[i + 1] = y[i] + (h / 24) \* (9 \* f[i + 1] + 19 \* f[i] - 5 \* f[i - 1] + f[i - 2]);

System.***out***.println("y[" + (i + 1) + "] = " + y[i + 1] + ", x[" + (i + 1) + "] = " + x[i + 1]);

}

System.***out***.println("Правило Рунге:");

// правило Рунге

**for** (**int** i = 0; i < n; i++) {

System.***out***.println("Погрешность y[" + i + "] = "

+ Math.*abs*((Math.*pow*(y[i], h) - Math.*pow*(y[i], h \* 2)) / (15)));

}

}

}

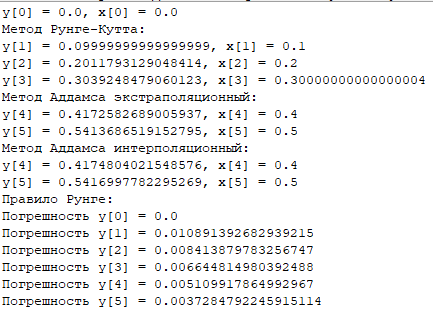
}

Рисунок 16. - Метод Рунге-Кутта и Адамса

## Метод прогонки

**package** Lab3;

**public** **class** RunTh {

**static** **double** *a* = 1;

**static** **double** *b* = 1.5;

**static** **int** *n* = 9;

**static** **double** *h* = (*b* - *a*) / (*n* - 1);

**static** **double**[] *y* = **new** **double**[*n*];

**static** **double**[] *x* = **new** **double**[*n*];

**static** **double**[] *L* = **new** **double**[*n*];

**static** **double**[] *K* = **new** **double**[*n*];

**public** **static** **double** A(**double** x) {

**return** 0.4 + 2 \* x + Math.*pow*(x, 2) / 2;

}

**public** **static** **double** B(**double** x) {

**return** Math.*pow*(x, 0.7);

}

**public** **static** **double** C(**double** x) {

**return** -0.95 \* Math.*pow*(x, 2);

}

**public** **static** **double** f(**double** x) {

**return** Math.*sqrt*(x) + Math.*cos*(2 \* x / Math.***PI***); // приближенное значение pi = 3.1415926

}

**public** **static** **double** ai(**double** x) {

**return** *A*(x) / (*h* \* *h*);

}

**public** **static** **double** bi(**double** x) {

**return** *C*(x) - (2 \* *A*(x) / (*h* \* *h*)) - (*B*(x) / *h*);

}

**public** **static** **double** ci(**double** x) {

**return** *A*(x) / (*h* \* *h*) + *B*(x) / *h*;

}

**public** **static** **void** main(String[] args) {

*y*[0] = 1.3;

*y*[*n* - 1] = 1.1;

*L*[0] = 0;

*K*[0] = *y*[0];

*x*[0] = 1;

*x*[*n* - 1] = 1.5;

**for** (**int** i = 1; i < *n* - 1; i++) {

*x*[i] = *a* + i \* *h*;

}

**for** (**int** i = 1; i < *n* - 1; i++) {

*L*[i] = -*ci*(*x*[i]) / (*ai*(*x*[i]) \* *L*[i - 1] + *bi*(*x*[i]));

*K*[i] = (*f*(*x*[i - 1]) - *ai*(*x*[i]) \* *K*[i - 1]) / (*ai*(*x*[i]) \* *L*[i - 1] + *bi*(*x*[i]));

}

**for** (**int** i = *n* - 2; i > -1; i--) {

*y*[i] = *L*[i] \* *y*[i + 1] + *K*[i];

}

System.***out***.println("Решение методом прогонки:");

**for** (**int** i = 0; i < *n*; i++) {

System.***out***.println("y[" + i + "] = " + *y*[i]);

}

}

}

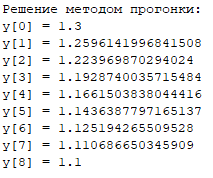


Рисунок 17. - Метод прогонки

## Проекционный метод

**package** Lab3;

**public** **class** Proj {

**public** **static** **double** A(**double** x) {

**return** 0.4 + 2 \* x + Math.*pow*(x, 2) / 2;

}

**public** **static** **double** B(**double** x) {

**return** Math.*pow*(x, 0.7);

}

**public** **static** **double** C(**double** x) {

**return** -0.95 \* Math.*pow*(x, 2);

}

**public** **static** **double** f(**double** x) {

**return** Math.*sqrt*(x) + Math.*cos*(2 \* x / Math.***PI***); // приближенное значение pi = 3.1415926

}

**public** **static** **double** L(**double** u, **double** a1, **double** b1, **double** c1) {

**return** a1 \* *A*(u) + b1 \* *B*(u) + c1 \* *C*(u);

}

**static** **double** *a* = 1;

**static** **double** *b* = 1.5;

**static** **int** *n* = 8;

**static** **double** *c*, *d*, *a2*, *b2*, *c2*, *sum*;

**static** **double** *h* = (*b* - *a*) / *n*;

**static** **double**[] *y* = **new** **double**[12];

**static** **double**[] *x* = **new** **double**[12];

**static** **double**[] *u* = **new** **double**[12];

**static** **double**[][] *matrix* = **new** **double**[2][2];

**public** **static** **void** main(String[] args) {

*x*[0] = *a*;

*x*[*n*] = *b*;

*y*[0] = 1.3;

*y*[*n*] = 1.1;

**for** (**int** i = 1; i < *n*; i++) {

*x*[i] = *a* + i \* *h*;

}

**double** det = *a* - *b*; // a \* 1 - b \* 1

*matrix*[0][0] = 1 / det;

*matrix*[0][1] = -1 / det;

*matrix*[1][0] = -*b* / det;

*matrix*[1][1] = *a* / det;

**for** (**int** i = 0; i < 2; i++) {

System.***out***.println(*matrix*[i][0]);

System.***out***.println(*matrix*[i][1]);

**if** (i == 0) *c* = *matrix*[i][0] \* *y*[0] + *matrix*[i][1] \* *y*[*n*];

*d* = *matrix*[i][0] \* *y*[0] + *matrix*[i][1] \* *y*[*n*];

}

System.***out***.println("c = " + *c* + ", d = " + *d*);

**for** (**int** i = 0; i < *n* + 1; i++) {

*u*[i] = *c* \* *x*[i] + *d*;

System.***out***.println("u[" + i + "] = " + *u*[i]);

}

**double** e = 0.24;

**double** starta = -5, enda = 0; //-14 -14 -12

**double** startb = 5, endb = 7; //-10.5 -11 -8

**double** startc = 0, endc = 2; //-9 -9 -8

**double** step = 0.01;

**boolean** gotit = **false**;

**for** (**double** a1 = starta; a1 <= enda && !gotit; a1 += step) {

**for** (**double** b1 = startb; b1 <= endb && !gotit; b1 += step) {

**for** (**double** c1 = startc; c1 <= endc && !gotit; c1 += step) {

*sum* = 0;

**for** (**int** i = 0; i < *n* + 1; i++) {

*sum* += Math.*abs*(*L*(*u*[i], a1, b1, c1) - *f*(*x*[i]));

}

**if** (*sum* < e) {

System.***out***.println("Найдено решение для коэффициентов: a1 = " + a1 + ", b1 = " + b1 + ", c1 = " + c1);

**for** (**int** j = 0; j < 3; j++) {

*a2* = a1;

*b2* = b1;

*c2* = c1;

}

gotit = **true**;

}

}

}

}

**for** (**int** i = 0; i < *n* + 1; i++) {

System.***out***.println("L(u[" + i + "]) = " + *L*(*u*[i], *a2*, *b2*, *c2*) + ", f(x[" + i + "]) = " + *f*(*x*[i]) + ", lu - fx = " + Math.*abs*(*L*(*u*[i], *a2*, *b2*, *c2*) - *f*(*x*[i])));

}

System.***out***.println("Сумма погрешностей = " + *sum*);

}

}

}

}

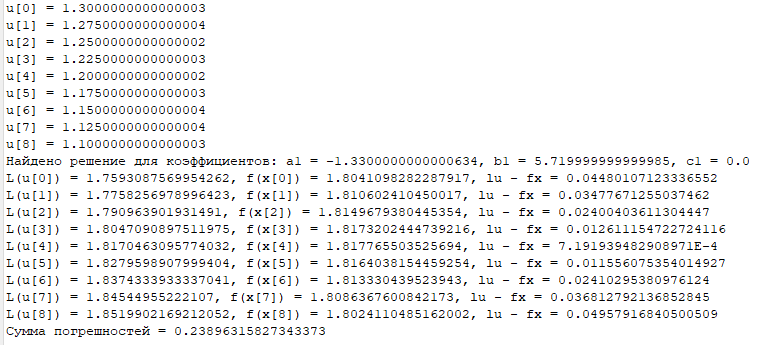


Рисунок 18. - Проекционный метод

# Результат и анализ всех реализаций

Таблица 6

Таблица результатов

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Метод | Ручной счёт | Excel | Mathcad | C++ | Java |
| Рунге-Кутта, Адамса |  |  |  |  |  |
| Прогонки |  |  |  |  |  |
| Проекционный |  |  |  |  |  |

Всеми инструментами можно достичь высокой точности.

# Список литературы

**1.** **Численное решение задач экономики с использованием EXCEL, C++ и MATLAB [Электронные текстовые данные] : Учеб.пособие / Л.Ю. Катаева [и др.]; НГТУ им.Р.Е.Алексеева. - Н.Новгород : [Изд-во НГТУ], 2020. - 230 с. : ил. - Прил.:c.188-230. - Библиогр.:с.187.:**

[**https://fdp.nntu.ru/books/Chisl\_reshenie\_zadach\_economiki/Chisl\_reshenie\_zadach\_economiki/assets/basic-html/index.html#189в**](https://fdp.nntu.ru/books/Chisl_reshenie_zadach_economiki/Chisl_reshenie_zadach_economiki/assets/basic-html/index.html#189%D0%B2)**.**

**2. Численные методы : Курс лекций / В.А. Срочко. - СПб.; М.; Краснодар : Лань, 2010. - 202 с. - (Учебники для вузов. Специальная литература). - Библиогр.:с.200. - ISBN 978-5-8114-1014-9 : 180-00.**

[**https://studfile.net/preview/5793014/page:4/**](https://studfile.net/preview/5793014/page:4/)

**3. Численные методы линейной алгебры : Учеб.пособие / Г.С. Шевцов, О.Г. Крюкова, Б.И. Мызникова. - 2-е изд.,испр.и доп. - СПб.; М.; Краснодар : Лань, 2011. - 495 с. - (Учебники для вузов. Специальная литература). - Предм.указ.:с.491-495. - Библиогр.:с.489-490. - ISBN 978-5-8114-1246-4 : 465-00.:**

[**https://dpm.pstu.ru/images/R/Z/shevcov\_lineynaya\_algebra.pdf**](https://dpm.pstu.ru/images/R/Z/shevcov_lineynaya_algebra.pdf)