МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«**Кубанский государственный университет»

Факультет компьютерных технологий и прикладной математики

Кафедра математического моделирования

**Отчет о научно-исследовательской работе**

(практике по получению первичных навыков научно-исследовательской работы)

Выполнил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Д.Е Гиренко

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Курс 2

Руководитель учебной практики

канд. физ.-мат. наук, доцент кафедры

математического моделирования \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ С.Е. Рубцов

Краснодар

2022г.

**СОДЕРЖАНИЕ**

[1 Постановка задачи 3](#_Toc101868748)

[2 Описание метода Эйлера 4](#_Toc101868749)

[3 Аналитическое решение задачи Коши 5](#_Toc101868750)

[4 Описание программы для численного решения задачи Коши 7](#_Toc101868751)

[5 Список литературы 13](#_Toc101868752)

[Приложение 14](#_Toc101868753)

**1 Постановка задачи**

Дано дифференциальное уравнение: , .

* Методом ломаных Эйлера получить приближенное решение задачи Коши для заданного дифференциального уравнения. Начальное условие,.
* Вычисления произвести при помощи программы, разработанной лично Вами на языке высокого уровня для различных значений N (например, при N=5, 20, 100). В программе предусмотреть ввод N.
* Получить аналитически точное решение задачи Коши.
* В одной системе координат построить графики точного и приближенного решений. Вычислить максимальную невязку (наибольшую по абсолютной величине разность между точными приближенным решениями для различных значений ).

**2 Описание метода Эйлера**

Метод Эйлера – простейший численный метод решения систем обыкновенных дифференциальных уравнений. Впервые описан Леонардом Эйлером в 1768 году в работе «Интегральное исчисление». Метод Эйлера является явным, одношаговым методом первого порядка точности. Он основан на аппроксимации интегральной кривой кусочно-линейной функцией, так называемой ломаной Эйлера.

Пусть дана задача Коши для уравнения первого порядка:

, ,

где функция определена на некоторой области . Решение ищется на интервале . На этом интервале введем узлы: . Приближенное решение в узлах , которое обозначим через , определяется по формуле:

Эти формулы непосредственно обобщаются на случай систем обыкновенных дифференциальных уравнений.

Что касается поставленной задачи, уравнение для численного решения имеет вид:

**3 Аналитическое решение задачи Коши**

Решим дифференциальное уравнение и задачу Коши для него:

,

Умножим на

Пусть и тогда . Тогда получаем

Получим соответствующее ЛОДУ и преобразуем для интегрирования

Умножим обе части на и проинтегрируем их

;

;

Потенцируем и дифференцируем

;

;

Подставим в изначальное ЛНДУ и сократим одинаковые части

Преобразуем, умножим обе части на и проинтегрируем

;

Заметим, что . Воспользуемся методом внесения под знак дифференциала и получим результат интегрирования

Перейдём обратно к

Решим поставленную ранее задачу Коши

Отсюда получаем, что  и имеем

;

**4 Описание программы для численного решения задачи Коши**

Использована платформа разработки Visual Studio 2019. Визуализация графиков проводится с помощью NuGet пакета ZedGraph. Пользовательский интерфейс реализует библиотека Windows Forms.

В самом начале программы проинициализируем данные объектов формы Windows Forms: подписи кнопок и текстовых окон.

Реализуем две функции, первая из которых позволяет получать значения по различным точного решения задачи Коши, а вторая для приближенного (входное дифференциальное уравнение, где выражен ). Укажем начальные условия для задачи Коши в двух переменных.

Далее идет метод build, в котором происходит расчет и отрисовка графиков на GraphPane. Идет настройка окна вывода графика для более удобной работы с ним. Следующим шагом находим шаг разбиения, согласно указанному пользователем в отдельной поле. Циклично проходя по заданному интервалу, с ранее рассчитанным шагом, вызываем функцию с точным решением, заполняем массив точек, размерности 1001, тем самым получая точки графика точного решения. По данным точкам строится график с помощью метода AddCurve с настройкой цвета и подписью. Ещё одним циклом записываем все точки для построения приближенного решения. Следующая точка находится с помощью метода ломанных Эйлера. В цикле попутно считая максимальное значение невязки. Аналогично предыдущему графику, выводим текущие точки на координатную плоскость, а в текстовое окно значение максимальной невязки.

Основным “двигателем” метода являются данные строки цикла:

где

– рассчитанное ранее значение шага по заданному N,

– функция, возвращающая значение для входных и .

Последний метод в программе вызывается по нажатию кнопки в окне пользователя. Проверяется значение , введенное в окно для ввода текста на предмет некорректного ввода и далее вызывается метод build, который, и строит график точного решения задачи Коши и приближенного, согласно введенным данным.

На изображениях ниже представлен график точного решения поставленной задачи Коши, а также несколько вариантов работы программы с различными входными данными и, также, графиком точного решения.

Максимальное значения невязки при различном

* При , значение , рисунок 2
* При , значение , рисунок 3
* При , значение , рисунок 4
* При , значение , рисунок 5
* При , значение , рисунок 6

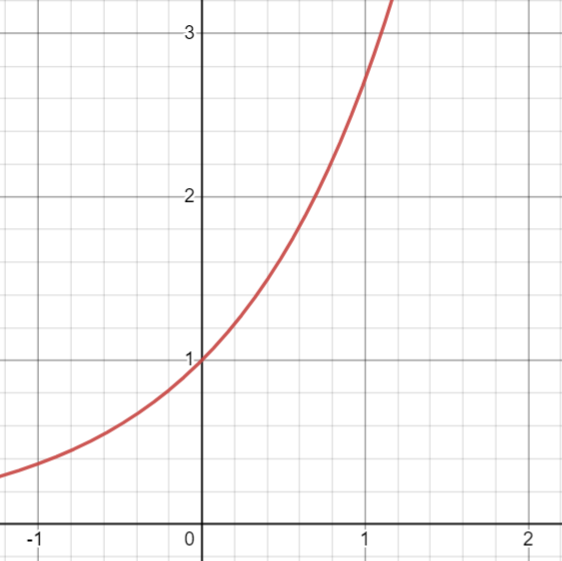


Рисунок 1 – График полученной функции, построенный с помощью Desmos

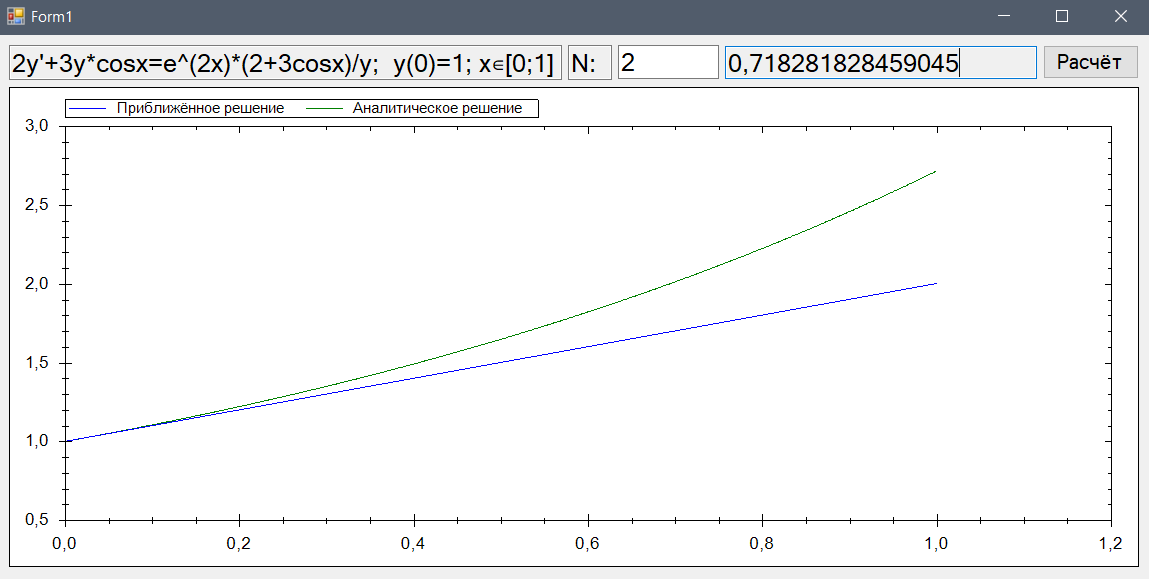


Рисунок 2 – Результат и исходная функция для

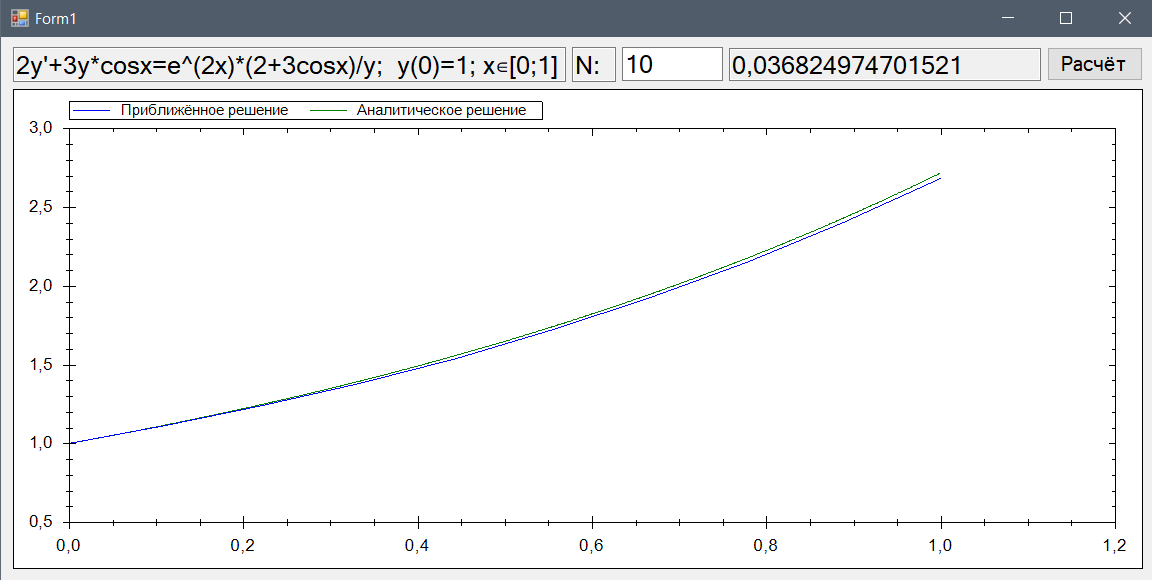


Рисунок 3 – Результат и исходная функция для

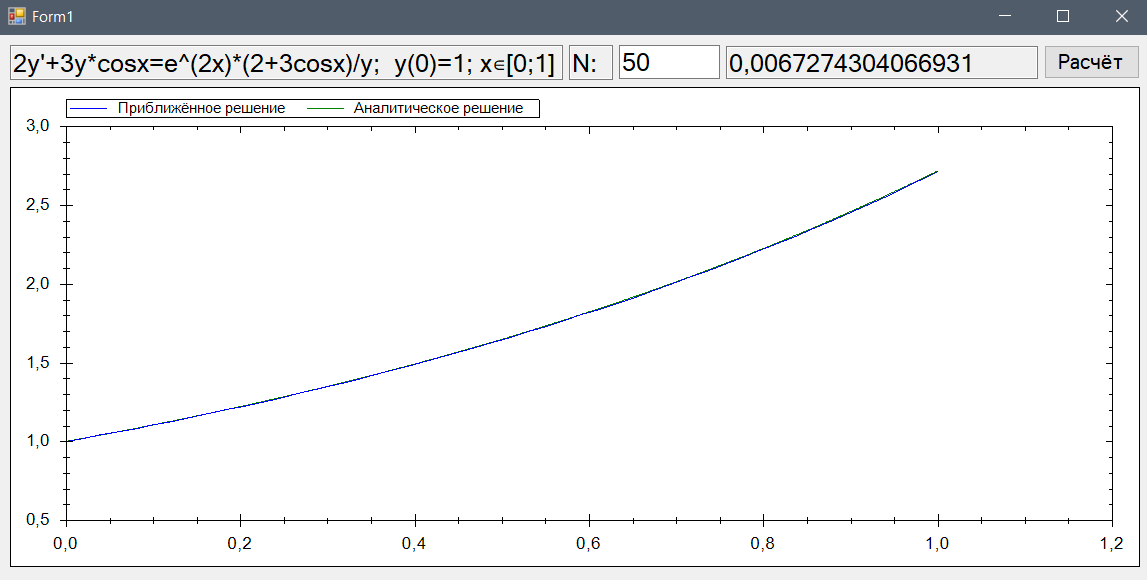


Рисунок 4 – Результат и исходная функция для

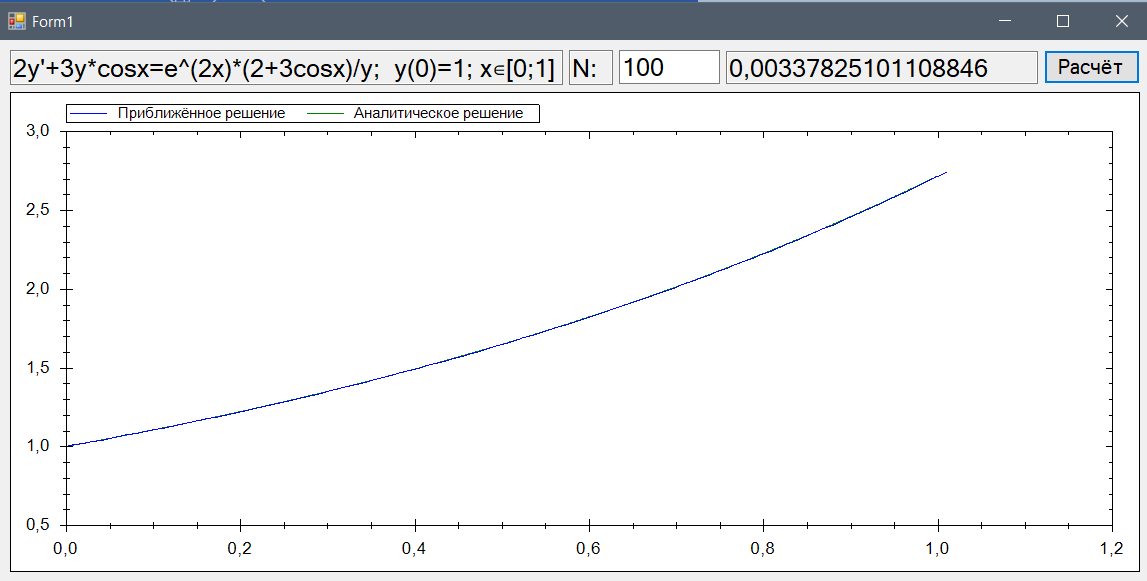


Рисунок 5 – Результат и исходная функция для

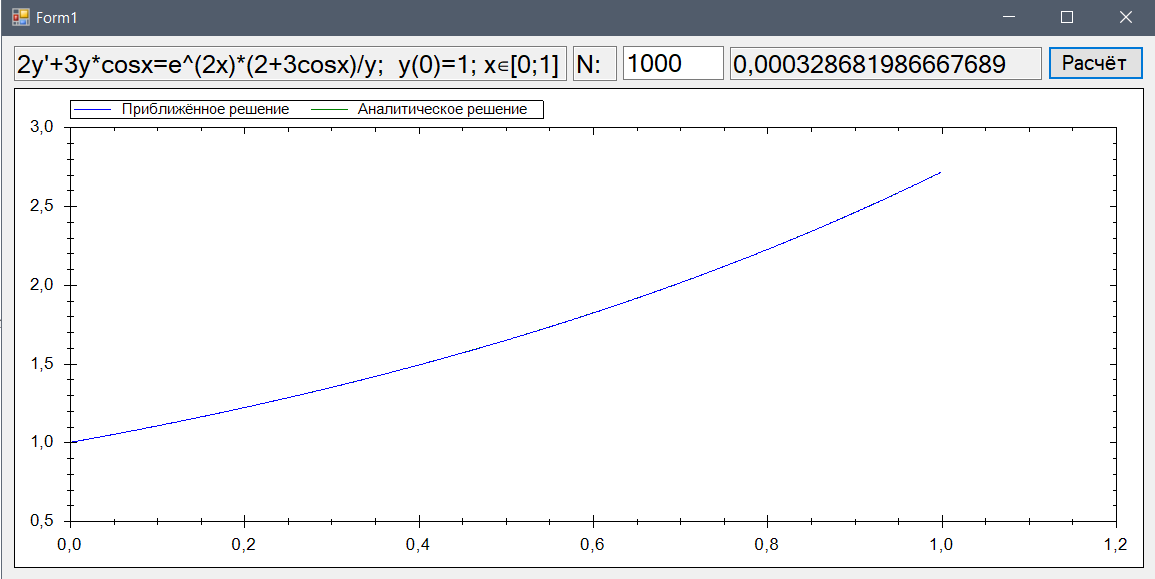


Рисунок 6 – Результат и исходная функция для

**5 Список литературы**

1. Пантелеев, А.В. Обыкновенные дифференциальные уравнения в примерах и задачах / А.В. Пантелеев. – М.: Высшая школа, 2001. – 376 c.
2. Бишоп, Дж. C# в кратком изложении / Дж. Бишоп, Н. Хорспул. – М.: Бином. Лаборатория знаний, 2013. – 472 c.
3. Несен, А.В. Microsoft Word 2010. От новичка к профессионалу / А.В. Несен. – М.: Солон-Пресс, 2016. – 320 c.

**Приложение**

using System;

using System.Drawing;

using System.Text;

using System.Windows.Forms;

using ZedGraph;

namespace metodEiler

{

public partial class Form1 : Form

{

double n, e;

// условия Каши и задание отрезка

const double x0=0, y0=1, minX = 0, maxX = 1;

// дифференциальное уравнение

static public double f(double x, double y)

{

return Math.Exp(2 \* x) \* (2 + 3 \* Math.Cos(x)) / (2 \* y) - 3 \* y \* Math.Cos(x) / 2;

}

//точное решение задачи Каши

static public double realF(double x)

{

return Math.Exp(x);

}

// метод очистки и настройки координатной плоскости

private void Clear(ZedGraphControl Zed\_GraphControl)

{

zedGraphControl1.GraphPane.CurveList.Clear();

zedGraphControl1.GraphPane.GraphObjList.Clear();

zedGraphControl1.GraphPane.XAxis.Type = AxisType.Linear;

zedGraphControl1.GraphPane.XAxis.Scale.TextLabels = null;

zedGraphControl1.GraphPane.XAxis.MajorGrid.IsVisible = false;

zedGraphControl1.GraphPane.YAxis.MajorGrid.IsVisible = false;

zedGraphControl1.GraphPane.YAxis.MinorGrid.IsVisible = false;

zedGraphControl1.GraphPane.XAxis.MinorGrid.IsVisible = false;

zedGraphControl1.GraphPane.XAxis.Title.IsVisible = false;

zedGraphControl1.GraphPane.YAxis.Title.IsVisible = false;

zedGraphControl1.GraphPane.Title.IsVisible = false;

zedGraphControl1.RestoreScale(zedGraphControl1.GraphPane);

zedGraphControl1.AxisChange();

zedGraphControl1.Invalidate();

}

// построение графиков

public void build(ZedGraphControl Zed\_GraphControl)

{

PointPairList list = new PointPairList();

PointPairList reallist = new PointPairList();

double h, h1, x=x0, y=y0;

double nowE, maxE = 0;

h = (maxX - minX) / (n-1);

h1 = (double)(maxX - minX) / 1000D;

// нахождение точек точной функции

while (x < x0+h1\*1000D)

{

reallist.Add(x, realF(x));

x += h1;

}

// нахождение точек приближённой функции

x = x0;

while (x < x0 + h \* n)

{

list.Add(x, y);

// вычисление максимальной невязки

nowE = Math.Abs(y - realF(x));

if (nowE > maxE)

maxE = nowE;

y += h \* f(x, y);

x += h;

}

textBox3.Text = Convert.ToString(maxE);

// отрисовка графиков функций

GraphPane my\_Pane = Zed\_GraphControl.GraphPane;

LineItem myCircle = my\_Pane.AddCurve("Приближённое решение", list, Color.Blue, SymbolType.None);

LineItem myCircle2 = my\_Pane.AddCurve("Аналитическое решение", reallist, Color.Green, SymbolType.None);

zedGraphControl1.AxisChange();

zedGraphControl1.Invalidate();

}

public Form1()

{

InitializeComponent();

Clear(zedGraphControl1);

}

// при нажатии на кнопку расчёт

private void button1\_Click(object sender, EventArgs ee)

{

if (textBox6.Text != "")

try

{ n = Convert.ToDouble(textBox6.Text); }

catch { return; }

Clear(zedGraphControl1);

build(zedGraphControl1);

}

}

}