

Национальный исследовательский университет ИТМО

Факультет инфокоммуникационных технологий

**Лабораторная работа 8**

Выполнил: Орел

Даниил Максимович

Группа № K3221

Проверил: Иванов С.Е.

Санкт-Петербург

2020

**Цель работы:**

Написать программу (C#) решения задачи для дифференциального уравнения

второго порядка методами:

1. Последовательных приближений, количество приближений оценивать всоответствии с точностью (формула в лекции).
2. Прогноза и коррекции. Для расчета начальных значений применить метод РунгеКутта.
3. Адамса. Для расчета начальных значений применить метод Рунге-Кутта.

**Ход работы:**

1. В программе были реализованы метод последовательных приближений, функции, реализующие метод прогноза и коррекции и метод Адамса:

using System;

using Expr = MathNet.Symbolics.SymbolicExpression;

namespace Lab\_08

{

class Program

{

static void Main(string[] args)

{

//объявление переменных

var x = Expr.Variable("x");

var y1 = Expr.Variable("y1");

var y2 = Expr.Variable("y2");

#region Zadanie 1

//Метод последовательных приближений для решения системы

//Объявление системы

Expr expr\_dy1 = (x + y1 \* y2);

Expr expr\_dy2 = (x \* x - y1 \* y1);

Func<double, double, double, double> dy1 = (expr\_dy1).Compile("x", "y1", "y2");

Func<double, double, double, double> dy2 = (expr\_dy2).Compile("x", "y1", "y2");

//Первое приближение

Expr expr\_y1\_1 = (1 + x \* x / 2);

Expr expr\_y2\_1 = (-x + x \* x \* x / 3);

Func<double, double> y1\_1 = expr\_y1\_1.Compile("x");

Func<double, double> y2\_1 = expr\_y2\_1.Compile("x");

//Второе приближение

Expr expr\_y1\_2 = (1 - (x.Pow(4)) / 24 + (x.Pow(6)) / 36);

Expr expr\_y2\_2 = (-x - x.Pow(5) / 20);

Func<double, double> y1\_2 = expr\_y1\_2.Compile("x");

Func<double, double> y2\_2 = expr\_y2\_2.Compile("x");

//диапозон значений x

double[] section\_x = new double[2] { 0, 1 };

//Количество делений

int divisions = 10;

//Определение величны шага

double h = (section\_x[1] - section\_x[0]) / divisions;

double[] x\_val = new double[divisions + 1];

x\_val[0] = section\_x[0];

for (int i = 1; i <= divisions; i++)

{

x\_val[i] = x\_val[i - 1] + h;

}

//массивы для хранения значений y1 и y2

double[] y1\_val = new double[divisions + 1];

double[] y2\_val = new double[divisions + 1];

y1\_val[0] = 1;

y2\_val[0] = 0;

//Печать начальных условий

Console.WriteLine($"y1' = {expr\_dy1.ToString()}");

Console.WriteLine($"y2' = {expr\_dy2.ToString()}");

Console.WriteLine("Начальные условия: ");

Console.WriteLine("y1(0) = 1; y2(0) = 0;");

//Печать приближений

Console.WriteLine("Исходя из формулы y\_n = y0 + integrate (f(x, y\_(n-1)))dx from 0 to x , получаем:\n");

Console.WriteLine($"y1\_1 = 1 + integrate (x + 0)dx from 0 to x = {expr\_y1\_1.ToString()}");

Console.WriteLine($"y2\_1 = 1 + integrate (x^2 - 1)dx from 0 to x = {expr\_y2\_1.ToString()}\n");

Console.WriteLine($"y1\_2 = 1 + integrate (x + (1 + x^2/2)\*(-x + x^3/3))dx from 0 to x = {expr\_y1\_2.ToString()}");

Console.WriteLine($"y2\_2 = 1 + integrate (x^2 + (1 + x^2 + x^4/4))dx from 0 to x = {expr\_y2\_2.ToString()}\n");

//Результат

Console.WriteLine($"x\ty1\_1\ty2\_1\ty1\_2\ty2\_2");

for (int i = 0; i <= divisions; i++)

{

Console.Write($"{x\_val[i]:0.00}\t");

Console.Write($"{y1\_1(x\_val[i]):0.0000}\t");

Console.Write($"{y2\_1(x\_val[i]):0.0000}\t");

Console.Write($"{y1\_2(x\_val[i]):0.0000}\t");

Console.WriteLine($"{y2\_2(x\_val[i]):0.0000}\t");

}

Console.WriteLine();

#endregion

Expr expr\_f1 = - 2 \* y1 + 4 \* y2;

Func<double, double, double, double> f1 = (expr\_f1).Compile("x", "y1", "y2");

Expr expr\_f2 = - y1 + 3 \* y2;

Func<double, double, double, double> f2 = (expr\_f2).Compile("x", "y1", "y2");

Console.WriteLine($"Уравнение:\ny1' = {expr\_f1.ToString()}\ny2' = {expr\_f2.ToString()}\n");

Console.WriteLine("Начальные условия: ");

Console.WriteLine("y1(0) = 3; y2(0) = 0;\n");

#region Zadanie 2

Miln(f1, f2, 10, 0, 1, 3, 0);

#endregion

#region Zadanie 3

Adams(f1, f2, 10, 0, 1, 3, 0);

#endregion

}

private static void Adams(Func<double, double, double, double> f1, Func<double, double, double, double> f2,

int divisions, double start\_point, double end\_point, double y1\_start, double y2\_start)

{

Console.WriteLine("Метод Адамса:\n");

double h = (end\_point - start\_point) / divisions;

double k1, k2, k3, k4, l1, l2, l3, l4;

double[] y1, y2, x;

x = new double[divisions + 1];

y1 = new double[divisions + 1];

y2 = new double[divisions + 1];

x[0] = start\_point;

y1[0] = y1\_start;

y2[0] = y2\_start;

for(int i = 0; i < 4; i++)

{

k1 = h \* f1(x[i], y1[i], y2[i]);

l1 = h \* f2(x[i], y1[i], y2[i]);

k2 = h \* f1(x[i] + h / 2, y1[i] + k1 / 2, y2[i] + l1 / 2);

l2 = h \* f2(x[i] + h / 2, y1[i] + k1 / 2, y2[i] + l1 / 2);

k3 = h \* f1(x[i] + h / 2, y1[i] + k2 / 2, y2[i] + l2 / 2);

l3 = h \* f2(x[i] + h / 2, y1[i] + k2 / 2, y2[i] + l2 / 2);

k4 = h \* f1(x[i] + h, y1[i] + k3, y2[i] + l3);

l4 = h \* f2(x[i] + h, y1[i] + k3, y2[i] + l3);

y1[i + 1] = y1[i] + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6;

y2[i + 1] = y2[i] + (l1 + 2 \* l2 + 2 \* l3 + l4) / 6;

x[i + 1] = x[i] + h;

}

for (int i = 4; i <= divisions; i++)

{

x[i] = x[i - 1] + h;

y1[i] = y1[i - 1] + (h / 24.0) \*

(55 \* f1(x[i - 1], y1[i - 1], y2[i - 1]) -

59 \* f1(x[i - 2], y1[i - 2], y2[i - 2]) +

37 \* f1(x[i - 3], y1[i - 3], y2[i - 3]) +

-9 \* f1(x[i - 4], y1[i - 4], y2[i - 4]));

y2[i] = y2[i - 1] + (h / 24.0) \*

(55 \* f2(x[i - 1], y1[i - 1], y2[i - 1]) -

59 \* f2(x[i - 2], y1[i - 2], y2[i - 2]) +

37 \* f2(x[i - 3], y1[i - 3], y2[i - 3]) +

-9 \* f2(x[i - 4], y1[i - 4], y2[i - 4]));

}

//Вывод ответа

Console.WriteLine($"x\ty1\ty2");

for (int i = 0; i <= divisions; i++)

{

Console.WriteLine($"{x[i]:0.00}\t{y1[i]:0.00}\t{y2[i]:0.00}");

}

}

private static void Miln(Func<double, double, double, double> f1, Func<double, double, double, double> f2,

int divisions, double start\_point, double end\_point, double y1\_start, double y2\_start)

{

Console.WriteLine("Метод Милна:\n");

double k1, l1, k2, l2, k3, l3, k4, l4;

double h = (end\_point - start\_point) / divisions;

double curr\_x = start\_point;

double[] x\_val = new double[divisions + 1];

double[] y1\_val = new double[divisions + 1];

double[] y2\_val = new double[divisions + 1];

double[] dy1 = new double[divisions + 1];

double[] dy2 = new double[divisions + 1];

x\_val[0] = start\_point;

dy1[0] = y1\_val[0] = y1\_start;

dy2[0] = y2\_val[0] = y2\_start;

// Решение системы уравнений методом Рунге-Кутта

for(int i = 0; i <= 3; i++)

{

k1 = h \* f1(x\_val[i], y1\_val[i], y2\_val[i]);

l1 = h \* f2(x\_val[i], y1\_val[i], y2\_val[i]);

k2 = h \* f1(x\_val[i] + h / 2, y1\_val[i] + k1 / 2, y2\_val[i] + l1 / 2);

l2 = h \* f2(x\_val[i] + h / 2, y1\_val[i] + k1 / 2, y2\_val[i] + l1 / 2);

k3 = h \* f1(x\_val[i] + h / 2, y1\_val[i] + k2 / 2, y2\_val[i] + l2 / 2);

l3 = h \* f2(x\_val[i] + h / 2, y1\_val[i] + k2 / 2, y2\_val[i] + l2 / 2);

k4 = h \* f1(x\_val[i] + h, y1\_val[i] + k3, y2\_val[i] + l3);

l4 = h \* f2(x\_val[i] + h, y1\_val[i] + k3, y2\_val[i] + l3);

y1\_val[i + 1] = y1\_val[i] + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6;

y2\_val[i + 1] = y2\_val[i] + (l1 + 2 \* l2 + 2 \* l3 + l4) / 6;

x\_val[i + 1] = x\_val[i] + h;

}

for(int i = 0; i <= 3; i++)

{

dy1[i] = f1(x\_val[i], y1\_val[i], y2\_val[i]);

dy2[i] = f2(x\_val[i], y1\_val[i], y2\_val[i]);

}

double curr\_y1, curr\_y2;

double dy1\_1, dy2\_1, dy1\_2, dy2\_2;

for(int i = 3; i < divisions; i++)

{

x\_val[i + 1] = x\_val[i] + h;

curr\_y1 = y1\_val[i - 3] + (4 \* h) / 3 \* (2 \* dy1[i] - dy1[i - 1] + 2 \* dy1[i - 2]);

curr\_y2 = y2\_val[i - 3] + (4 \* h) / 3 \* (2 \* dy2[i] - dy2[i - 1] + 2 \* dy2[i - 2]);

dy1\_1 = f1(x\_val[i + 1], curr\_y1, curr\_y2);

dy2\_1 = f2(x\_val[i + 1], curr\_y1, curr\_y2);

int count = 0;

while (true)

{

curr\_y1 = y1\_val[i - 1] + (h / 3) \* (dy1\_1 + 4 \* dy1[i] + dy1[i - 1]);

curr\_y2 = y2\_val[i - 1] + (h / 3) \* (dy2\_1 + 4 \* dy2[i] + dy2[i - 1]);

dy1\_2 = f1(x\_val[i + 1], curr\_y1, curr\_y2);

dy2\_2 = f2(x\_val[i + 1], curr\_y1, curr\_y2);

if (Math.Abs(dy2\_2 - dy2\_1) < 1e-4 && Math.Abs(dy1\_2 - dy1\_1) < 1e-4) break;

dy1\_1 = dy1\_2;

dy2\_1 = dy2\_2;

count++;

}

dy1[i + 1] = dy1\_2;

dy2[i + 1] = dy2\_2;

y1\_val[i + 1] = y1\_val[i - 1] + (h / 3) \* (dy1[i + 1] + 4 \* dy1[i] + dy1[i - 1]);

y2\_val[i + 1] = y2\_val[i - 1] + (h / 3) \* (dy2[i + 1] + 4 \* dy2[i] + dy2[i - 1]);

}

Console.WriteLine("x\ty1\ty2");

for (int i = 0; i < 11; i++)

{

Console.WriteLine($"{x\_val[i]:0.00}\t{y1\_val[i]:0.00}\t{y2\_val[i]:0.00}");

}

Console.WriteLine();

}

}

}

1. Результат работы программы:

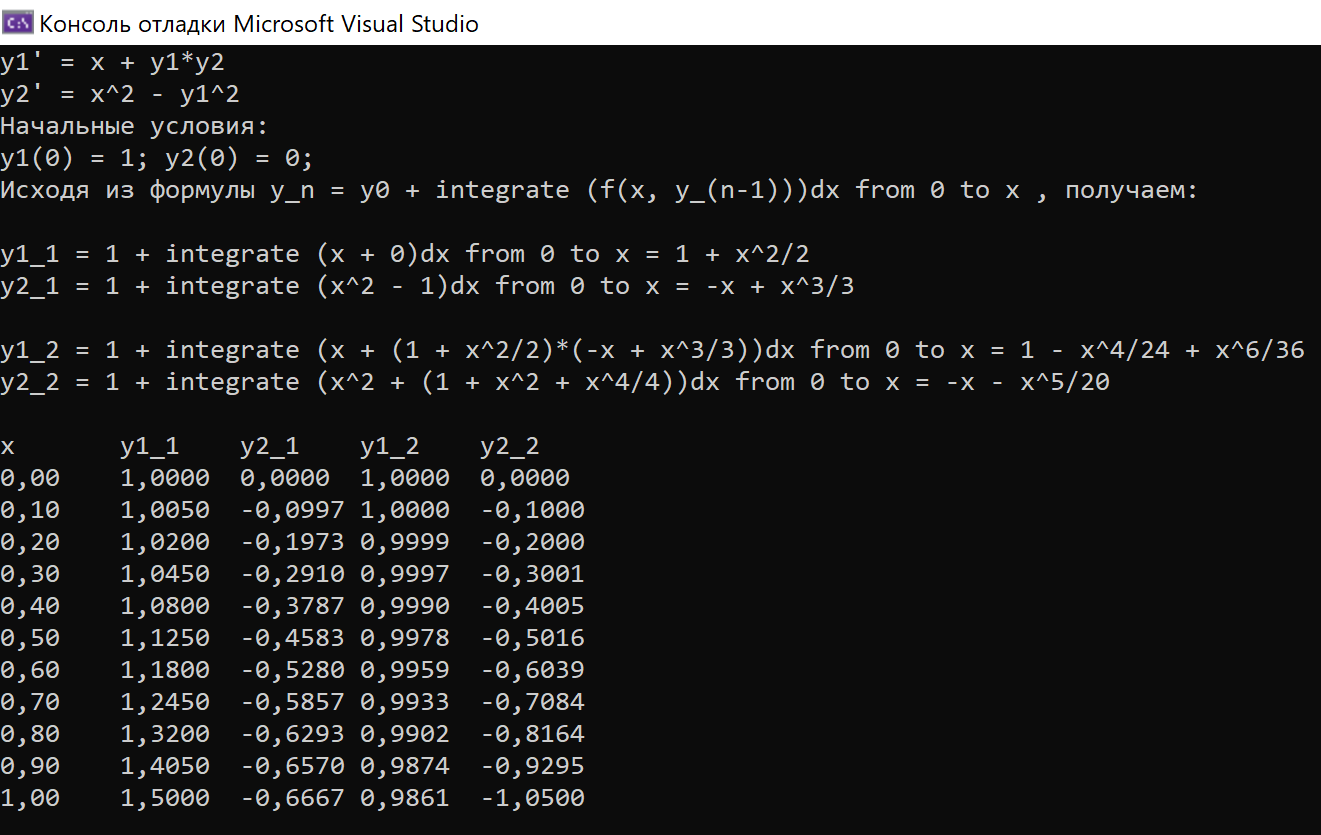


Рисунок 1 – Метод последовательных приближений



Рисунок 2 – Результаты решения одного и того же уравнения методом прогноза и коррекции и методом Адамса

**Вывод:**

В ходе выполнения данной практической работы были реализованы следующие методы решения ДУ: метод последовательных приближений, метод прогноза и коррекции и метод Адамса.