МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение высшего образования
«Нижегородский государственный университет им. Н.И. Лобачевского»
Национальный исследовательский университет

ЛАБОРАТОРНАЯ РАБОТА

Численное решение задачи Коши для ОДУ

Выполнил:
Антипин Александр Сергеевич 381706-2
____________ Подпись

Проверил:
Ассистент кафедры ДУМЧА:
Морозов Кирилл Евгеньевич

____________ Подпись

Нижний Новгород

2020

Оглавление

Введение

Дифференциальное уравнение — уравнение, в которое входят производные функции, и может входить сама функция, независимая переменная и параметры. Порядок входящих в уравнение производных может быть различен. Производные, функции, независимые переменные и параметры могут входить в уравнение в различных комбинациях или могут отсутствовать вовсе, кроме хотя бы одной производной.

В отличие от алгебраических уравнений, в результате решения которых ищется число (несколько чисел), при решении дифференциальных уравнений ищется функция (семейство функций).

Дифференциальное уравнение порядка выше первого можно преобразовать в систему уравнений первого порядка, в которой число уравнений равно порядку исходного дифференциального уравнения.

Современные быстродействующие ЭВМ эффективно дают численное решение обыкновенных дифференциальных уравнений, не требуя получения его решения в аналитическом виде.

Теория

Рассмотрим обыкновенное дифференциальное уравнение 2-го порядка, разрешенное относительно второй производной на отрезке с начальным условием или, как говорят, задачу Коши. Решением задачи Коши является функция, которая при подстановке ее в уравнение обращает данное уравнение в тождество, и удовлетворяет начальному условию.

Теорема существования и единственности утверждает, что если функция *f(x,y)* дифференцируема по *y* в окрестности точки плоскости (x_0, y_0) и ее производная в этой точке, то задача Коши однозначно разрешима в окрестности данной точки. Если функция $\frac{\partial f}{\partial y}$ дифференцируема на всем отрезке [a, b], то решение задачи Коши также существует на всем отрезке. При численном решении задач чаще всего требуется найти решение для значений $x > x_0$, т.е. $a=x_0$.

Одним из методов численного решения дифференциальных уравнений является метод **Рунге-Кутты**.

Метод Рунге-Кутты

Метод Рунге-Кутты, который, вообще говоря, является более точным и имеет большее практическое значение чем метод Эйлера, хотя идея метода схожа. Приращение функции приближенно выражается в виде линейной комбинации дифференциалов, вычисленных в текущем узле и в окрестных точках узла.

Рассмотрим уравнение второго порядка, разрешенное относительно второй производной:

$$y'' = f(x, y, y')$$

На отрезке [a, b] с начальным условием: $y(a) = y_0, \; y'(a) = y'_0$

Это уравнение легко свести к системе к системе уравнений первого порядка с помощью замены переменных $y' = z$. Тогда $Y'' = z'$

и уравнение сводится к системе первого порядка.

$$\begin{cases} y' = f_1(x, y, z) \\ z' = f_2(x, y, z) \end{cases} \qquad y(a) = y_0 \; , \; z(a) = z_0$$

Напишем формулы для системы двух уравнений:

k1 = z[i]

q1 = f(x[i], y[i], z[i])

k2 = z[i] + q1h/2
q2 = f(x[i] + h/2, y[i] + k1h/2, z[i] + q1h/2)

k3 = z[i] + q2h/2
q3 = f(x[i] + h/2, y[i] + k2h/2, z[i] + q2h/2)

k4 = z[i] + q2h
q4 = f(x[i] + h, y[i] + k3h, z[i] + q3h)

y[i+1] = y[i] + h(k1 + 2k2 + 2k3 + k4)/6
z[i+1] = z[i] + h(q1 + 2q2 + 2*q3 + q4)/6

Приведенные формулы k1, q1, k2, q2, k3, q3, k4, q4 последовательно вычисляются на каждом шаге, после чего вычисляются y[i+1] и z[i+1].

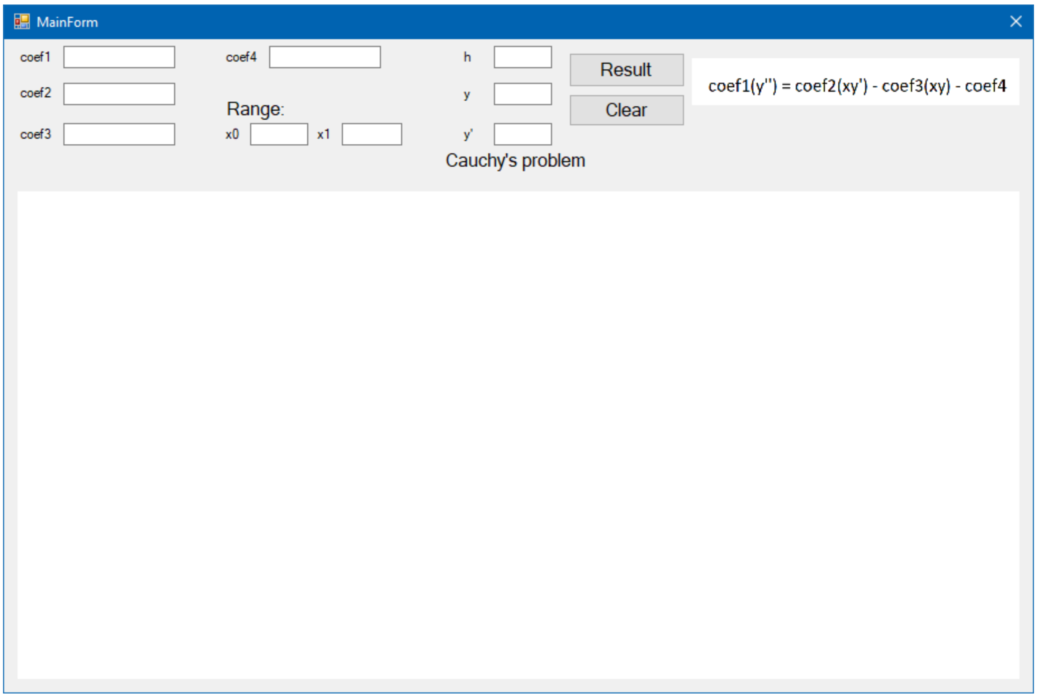
Программная реализация

Программа для решения дифференциального уравнения вида:

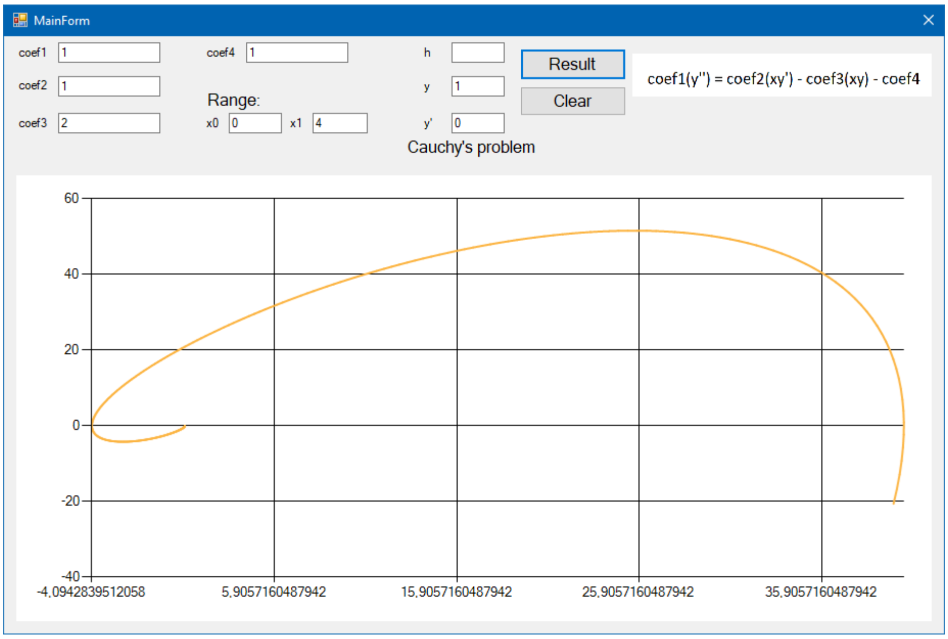
$$ay'' - bxy' + cxy = d$$

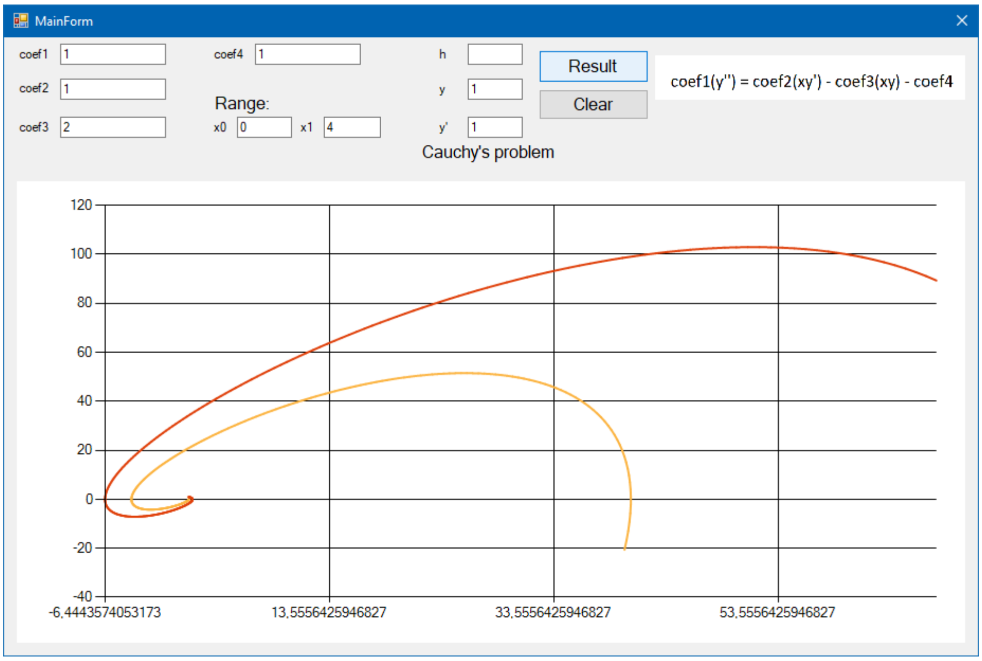
реализована на алгоритмическом языке высокого уровня C++ с использованием фреймворка Window Form.

На следующем рисунке приведен интерфейс программы, где:



coef1, coef2, coef3, coef4 – параметры дифференциального уравнения, x0 и x1 – интервалы нахождения функции, h – шаг с которым будет находиться значение функции, y и y' значения начальной задачи Коши. Поле h не является обязательным, если его не вводить, то по умолчанию она будет равно 0,01. Также дробные значения в поля должны вводиться через запятую, а не через точку. При нажатии на кнопу Result, строится фазовая траектория для данной задачи Коши. Кнопку Clear очищает график функции.





Далее представлены несколько скриншотов вывода графика в зависимости от заданных начальных условий.

Вывод

Ходе работы было сделано несколько выводов: метод Рунге-Кутта не сложен в реализации, и имеет высокую точность. Этот метод имеет четвёртый порядок точности. Это значит, что ошибка на одном шаге имеет порядок O(h5), а суммарная ошибка на конечном интервале интегрирования имеет порядок O(h4).

К достоинствам этого метода можно добавить то, что он является явными, т.е. значения yi+1находится по ранее найденным значениям y1,y2, … yi. Также метод допускает введения переменного шага h.

Список использованной литературы

- А.А.Самарский. Введение в численные методы М.: Наука, 1982.
- https://ru.wikipedia.org/wiki/Дифференциальное_уравнение
- https://portal.tpu.ru/SHARED/l/LOPATKIN/Students/DG/9-Problems_1-3.pdf

Приложение

Функция метода Рунге-Кутты:

```
void Runge_Kutt(std::vector<double>& x, std::vector<double>& y, std::vector<double>& z,
    double t0, double t1, double h) {
    size_t i = 0;
    while (t0 < t1 - h) {
        double k1 = z[i];
        double q1 = func(x[i], y[i], z[i]);

        double k2 = z[i] + q1 * h / 2;
        double q2 = func(x[i] + h / 2, y[i] + k1 * h / 2, z[i] + q1 * h / 2);

        double k3 = z[i] + q2 * h / 2;
        double q3 = func(x[i] + h / 2, y[i] + k2 * h / 2, z[i] + q2 * h / 2);

        double k4 = z[i] + q2 * h;
        double q4 = func(x[i] + h, y[i] + k3 * h, z[i] + q3 * h);

        double yt = y[i] + h * (k1 + 2 * k2 + 2 * k3 + k4) / 6;
        double zt = z[i] + h * (q1 + 2 * q2 + 2 * q3 + q4) / 6;

        x[i + 1] = x[i] + h;
        y[i + 1] = yt;
        z[i + 1] = zt;
        ++i;
        t0 += h;
    }
}
```