МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение Высшего образования

# «Нижегородский государственный университет им. Н.И. Лобачевского» Национальный исследовательский университет

**Институт информационных технологий, математики и механики Кафедра математического обеспечения и суперкомпьютерных технологий**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

*«Численное решение задачи Коши для ОДУ»*

**Выполнил:** студент группы 381706-1 Соколов А. Д.

Подпись

# Принял:

Эгамов А. И.

Подпись

Нижний Новгород 2020.

## Содержание

1. [Введение 3](#_bookmark0)
2. [Постановка задачи 4](#_bookmark1)
3. [Руководство пользователя 5](#_bookmark2)
4. [Описание алгоритмов 7](#_bookmark3)
   1. [Метод Рунге-Кутта 4-ого порядка 7](#_bookmark4)
5. [Заключение 8](#_bookmark5)
6. [Список литературы 9](#_bookmark6)
7. [Приложение 10](#_bookmark7)
   1. [Код для решения задачи Коши с заданным начальным условием 10](#_bookmark8)
   2. [Код для построения фазового портрета для заданного уравнения второго](#_bookmark9)

[порядка: 10](#_bookmark9)

Многие задачи физики, химии, экологии, механики и других разделов науки и техники при их математическом моделировании сводятся к дифференциальным уравнениям. Поэтому решение дифференциальных уравнений является одной из важнейших математических задач.

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 (формально он ничем не ограничен).

Производные, функции, независимые переменные и параметры могут входить в уравнение в различных комбинациях или могут отсутствовать вовсе, кроме хотя бы одной производной.

В вычислительной математике изучаются численные методы решения дифференциальных уравнений, которые особенно эффективны в сочетании с использованием персональных компьютеров.

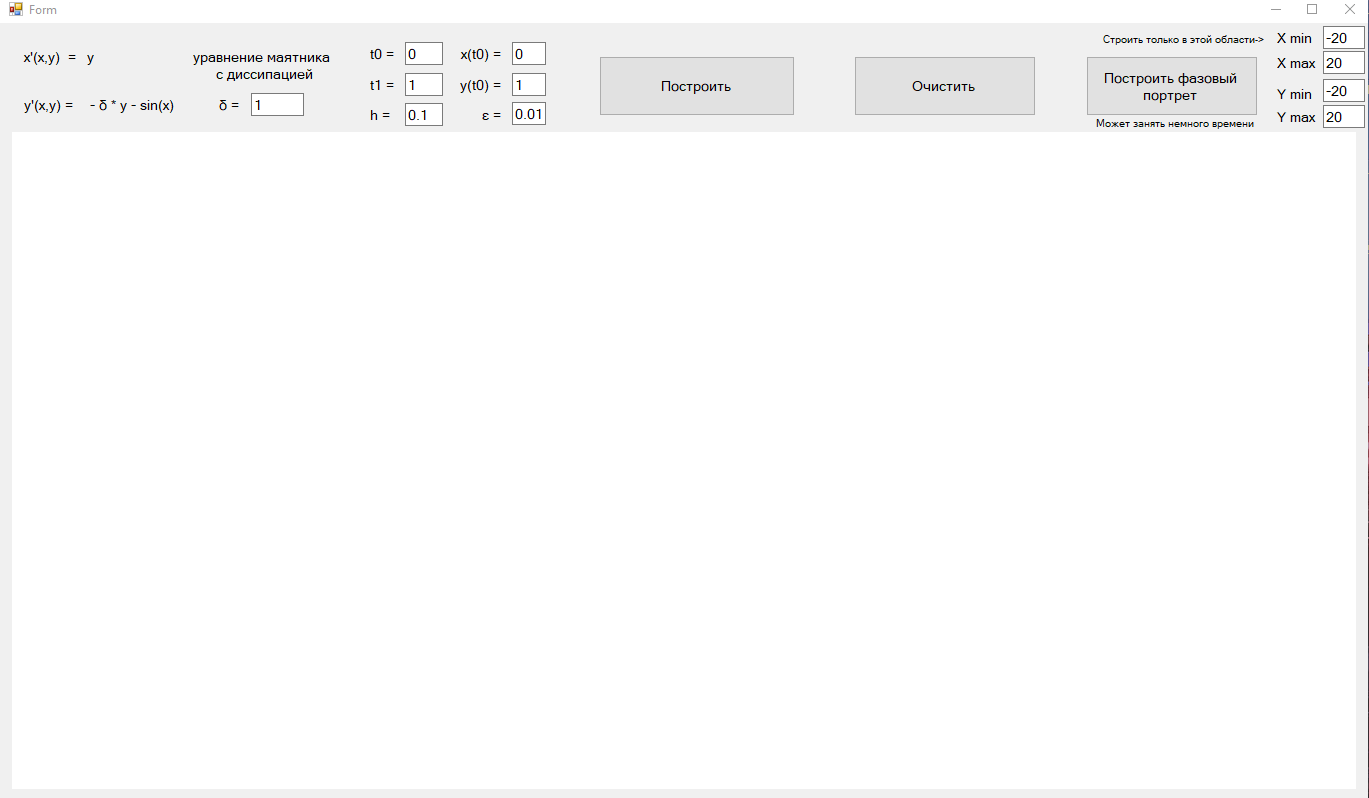
В этой лабораторной работе нам необходимо написать программу, выполняющую численном решении задачи Коши для автономного ОДУ второго порядка и вывод на экран результаты выполнения программы – проекция решений на фазовую плоскость.

Программа должна принимать параметры ДУ, локальную погрешность, отрезок интегрирования. Также должна присутствовать возможность очистить график без перезапуска программы.

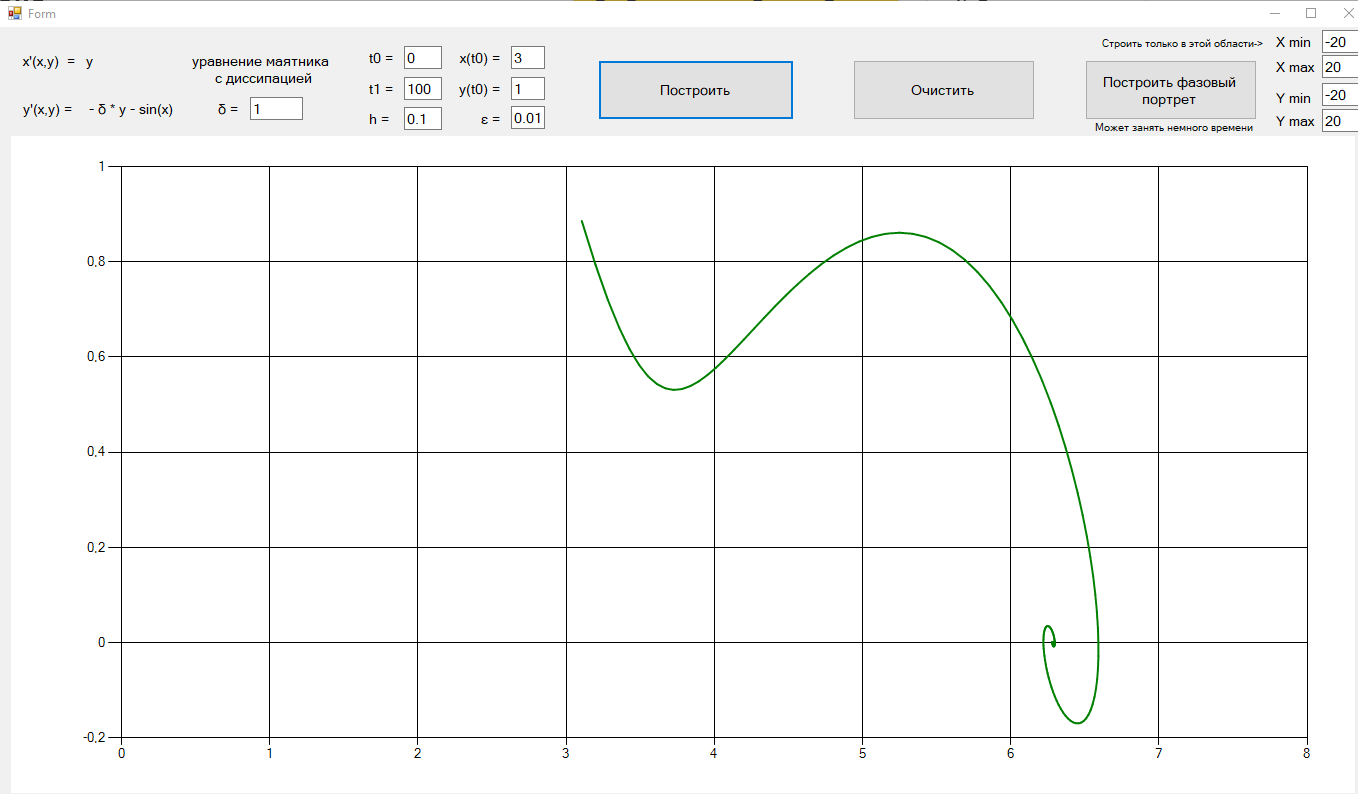
При решении ДУ уравнения будет использоваться метод Рунге-Кутта 4-ого порядка точности из-за его простоты и относительно хорошей точности вычислений. Программный код будет написан на языке C# из-за его распространённости и возможности построить графический интерфейс с помощью Windows Forms. В качестве исследуемого уравнения возьмем уравнения маятника с диссипацией, имеющий вид 

После запуска программы пользователь должен ввести параметры, с помощью которых будет построено решение задачи коши на фазовой плоскости.

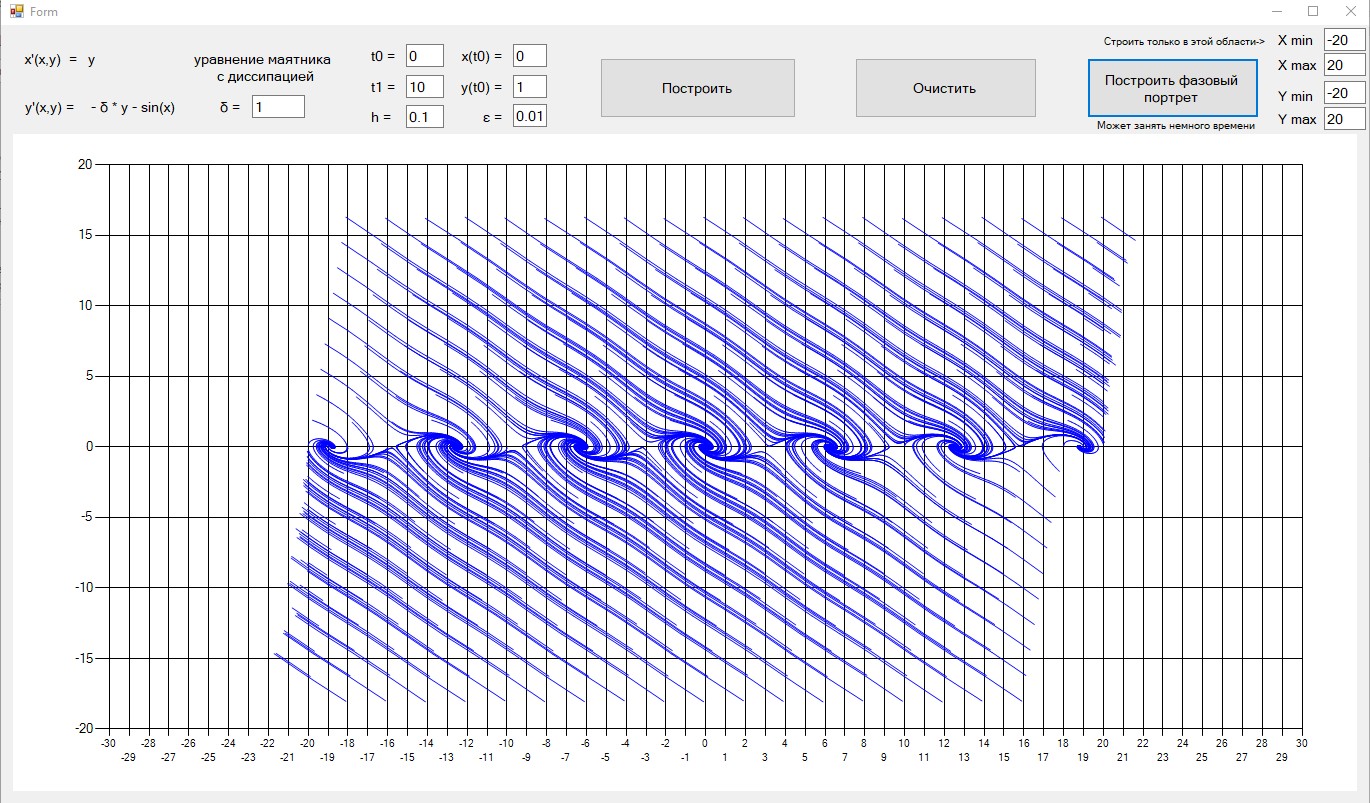
После ввода можно нажать кнопку «Построить». После этого появится на фазовой плоскости появиться решение задачи коши для введённых параметров.

Если же после введения параметров нажать на кнопку «Построить фазовый портрет», то на фазовую плоскость выведется фазовый портрет в области, указанной в параметрах.

*Рисунок 1. Программа до ввода параметров*



*Рисунок 2. Результат работы программы при решении задачи Коши*



*Рисунок 3. Результат работы программы при построении фазового портрета*

## 4.1. Метод Рунге-Кутта 4-ого порядка

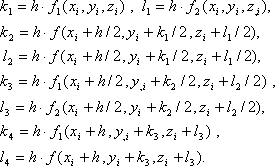
Метод позволяет решать системы обыкновенных дифференциальных уравнений (ОДУ) первого порядка следующего вида:

На отрезке с начальными условиями: 

Формулы метода Рунге-Кутта для системы из двух уравнений:



Где коэффициенты 𝑘𝑖 и 𝑙𝑖равны:



На каждом шаге алгоритма последовательно вычисляются сначала коэффициенты 𝑘𝑖 и 𝑙𝑖, а затем 𝑦𝑖+1 и 𝑧𝑖+1 . Полученные координаты используются для построения решения задачи Коши.

В данной работе была реализована программа, реализующая решение задачи Коши для автономного ДУ второго порядка. Был освоен численный метод Рунге-Кутта 4-ого порядка. Было исследовано уравнение маятника с диссипацией. В программе был реализован графический интерфейс для построение фазового портрета и решения задачи Коши.

1. Самарский А.А. Введение в численные методы – М.: Наука, 1989. – 271с.

## Код для решения задачи Коши с заданным начальным условием:

Вычисление значений коэффициентов для метода Рунге-Кутта 4-ого порядка: double k1 = functionX(x, y);

double k2 = functionX(x + 0.5d \* h \* k1, y + 0.5d \* h \* k1); double k3 = functionX(x + 0.5d \* h \* k2, y + 0.5d \* h \* k2); double k4 = functionX(x + h \* k3, y + h \* k3);

double l1 = functionY(x, y, delta);

double l2 = functionY(x + 0.5d \* h \* l1, y + 0.5d \* h \* l1, delta); double l3 = functionY(x + 0.5d \* h \* k2, y + 0.5d \* h \* k2, delta); double l4 = functionY(x + h \* l3, y + h \* l3, delta);

double k = 1d / 6d \* (k1 + 2 \* k2 + 2 \* k3 + k4); double l = 1d / 6d \* (l1 + 2 \* l2 + 2 \* l3 + l4);

где delta – константа в уравнении маятника с диссипацией

Вычисление координат на фазовой плоскости: x = x + h \* k;

y = y + h \* l;

После каждой итерации проверяется погрешность вычислений, если погрешность удовлетворительная, то отрисовываем полученную точку, иначе уменьшаем параметр h в два раза.

E = (E1 – E2) / 15;

где E1 –значении точки при обычном вычислении, а Е2 – значение точки при вычислении в два шага

if (Math.Abs(E) <= e)

{

}

else

{

}

Добавление точки на фазовую плоскость chart.Series[buildCount.ToString()].Points.AddXY(x, y);

h /= 2d;

/\*Далее идет повторное вычисление с измененным параметром\*/

## Код для построения фазового портрета для заданного уравнения второго порядка:

Функция построения фазового портрета почти не отличается от функции для решения задачи Коши. В ней перебираются точки в пределе заданной области с шагом, зависящим от количества построенных фазовых портретов на фазовой плоскости. По умолчанию идет решение для каждой второй точки.

double step = 2.0; if (iter > -2000) step = 3.0;

for (double i = xMin; i < xMax; i+= step)

for (double j = yMin; j < yMax; j+= step)

/\* Решение задачи Коши для текущей точки\*/