МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Отчет по учебной практике по дисциплине

«Вычислительная математика»

УП.170007636.ТД

Выполнил студент Маслеников М.В.

Факультет, группа ФКФН, ПО(аб)-71

Руководитель работы Вихтенко Э. М.

Виза:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(доработать, к защите и т.д.)

### 

### Хабаровск – 2019г.

**Содержание**

Введение 3

Постановка задачи 4

Задание на практику 4

Метод Рунге-Кутта 5

Программное обеспечение 6

Результаты тестирования 7

Результаты решения задачи 9

Заключение 13

Список используемой литературы 14

Приложение А 15

**Введение**

В ходе практической работы нам необходимо выполнить следующие пункты:

1) Написать программу решения задачи Коши для произвольной системы ОДУ заданным методом Рунге – Кутты.

2) Решить тестовую задачу с целью отладки написанной программы и экспериментального подтверждения теоретического порядка точности реализованного метода Рунге – Кутты.

3) Решить основную задачу — заданную задачу Коши, содержащую несколько параметров. Изучить влияние одного из параметров на качественные и количественные свойства решения.

Основная задача каждого задания представляет собой математическую модель из некоторой предметной области, например, физики, химии, биологии, медицины, астрономии, техники и так далее.

В первом пункте задания дается краткое описание предмета моделирования, основных предположений и законов, положенных в основу модели. Этот пункт написан для того, чтобы дать студенту некоторые представления о том, каким примерно образом была получена соответствующая математическая модель. Как правило, исходная модель записывается в терминах размерных зависимых и независимых переменных. Простым масштабированием (с математической точки зрения) от размерных переменных и уравнений осуществляется переход к безразмерным переменным и уравнениям (основной задаче), которые и решаются.

Стоит отметить, что числовые данные основных задач имеют математический смысл и подобраны так, чтобы обеспечить необходимые свойства решений.

**Постановка задачи**

Имеется реактор (цилиндрический сосуд объемом V, снабженный охлаждающим кожухом).

Будем предполагать, что реакционная смесь полностью перемешивается и что в реакторе происходит практически необратимая реакция. Объемные расходы , и входные концентрации компонентов и реакционной смеси, температура на входе , коэффициент теплопередачи, площадь поверхности теплообмена и теплота реакции считаются постоянными; молярные концентрации *A* и *B*, а также температура смеси и охлаждающей среды не зависят от координат и могут быть только функциями времени *t*. С учетом этих предположений модель проточного реактора, в котором происходят последовательные экзотермические реакции первого порядка типа *A → B → C*, можно описать с помощью следующих трех дифференциальных уравнений

1) *X′ = 1−X −DaXa(θ),*

2) *Y′ = −Y + DaXa(θ)−DaSY(θ),*

3) *θ′ = −θ + DaBXa(θ)−β (θ−) + DaBαSY (θ),*

где

*(θ) = exp(),*

*θ* — безразмерная температура, *X* и *Y* — безразмерные концентрации компонентов A и B. Уравнения дополняются начальными условиями

*X(0) =, Y(0) = , θ(0) = .*

**Задание на практику**

1. Напишите программу интегрирования задачи Коши для системы из *n* уравнений первого порядка *вида y′ = f(t,y), y(0) = , y(t) ∈* , на произвольном отрезке [a,b], используя метод Рунге – Кутты 3-го порядка точности с постоянным шагом h:

*= f(,),*

*= f( + h/2, + /2),*

*= f( + h, – + 2),*

*= + h( + 4 + )/6.*

2. Протестируйте программу на примере системы уравнений

= + (+ −1),

= + (+ −1),

на отрезке [0,5] с точным решением

= ,

= .

3. Для тестовой задачи постройте графики зависимости максимальной погрешности решения *e* и *e/* от выбранного шага *h*. Поясните результаты вычислений.

4. Решите исходную систему уравнений при помощи разработанной программы. Приведите наиболее характерные графики траекторий в фазовых координатах *(x,θ), (y,θ)* и графики *x(t), y(t), θ(t)* на интервале интегрирования при разных значениях *Da* из отрезка [0.2,0.7] при следующих исходных данныx:

= 0.4, = 0.5, = 1.3, *k* = 1, *γ* = 20000, *α* = *S* = 0.03, *β* = 4, = 0, *B* = 13. Найдите те значения *Da,* при которых в системе cуществует предельный цикл (автоколебательный режим работы реактора).

**Метод Рунге-Кутта**

Наиболее эффективными и часто встречаемыми методами решениями задачи Коши являются методы Рунге - Кутта. Они основаны на аппроксимации искомой функции *у(х)* в пределах каждого шага многочленом, который получен при помощи разложения функции *у(х)* в окрестности шага *h* каждой *i*-ой точки в ряд Тейлора: *u( + h) = u() + hu′() + u′′( + h), ∈ [0,1].*

Пусть требуется найти приближенное решение задачи *u′(t) = f(t,u(t)), t ∈ (a,b], u(a)=*, на равномерной с шагом h сетке узлов.

Поскольку в силу уравнения справедливо равенство *u′() = f(,u()),* то соотношение (5) перепишется в виде

*= + hf(,) + h,*

*= ,*

*u′′( + h).*

В этой формуле слагаемое = O() является малой величиной, если *h* достаточно мало. Отбрасывая его, придем к методу Эйлера*: = + hf(,),* *i* = 0,1,...,N −1, = . Эти соотношения позволяют вычислить приближенное решение в точке сетки , зная приближенное решение в предыдущей точке . Такие численные методы называются одношаговыми.

Методы Рунге – Кутты при *q* = 3:

Формулы имеют вид  *= + h( + + )*, где

*= f(,),*

*= f(+ h, + h),*

*= f( + h, + h + h).*

Получается следующая система из шести уравнений для определения восьми коэффициентов метода при m = 3:

= ,

= + ,

+ + = 1,

2( + ) = 1,

3( + ) = 1,

6 = 1.

Эта система имеет два семейства решений: двухпараметрическое со свободными параметрами c2 и c3, причем, c2 != c3 и c2 != 2/3, и однопараметрическое со свободным параметром (при = = 2/3).

В задаче с моим вариантом требуется написать программу, используя метод Рунге – Кутты 3-го порядка точности с постоянным шагом h:

*= f(,),*

*= f( + h/2, + h/2),*

*= f( + h, − h + 2h),*

*= + h( + 4 + )/6.*

**Программное обеспечение**

Программа была написана на языке C# в среде qt creator, так как программа содержит визуальные компоненты, при желании она может быть запущена на любом компьютере.

Для реализации программы было написано несколько функций, код которых можно увидеть в приложении А.

**Результаты тестирования**

В ходе решения тестовой задачи мы задаем отрезок [a, b] и шаг h.

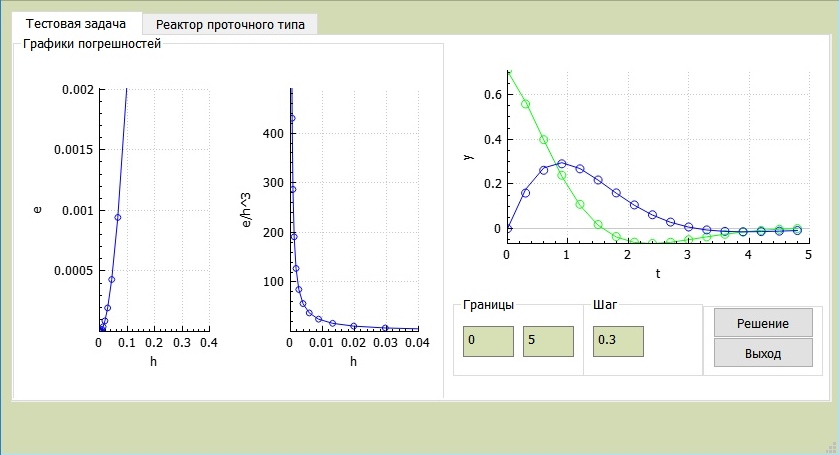


Рисунок 1 – Решение тестовой задачи при h=0,3

При h = 0,3 погрешность достаточно мала, это видно по точкам, которые лежат на графиках функции (рис. 1).

Однако если увеличить h до 0,5, то погрешность возрастает. Некоторые точки уже не лежат на графиках функции (рис. 2).

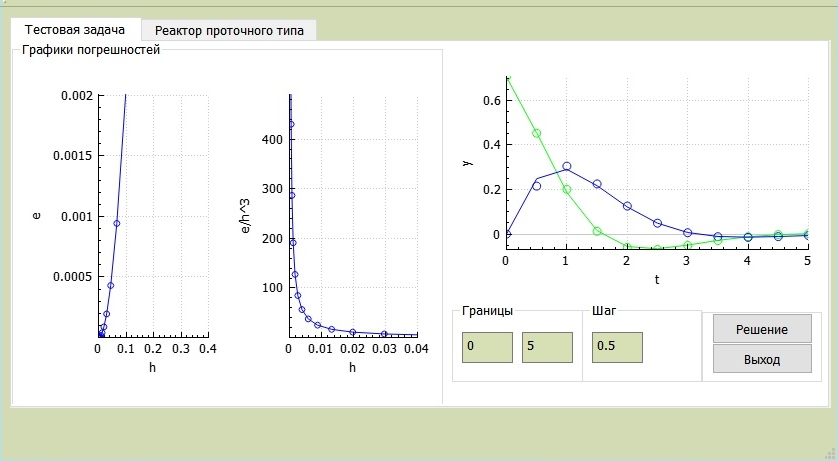


Рисунок 2 – Решение тестовой задачи при h=0,5

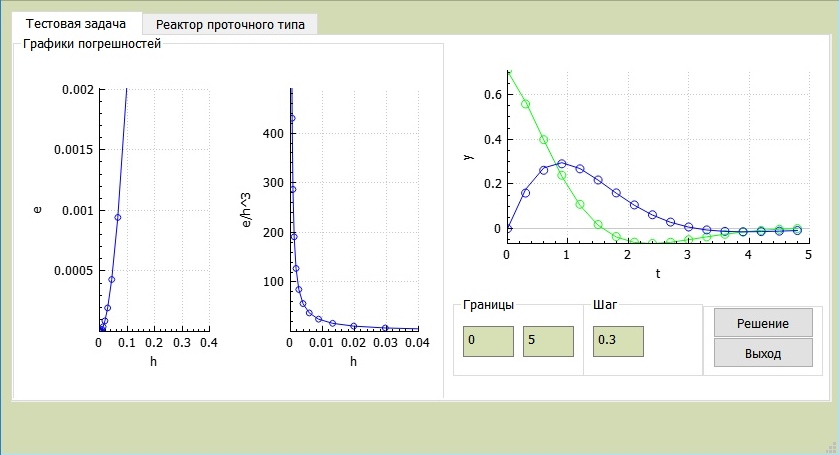


Рисунок 3 – Графики зависимости максимальной погрешности решения e и e/ от выбранного шага h.

**Результаты решения задачи**

В ходе решения задачи мы задаем отрезок [a, b], шаг h и значение Da.

На первых двух графиках мы имеем траектории в фазовых координатах (x,θ), (y,θ) (рис. 4, рис. 5).

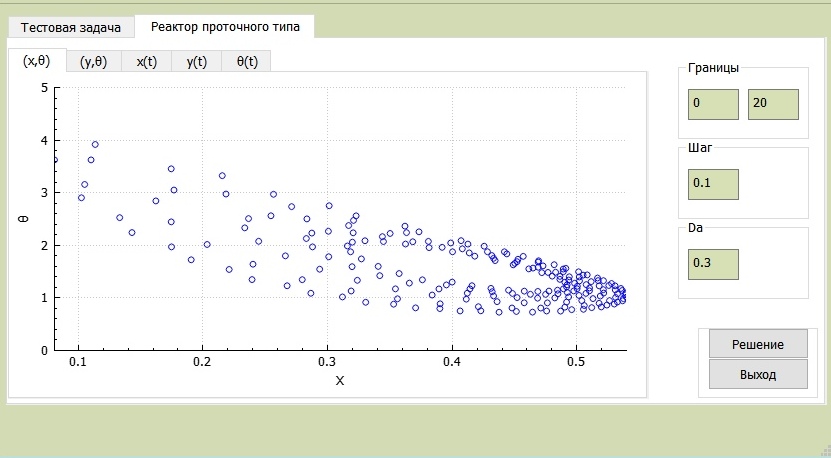
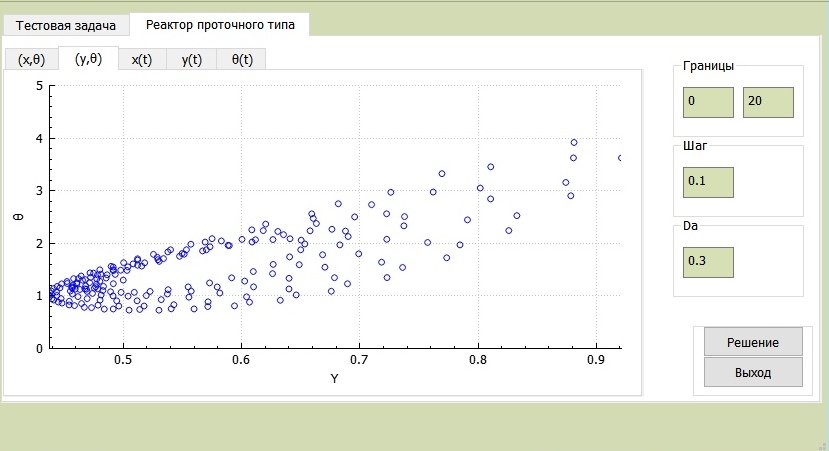


Рисунок 4 – траектории в фазовых координатах (x,θ) при шаге 0,1 и значении Da 0,3

Рисунок 5 – траектории в фазовых координатах (y,θ) при шаге 0,1 и Da = 0,3

На следующих трех рисунках мы имеем графики x(t), y(t), θ(t) (рис. 6, 7, 8).

При этом видно, что при значении Da, равном 0,3, со временем колебания на всех трех графиках увеличиваются.

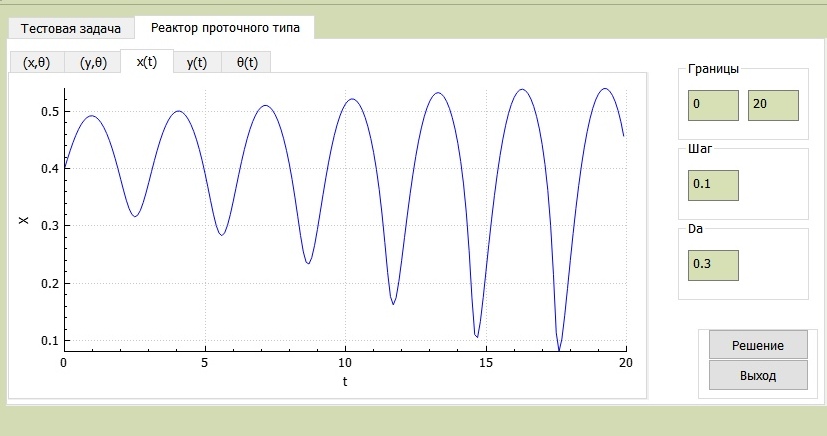


Рисунок 6 – график x(t) при шаге 0,1 и значении Da = 0,3

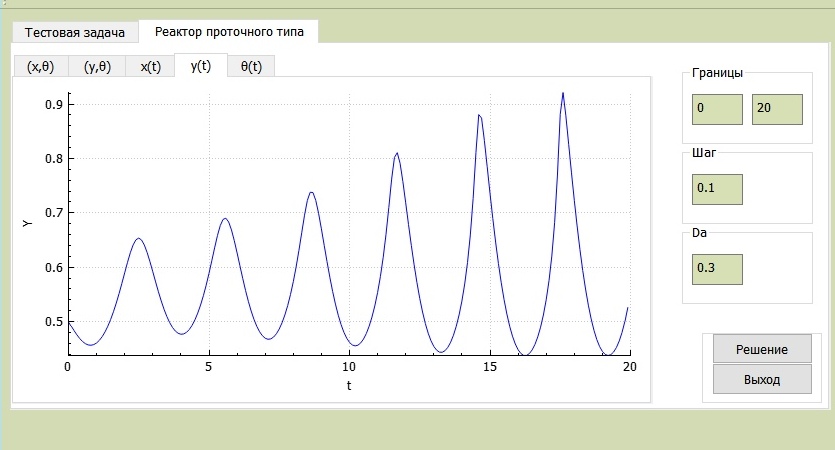


Рисунок 7 – график y(t) при шаге 0,1 и значении Da = 0,3

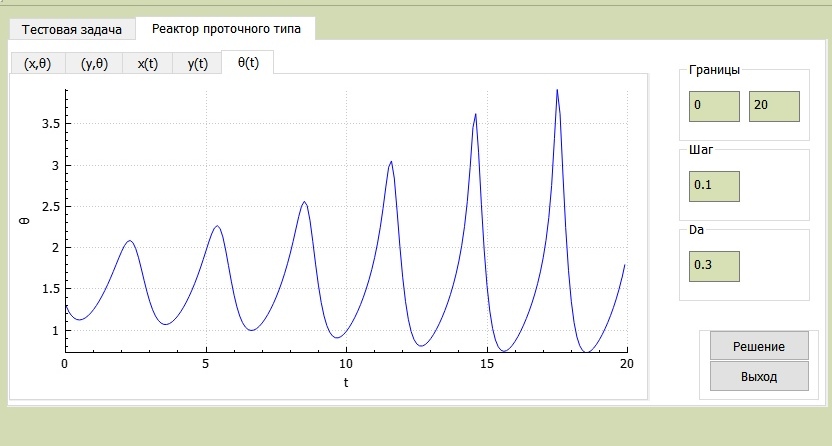


Рисунок 8 – график θ(t) при шаге 0,1 и значении Da = 0,3

При значении Da, равном 0,4, колебания остаются практически неизменными:

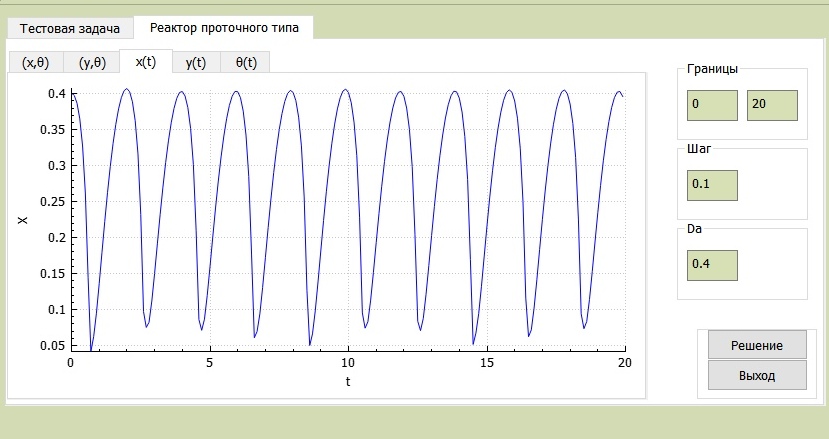


Рисунок 9 – график x(t) при шаге 0,1 и значении Da = 0,4

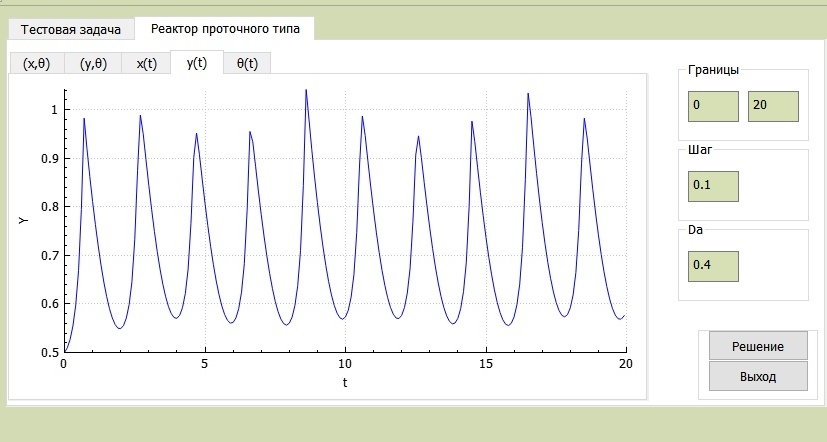


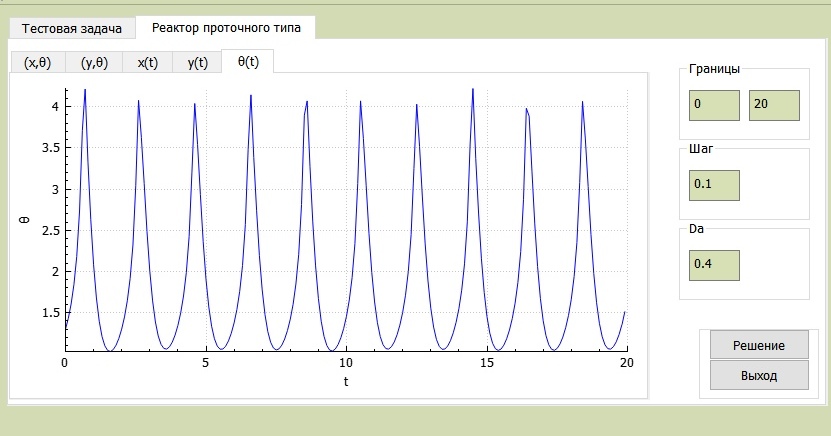
Рисунок 10 – график y(t) при шаге 0,1 и значении Da = 0,4

Рисунок 11 – график θ(t) при шаге 0,1 и значении Da = 0,4

При значении Da, которое больше чем 0,4, колебания затихают.

**Заключение**

В ходе практической работы рассмотрено решение задачи Коши методом Рунге-Кутты 3 порядка точности, примененное для решения задачи описания реактора проточного типа. Метод РГ реализован в программе, проведены расчеты на тестовом примере. В ходе вычислительных экспериментов найдено значение Da, при котором в системе cуществует предельный цикл (автоколебательный режим работы реактора).

**Список используемой литературы**

1. elementy.ru [Электронный ресурс]. Режим доступа: <https://elementy.ru/trefil/51/Teoriya_ravnovesiya_MakarturaUilsona> (дата обращения 17.04.2019)
2. Хайрер Э., Нерсетт С., Ваннер Г. Решение обыкновенных дифференциальных уравнений. – М.: Мир, 1990.
3. Даутов, Р.З. Практикум по курсу численные методы. Решение задачи Коши для системы ОДУ / P.З. Даутов. – Изд-во Казанск. федер. ун-та, 2014 г. – 100 с.
4. Холодниок М., Клич А., Кубичек М., Марек М. Методы анализа нелинейных динамических моделей. — М.: Мир, 1991.
5. Самарский А.А., Гулин А.В. Численные методы.— М.:Наука,1989. 4. Бахвалов Н.С., Жидков Н.П., Кобельков Г.М. Численные методы. — М.: Наука, 1987.

**Приложение А**

Код программы

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include "QStandardItemModel"

#include "QStandardItem"

#include "math.h"

#include "iostream"

double a, b, h;

QVector<double> t, TestY1, TestY2, Y1, Y2;

MainWindow::**MainWindow**(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

}

MainWindow::~***MainWindow***()

{

delete ui;

}

double MainWindow::**y1**(double t, double y1, double y2)

{

return (-sin(t) / (sqrt((1 + exp(2 \* t)))) + y1\*(y1\*y1 + y2\*y2 - 1));

}

double MainWindow::**y2**(double t, double y1, double y2)

{

return (cos(t)/sqrt((1+exp(2\*t)))+y2\*(y1\*y1+y2\*y2-1));

}

void MainWindow::**y12**()

{

ui->widget->clearGraphs();

ui->widget->addGraph();

ui->widget->graph(0)->setData(t, TestY1);

ui->widget->xAxis->setLabel("t");

ui->widget->yAxis->setLabel("y");

ui->widget->xAxis->setRange(a, b);

double minY = TestY1[0], maxY = TestY1[0];

for (int i=1; i<t.size(); i++)

{

if (TestY1[i]<minY) minY=TestY1[i];

if (TestY2[i]<minY) minY=TestY2[i];

if (TestY1[i]>maxY) maxY=TestY1[i];

if (TestY2[i]>maxY) maxY=TestY2[i];

}

ui->widget->graph(0)->setPen(QPen(Qt::green));

ui->widget->graph(0)->setLineStyle(QCPGraph::lsNone);

ui->widget->graph(0)->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 8));

ui->widget->addGraph();

ui->widget->graph(1)->setData(t,TestY2);

ui->widget->graph(1)->setPen(QPen(Qt::blue));

ui->widget->graph(1)->setLineStyle(QCPGraph::lsNone);

ui->widget->graph(1)->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 8));

ui->widget->yAxis->setRange(minY, maxY);

ui->widget->replot();

}

void MainWindow::**Test**()

{

ui->widget->addGraph();

ui->widget->graph(2)->setData(t,Y1);

ui->widget->graph(2)->setPen(QPen(Qt::green));

ui->widget->addGraph();

ui->widget->graph(3)->setData(t,Y2);

ui->widget->graph(3)->setPen(QPen(Qt::blue));

ui->widget->replot();

}

double MainWindow::**norm**(QVector <double> exY, QVector<double> apprY)

{

double max=0;

QVector<double> norma(exY.size());

for (int i=0; i<exY.size(); i++)

norma[i]=fabs(exY[i]-apprY[i]);

max=norma[0];

for (int i=0; i<exY.size(); i++)

if (fabs(norma[i])>max)

max=norma[i];

return max;

}

void MainWindow::**runge\_kutt**()

{

TestY1.push\_back(cos(a)/sqrt(1+exp(2\*a)));

TestY2.push\_back(sin(a)/sqrt(1+exp(2\*a)));

double k1\_1, k1\_2, k2\_1, k2\_2, k3\_1,k3\_2;

for(int i=0; i<t.size(); i++)

{

k1\_1 = y1(t[i], TestY1[i], TestY2[i]);

k1\_2 = y2(t[i], TestY1[i], TestY2[i]);

k2\_1 = y1(t[i] + h / 2, TestY1[i] + h\*k1\_1/2, TestY2[i] + h\*k1\_2/2);

k2\_2 = y2(t[i] + h / 2, TestY1[i] + h\*k1\_1/2, TestY2[i] + h\*k1\_2/2);

k3\_1 = y1(t[i] + h , TestY1[i] - h\*k1\_1 + 2\*h\*k2\_1, TestY2[i] - h\*k1\_2 + 2\*h\*k2\_2);

k3\_2 = y2(t[i] + h , TestY1[i] - h\*k1\_1 + 2\*h\*k2\_1, TestY2[i] - h\*k1\_2 + 2\*h\*k2\_2);

TestY1.push\_back(Y1[i] + h\*(k1\_1 + 3\*k3\_1) / 4);

TestY2.push\_back(Y2[i] + h\*(k1\_2 + 3\*k3\_2) / 4);

}

}

void MainWindow::**on\_pushButton\_clicked**()

{

t.clear();

TestY1.clear();

TestY2.clear();

Y1.clear();

Y2.clear();

QString text\_1=ui->textEdit->toPlainText();

a=text\_1.toDouble();

QString text\_2=ui->textEdit\_2->toPlainText();

b=text\_2.toDouble();

QString text\_3=ui->textEdit\_3->toPlainText();

h=text\_3.toDouble();

for (double i=a; i<=b; i+=h)//Пробегаем по всем точкам

{

t.push\_back(i);

}

for (int i=0; i<t.size(); i++)

{

Y1.push\_back(cos(t[i])/sqrt(1+exp(2\*t[i])));

Y2.push\_back(sin(t[i])/sqrt(1+exp(2\*t[i])));

}

runge\_kutt();

y12();

Test();

t.clear();

TestY1.clear();

TestY2.clear();

Y1.clear();

Y2.clear();

double e1, e2;

//

h = 0.1;

QVector <double> E1, E2, powE2, powE1, H;

for(int j=0; j<20; j++)

{

for (double i = a; i <= b; i += h)

t.push\_back(i);

TestY1.push\_back(cos(a)/sqrt(1+exp(2\*a)));

TestY2.push\_back(sin(a)/sqrt(1+exp(2\*a)));

for(int i=a; i<t.size(); i++)

{

Y1.push\_back(cos(t[i])/sqrt(1+exp(2\*t[i])));

Y2.push\_back(sin(t[i])/sqrt(1+exp(2\*t[i])));

double k1\_1, k1\_2, k2\_1, k2\_2, k3\_1,k3\_2;

k1\_1 = y1(t[i], TestY1[i], TestY2[i]);

k1\_2 = y2(t[i], TestY1[i], TestY2[i]);

k2\_1 = y1(t[i] + h / 2, TestY1[i] + h\*k1\_1/2, TestY2[i] + h\*k1\_2/2);

k2\_2 = y2(t[i] + h / 2, TestY1[i] + h\*k1\_1/2, TestY2[i] + h\*k1\_2/2);

k3\_1 = y1(t[i] + h , TestY1[i] - h\*k1\_1 + 2\*h\*k2\_1, TestY2[i] - h\*k1\_2 + 2\*h\*k2\_2);

k3\_2 = y2(t[i] + h , TestY1[i] - h\*k1\_1 + 2\*h\*k2\_1, TestY2[i] - h\*k1\_2 + 2\*h\*k2\_2);

TestY1.push\_back(Y1[i] + h\*(k1\_1 + 3\*k3\_1) / 4);

TestY2.push\_back(Y2[i] + h\*(k1\_2 + 3\*k3\_2) / 4);

}

e1=norm(Y1, TestY1);

e2=norm(Y2, TestY2);

H.push\_back(h);

E1.push\_back(e1);

powE1.push\_back(e1/(h\*h\*h));

E2.push\_back(e2);

powE2.push\_back(e2/(h\*h\*h));

h=h/1.5;

t.clear();

TestY1.clear();

TestY2.clear();

Y1.clear();

Y2.clear();

}

double minH=H[0], maxH=H[0];

for(int i=1; i<H.size(); i++)

{

if(H[i]<minH) minH=H[i];

if(H[i]>maxH) maxH=H[i];

}

{

ui->widget\_2->clearGraphs();

ui->widget\_2->addGraph();

//ui->widget\_2->graph()->setData(H,E1);

//dopolnenie

ui->widget\_2->graph()->addData(H,E1);

ui->widget\_2->graph()->setPen(QPen(Qt::red));

// ui->widget\_2->graph()->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

ui->widget\_2->graph()->setLineStyle(QCPGraph::lsNone);

//

ui->widget\_2->graph()->setPen(QPen(Qt::black));

ui->widget\_2->xAxis->setLabel("h");

ui->widget\_2->yAxis->setLabel("е");

// ui->widget\_2->xAxis->setRange(minH,maxH);

ui->widget\_2->xAxis->setRange(0,0.4);

double minY=E1[0], maxY=E1[0];

for(int i=1; i<E2.size(); i++)

{

if(E1[i]<minY) minY=E1[i];

if(E1[i]>maxY) maxY=E1[i];

}

//

//ui->widget\_2->graph(0)->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

ui->widget\_2->yAxis->setRange(minY,maxY);

ui->widget\_2->addGraph();

ui->widget\_2->graph()->setData(H,E2);

ui->widget\_2->graph()->setPen(QPen(Qt::blue));

ui->widget\_2->graph()->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

ui->widget\_2->replot();

}

{

ui->widget\_3->clearGraphs();

ui->widget\_3->addGraph();

ui->widget\_3->graph()->setData(H,powE1);

// ui->widget\_3->xAxis->setLabel("h^3");

// ui->widget\_3->yAxis->setLabel("e/h^3");

ui->widget\_3->xAxis->setLabel("h");

ui->widget\_3->yAxis->setLabel("e/h^3");

//ui->widget\_3->xAxis->setRange(minH/10, maxH/10);

ui->widget\_3->xAxis->setRange(0, 0.04);

double minE=powE1[0], maxE=powE1[0];

for(int i=1; i<powE1.size(); i++)

{

if(powE1[i]<minE) minE=powE1[i];

if(powE1[i]>maxE) maxE=powE1[i];

}

ui->widget\_3->yAxis->setRange(minE/10, maxE/10);

ui->widget\_3->graph()->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

ui->widget\_3->replot();

}

}

QVector <double> X, Y, t\_main, Oo;

double landa = 20000, betta = 4, Oc = 0,B = 13,alpha = 0.03, S = 0.03,c,d,h1, Da;

double MainWindow::**x\_1**(double x, double y,double O,double Da)

{

return (1-x-Da\*x\*exp(O/(1+(O/landa))));

}

double MainWindow::**y\_1**(double x, double y,double O,double Da)

{

return (-y+Da\*x\*exp(O/(1+(O/landa)))-Da\*S\*y\*exp(O/(1+(O/landa))));

}

double MainWindow::**O**(double x, double y,double O,double Da)

{

return (-O+Da\*B\*x\*exp(O/(1+(O/landa)))-betta\*(O-Oc)+Da\*B\*alpha\*S\*y\*exp(O/(1+(O/landa))));

}

void MainWindow::**Ot**()

{

ui->widget\_8->clearGraphs();

ui->widget\_8->addGraph();

ui->widget\_8->graph()->setData(t\_main, Oo);

ui->widget\_8->xAxis->setLabel("t");

ui->widget\_8->yAxis->setLabel("θ");

ui->widget\_8->xAxis->setRange(c,d);

double minY=Oo[0], maxY=Oo[0];

for (int i=0; i<Oo.size(); i++)

{

if(Oo[i]<minY) minY=Oo[i];

if(Oo[i]>maxY) maxY=Oo[i];

}

ui->widget\_8->yAxis->setRange(minY, maxY);

ui->widget\_8->replot();

}

void MainWindow::**yt**()

{

ui->widget\_7->clearGraphs();

ui->widget\_7->addGraph();

ui->widget\_7->graph()->setData(t\_main, Y);

ui->widget\_7->xAxis->setLabel("t");

ui->widget\_7->yAxis->setLabel("Y");

ui->widget\_7->xAxis->setRange(c,d);

double minY=Y[0], maxY=Y[0];

for (int i=0; i<Y.size(); i++)

{

if(Y[i]<minY) minY=Y[i];

if(Y[i]>maxY) maxY=Y[i];

}

ui->widget\_7->yAxis->setRange(minY, maxY);

ui->widget\_7->replot();

}

void MainWindow::**xt**()

{

ui->widget\_5->clearGraphs();

ui->widget\_5->addGraph();

ui->widget\_5->graph()->setData(t\_main, X);

ui->widget\_5->xAxis->setLabel("t");

ui->widget\_5->yAxis->setLabel("X");

ui->widget\_5->xAxis->setRange(c,d);

double minX=X[0], maxX=X[0];

for (int i=0; i<X.size(); i++)

{

if(X[i]<minX) minX=X[i];

if(X[i]>maxX) maxX=X[i];

}

ui->widget\_5->yAxis->setRange(minX, maxX);

ui->widget\_5->replot();

}

void MainWindow::**xO**()

{

ui->widget\_6->clearGraphs();

ui->widget\_6->addGraph();

ui->widget\_6->graph()->setData(X,Oo);

ui->widget\_6->xAxis->setLabel("X");

ui->widget\_6->yAxis->setLabel("θ");

double minX=X[0], maxX=X[0];

for(int i=1; i<X.size(); i++)

{

if(X[i]<minX) minX=X[i];

if(X[i]>maxX) maxX=X[i];

}

double minY=Oo[0], maxY=Oo[0];

for(int i=1; i<Oo.size(); i++)

{

if(Oo[i]<minY) minY=Oo[i];

if(Oo[i]>maxY) maxY=Oo[i];

}

ui->widget\_6->xAxis->setRange(minY/100, maxY/100);

ui->widget\_6->xAxis->setRange(minX, maxX);

ui->widget\_6->graph()->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

//

ui->widget\_6->graph()->setLineStyle(QCPGraph::lsNone);

ui->widget\_6->replot();

}

void MainWindow::**yO**()

{

ui->widget\_4->clearGraphs();

ui->widget\_4->addGraph();

ui->widget\_4->graph()->setData(Y,Oo);

ui->widget\_4->xAxis->setLabel("Y");

ui->widget\_4->yAxis->setLabel("θ");

double minX=Y[0], maxX=Y[0];

for(int i=1; i<Y.size(); i++)

{

if(Y[i]<minX) minX=Y[i];

if(Y[i]>maxX) maxX=Y[i];

}

double minY=Oo[0], maxY=Oo[0];

for(int i=1; i<Oo.size(); i++)

{

if(Oo[i]<minY) minY=Oo[i];

if(Oo[i]>maxY) maxY=Oo[i];

}

ui->widget\_4->xAxis->setRange(minY/100, maxY/100);

ui->widget\_4->xAxis->setRange(minX, maxX);

ui->widget\_4->graph()->setScatterStyle(QCPScatterStyle(QCPScatterStyle::ssCircle, 6));

ui->widget\_4->graph()->setLineStyle(QCPGraph::lsNone);

ui->widget\_4->replot();

}

void MainWindow::**on\_pushButton\_3\_clicked**()

{

t\_main.clear();

X.clear();

Y.clear();

Oo.clear();

QString text\_c=ui->textEdit\_4->toPlainText();

c=text\_c.toDouble();

QString text\_d=ui->textEdit\_5->toPlainText();

d=text\_d.toDouble();

QString text\_h1=ui->textEdit\_6->toPlainText();

h1=text\_h1.toDouble();

QString text\_alpha=ui->textEdit\_7->toPlainText();

Da=text\_alpha.toDouble();

for (double i=c; i<=d; i+=h1)//Пробегаем по всем точкам

{

t\_main.push\_back(i);

}

X.push\_back(0.4);

Y.push\_back(0.5);

Oo.push\_back(1.3);

double k1\_1, k1\_2,k1\_3, k2\_1, k2\_2,k2\_3 ,k3\_1, k3\_2, k3\_3;

for (int i=0; i<t\_main.size(); i++)

{

k1\_1 = x\_1(X[i], Y[i], Oo[i], Da);

k1\_2 = y\_1(X[i], Y[i],Oo[i], Da);

k1\_3 = O(X[i], Y[i], Oo[i], Da);

k2\_1 = x\_1(X[i] + h1\*k1\_1/2, Y[i] + h1\*k1\_2/2,Oo[i]+ h1\*k1\_3/2, Da);

k2\_2 = y\_1(X[i] + h\*k1\_1/2 , Y[i] + h1\*k1\_2/2,Oo[i]+ h1\*k1\_3/2, Da);

k2\_3 = O(X[i]+ h1\*k1\_1/2, Y[i]+ h1\*k1\_2/2, Oo[i]+ h1\*k1\_3/2, Da);

k3\_1 = x\_1(X[i] - h1\*k1\_1+ 2\*h1\*k2\_1, (Y[i] -h1\*k1\_2+ 2\*h1\*k2\_2),Oo[i]-h1\*k1\_3+ 2\*h1\*k2\_3, Da);

k3\_2 = y\_1(X[i] -h1\*k1\_1 + 2\*h1\*k2\_1, (Y[i] -h1\*k1\_2+ 2\*h1\*k2\_2),Oo[i]-h1\*k1\_3+ 2\*h1\*k2\_3, Da);

k3\_3 = O(X[i]-h1\*k1\_1 + 2\*h1\*k2\_1, Y[i]-h1\*k1\_2+ 2\*h1\*k2\_2, Oo[i]-h1\*k1\_3+ 2\*h1\*k2\_3, Da);

X.push\_back(X[i] + h1\*(k1\_1 +4\* k2\_1+k3\_1) / 6);

Y.push\_back(Y[i] + h1\*(k1\_2 + 4\*k2\_2+k3\_2) / 6);

Oo.push\_back(Oo[i] + h1\*(k1\_3 + 4\*k2\_3+k3\_3) / 6);

std::cout<<X[i]<<" "<<Y[i]<<" "<<std::endl;

}

xO();

xt();

yt();

Ot();

yO();

}