Содержание

[1. Постановка задача 2](#_Toc510865415)

[2. Математическая формулировка 3](#_Toc510865416)

[3. Схема алгоритма решения 6](#_Toc510865417)

[4. Результаты вычисления 8](#_Toc510865418)

[5. Анализ результатов и выводы 9](#_Toc510865419)

[6. Приложение 10](#_Toc510865420)

1. Постановка задача

1. Изучить методику обработки экспериментальных данных и получить параметры модели технологического процесса методом наименьших (МНК).
2. Проверить адекватность полученной модели по критерию Фишера.

В данной работе по экспериментальным данным, полученным на промышленных технологических установках ректификации, дегидрирования углеводов, сушки и т.д, необходимо выработать вид однопараметрической линии регрессии, определить параметры и оценить адекватность предложенной регрессионной модели реальному процессу.



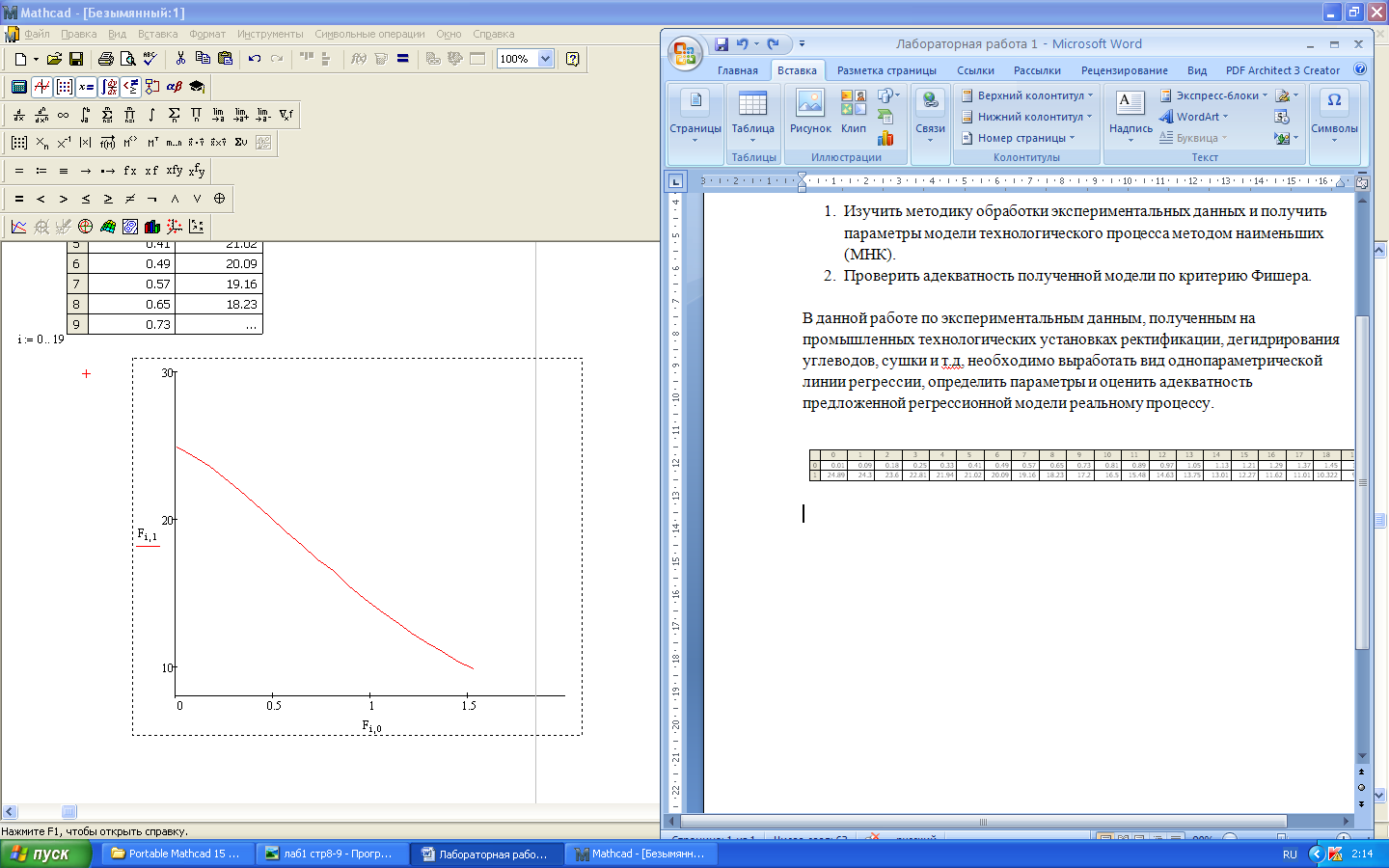


Рис. 1 график

Выбирать уравнение регрессии *y = ax + b*

2. Математическая формулировка

Где N =20 - объем выборки

Исходя из условия существования экстремума функции нескольких переменных необходимым условием минимума Ф(a,b) является выполнение равенств:

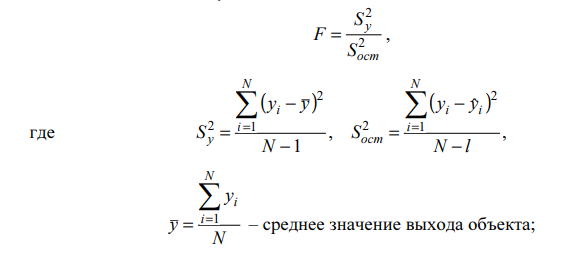
Где -коэффициенты системы:

;

Значения отклика по модели найдем как *yi = axi + b*

Проверка адекватности модели

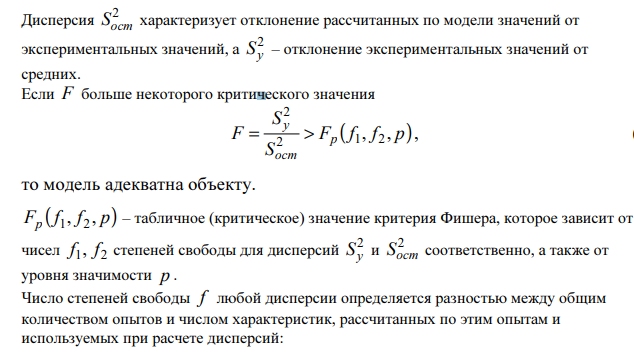
Экспериментальные данные



yˆ – значения отклика, рассчитанные по модели; N = 20 – объем выборки;

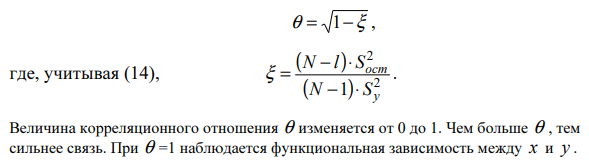
*l* – число связей, наложенных на выборку, равное числу определенных

коэффициентов в уравнении (для линейной регрессии (2) *l* = 2).



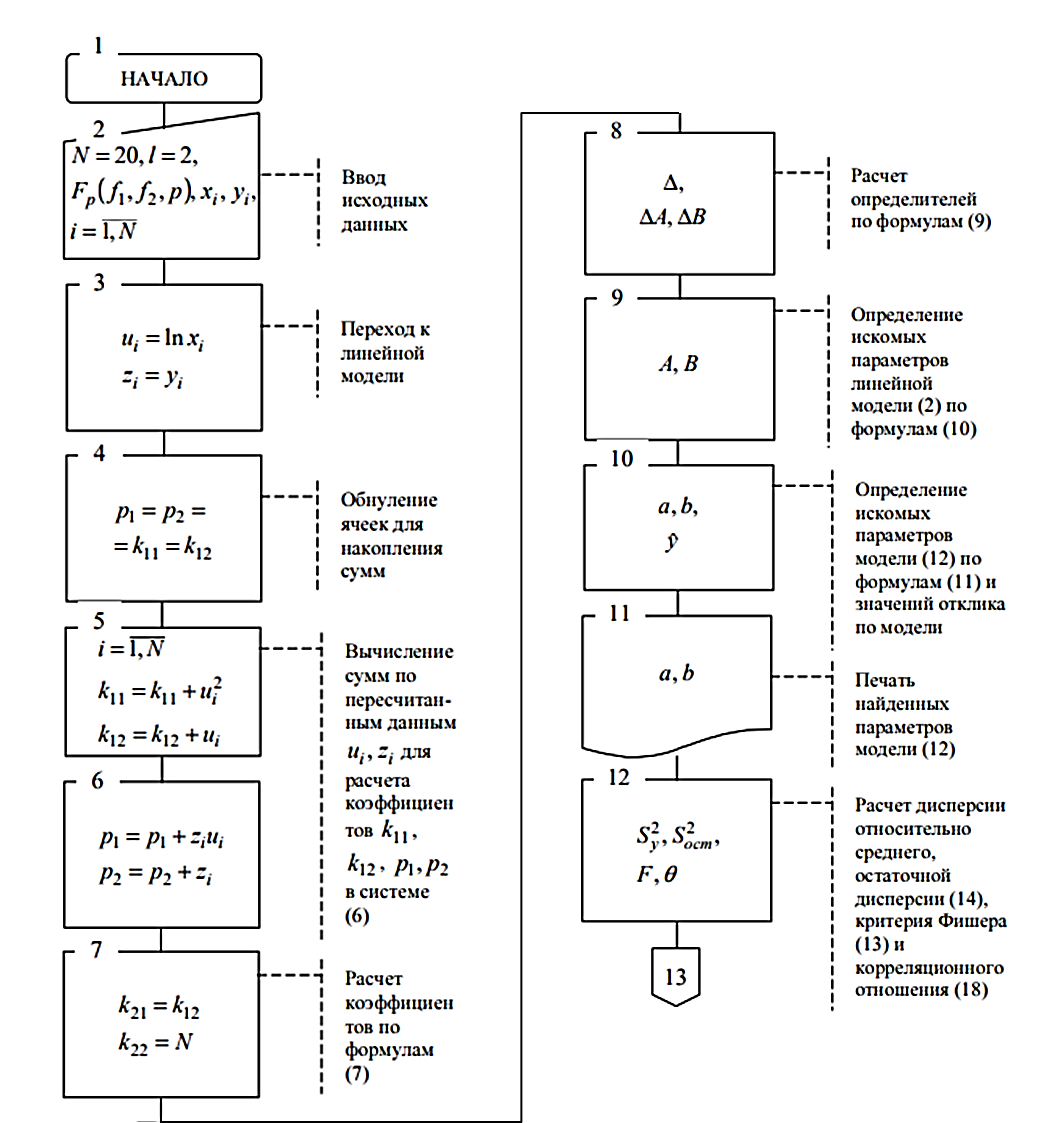
Оценка тесноты связи

оценка тесноты связи характеризуется величиной корреляционного отношения θ



1. Схема алгоритма решения

Представленный алгоритм (рис.1) предназначен для вычисления параметров модели, величины корреляционного отношения и установления адекватности линейной модели при отсутствии параллельных опытов в эксперименте.



Подставим что u = x; z=y

Рисунок 1. Схема алгоритма решения задачи

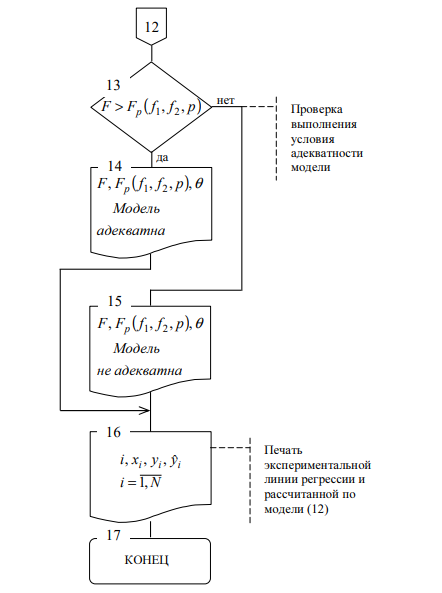


Рисунок 2. Окончание схемы алгоритма решения задачи

На основании схемы алгоритма решения задачи (рис. 1,2) составим программу расчета параметров модели, критерия Фишера, а также коэффициента корреляционного отношения, используя C#.

1. Результаты вычисления

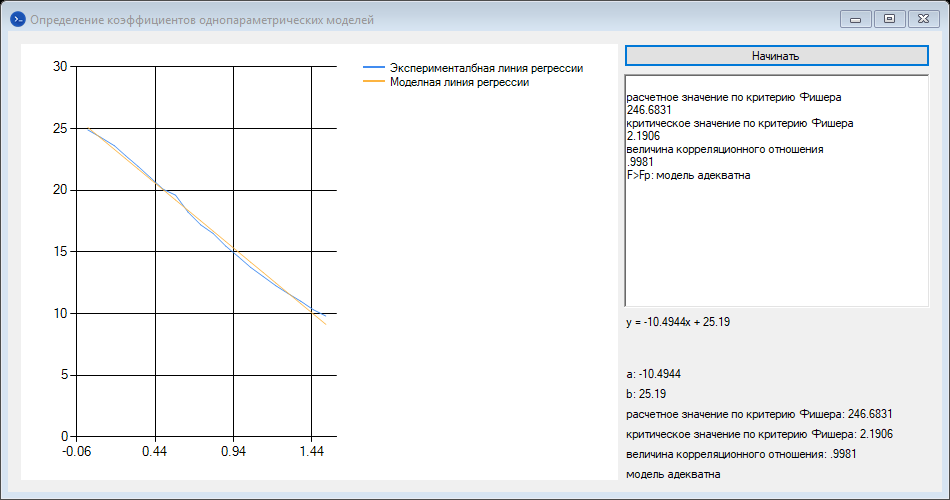


Рисунок 3.

1. Анализ результатов и выводы

Критическое значение по критерию Фишера, Fp = 2.1906

Расчетное значение по критерию Фишера, F = 246.6831

Величина корреляционного отношения = 0.9981

Значение а = -10.4944, b = 25.19

*Модель y =* -10.4944*x +* 25.19

F > Fp то модель адекватна

Вывод:

Изучил методику обработки экспериментальных данных и получил параметры модели технологического процесса методом наименьших (МНК) и проверил адекватность полученной модели по критерию Фишера

1. Приложение

//LinearRegression.cs

class LinearRegression {

private double[] yexp, ymodel;

private double[] xexp;

double Fp;

int N, l;

int error; // 0 everything okay, 1 division by 0, 2 square root of negative

private double A, B;

public LinearRegression(int n, double fp) {

error = 0;

Fp = fp;

N = n;

l = 2; //linear constant

//get values from file or std input

yexp = new double[20] { 24.89, 24.3, 23.6, 22.81, 21.94, 21.02, 20.09, 19.6, 18.23, 17.2, 16.5, 15.48, 14.63, 13.75, 13.01, 12.27, 11.62, 11.01, 10.322, 9.81 };

xexp = new double[20] { 0.01, 0.09, 0.18, 0.25, 0.33, 0.41, 0.49, 0.57, 0.65, 0.73, 0.81, 0.89, 0.97, 1.05, 1.13, 1.21, 1.29, 1.37, 1.45, 1.53 };

ymodel = new double[20];

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

ymodel[i] = 0.0;

}

private double[] getDeterminant() {

double k11, k12, k21, k22, p1, p2;

p1 = p2 = k11 = k12 = 0;

double D, DA, DB;

for (int i = 0; i < N; ++i) {

k11 += xexp[i] \* xexp[i];

k12 += xexp[i];

p1 += xexp[i] \* yexp[i];

p2 += yexp[i];

}

k21 = k12;

k22 = N;

D = k11 \* k22 - k21 \* k12;

DA = p1 \* k22 - p2 \* k12;

DB = k11 \* p2 - k21 \* p1;

return new double[3] { D, DA, DB };

}

public double fillModelData() {

/\*Расчет дисперсии относительно среднего,

остаточной дисперсии, критерия Фишера

и корреляционного отношения

\*/

double[] D = getDeterminant();

if (D[0] == 0) {

error = 1;

return 0;

}

//Определение искомых параметров

A = D[1] / D[0];

B = D[2] / D[0];

double S = 0;

for (int i = 0; i < N; ++i) {

//Определение искомых параметров модели

ymodel[i] = A \* xexp[i] + B;

S += yexp[i];

}

return S / N; /\*расчет среднего\*/

}

public double[] calculateDispersionRelativeToAvg(double y\_) {

/\* расчет дисперсии относительно среднего, остаточной дисперсии\*/

double Sy = 0, So = 0;

for (int i = 0; i < N; ++i) {

Sy += (yexp[i] - y\_) \* (yexp[i] - y\_);

So += (yexp[i] - ymodel[i]) \* (yexp[i] - ymodel[i]);

}

double S2y = Sy / (N - 1);

double S2o = So / (N - l);

return new double[2] { S2y, S2o };

}

public double getA() {

return A;

}

public double getB() {

return B;

}

public double getFp() {

return Fp;

}

public double[] getY() {

return yexp;

}

public double[] getYModel() {

return ymodel;

}

public double[] getX() {

return xexp;

}

public int getL() {

return l;

}

public int getError() {

return error;

}

}

//Form1.cs

public partial class FormMain : Form{

public FormMain() {

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e){

//to get N value from number field

int N = 20;

double A, B;

double[] x, y, ymodel;

int l;

double tablichniFp = 2.1906; // при 5%

LinearRegression driver = new LinearRegression(N,tablichniFp);

double y\_ = driver.fillModelData();

double[] SySo = driver.calculateDispersionRelativeToAvg(y\_);

double Fp = driver.getFp();

A = driver.getA();

B = driver.getB();

l = driver.getL();

this.labela.Text = "a: " + A.ToString("#.####");

this.labelb.Text = "b: " + B.ToString("#.####");

this.labelequation.Text = "y = " + A.ToString("#.####") + "x + " + B.ToString("#.####");

if (driver.getError() == 1) {

this.richTextBoxInfo.AppendText("\n ощибка деление с 0");

return;

}

// расчет критерия Фишера

double F = SySo[0]/ SySo[1];

this.labelFp.Text = "критическое значение по критерию Фишера: " + Fp.ToString("#.####");

this.labelF.Text = "расчетное значение по критерию Фишера: " + F.ToString("#.####");

//расчет корреляционного отношения

double correllationRelationShip = (N - l) \* SySo[1] / ((N - 1) \* SySo[0]);

if (1 - correllationRelationShip < 0) {

this.richTextBoxInfo.AppendText("\n ощибка корень сложный");

return;

}

double tet = Math.Sqrt(1 - correllationRelationShip);

this.labeltet.Text = "величина корреляционного отношения: "+tet.ToString("#.####");

this.richTextBoxInfo.AppendText("\nрасчетное значение по критерию Фишера\n" + F.ToString("#.####") + "\nкритическое значение по критерию Фишера\n" + Fp.ToString("#.####") + "\nвеличина корреляционного отношения\n" + tet.ToString("#.####"));

if (F > Fp) { //Проверка выполнения условия адекватности модели

this.richTextBoxInfo.AppendText("\nF>Fp: модель адекватна");

this.labelconclusion.Text = "модель адекватна";

} else {

this.richTextBoxInfo.AppendText("\nF<=Fp: модель не адекватна");

this.labelconclusion.Text = "модель не адекватна";

}

x = driver.getX();

y = driver.getY();

ymodel = driver.getYModel();

for (int i = 0; i < 20; ++i) {

this.chartShow.Series[1].Points.AddXY(x[i], ymodel[i]);

this.chartShow.Series[0].Points.AddXY(x[i],y[i]);

}

this.chartShow.Show();

}

}

//Program.cs

static class Program{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main(){

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new FormMain());

}

}