Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут ім. Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

ЛАБОРАТОРНА РОБОТА № 3

з дисципліни «Методи планування експерименту»

на тему «ПРОВЕДЕННЯ ТРЬОХФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ.»

Виконав:
студент 2 курсу
групи ІВ-92
Дмитришин А. Д.
Номер у списку групи: 6

ПЕРЕВІРИВ:
ас. Регіда П.Г.

Київ - 2021

Хід роботи

Мета: провести дробовий трьохфакторний експеримент. Скласти матрицю планування, знайти
коефіцієнти рівняння регресії, провести 3 статистичні перевірки.

Завдання:

Завдання на лабораторну роботу

1. Скласти матрицю планування для дробового трьохфакторного експерименту. Провести експеримент в усіх точках факторного простору, повторивши N експериментів, де N – кількість експериментів (рядків матриці планування) в усіх точках факторного простору – знайти значення функції відгуку У. Значення функції відгуку знайти у відповідності з варіантом діапазону, зазначеного далі (випадковим чином).

$y_{max} = 200 + x_{cp\ max}$;

$y_{min} = 200 + x_{cp\ min}$

де $x_{cp\,max} = \frac{x_{1max} + x_{2max} + x_{3max}}{3}$, $x_{cp\,min} = \frac{x_{1min} + x_{2min} + x_{3min}}{3}$

№варианта	X_1		X_2		X_3	
	min	max	min	max	min	max
206	10	40	-15	35	-15	5

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.

3. Провести 3 статистичні перевірки.

4. Написати комп'ютерну програму, яка усе це виконує.

Лістинг

```
package com.company;

import java.util.Random;

public class Lab3 {
    static double avarege(double[] y) {
        double sum = 0;
        for (int i = 0; i < y.length; i++) {
            sum += y[i];
        }
        return sum / y.length;
    }
    static double[] dispersion(double y[][]) {
        double d[] = new double[y.length];
        for (int i = 0; i < y.length; i++) {
            double sum = 0;
            for (int j = 0; j < y[i].length; j++) {
                sum += Math.pow(y[i][j] - avarege(y[i]), 2);
            }
            d[i] = sum / y[i].length;
        }
        return d;
    }

    static double determinant4(double a[][]) {
        double[][] k1 = {{a[1][1], a[1][2], a[1][3]}, {a[2][1], a[2][2], a[2][3]},
{a[3][1], a[3][2], a[3][3]}};
        double[][] k2 = {{a[1][0], a[1][2], a[1][3]}, {a[2][0], a[2][2], a[2][3]},
{a[3][0], a[3][2], a[3][3]}};
        double[][] k3 = {{a[1][1], a[1][0], a[1][3]}, {a[2][1], a[2][0], a[2][3]},
{a[3][1], a[3][0], a[3][3]}};
        double[][] k4 = {{a[1][1], a[1][2], a[1][0]}, {a[2][1], a[2][2], a[2][0]},
{a[3][1], a[3][2], a[3][0]}};

        return a[0][0] * determinant(k1) - a[0][1] * determinant(k2) - a[0][2] *
determinant(k3) - a[0][3] * determinant(k4);
    }

    static double determinant(double a[][]) {
        return a[0][0] * a[1][1] * a[2][2] + a[0][1] * a[1][2] * a[2][0] + a[1][0] *
a[0][2] * a[2][1] - a[0][2] * a[1][1] * a[2][0] - a[0][1] *
                a[1][0] * a[2][2] - a[0][0] * a[1][2] * a[2][1];
    }
    static double max(double a[]) {
        double max=a[0];
        for (int i = 1; i < a.length; i++) {
            if (max<a[i]) max=a[i];
        }
        return max;
    }
    public static void main(String[] args) {
        double x[][] = {{1, 1, 1, 1},
                {-1, -1, 1, 1},
                {-1, 1, -1, 1},
                {-1, 1, 1, -1}};
        double xmin[] = {10, -15, -15};
        double xmax[] = {40, 35, 5};
        double xmax_avr = avarege(xmax);
        double xmin_avr = avarege(xmin);
        double ymax = 200 + xmax_avr;
        double ymin = 200 + xmin_avr;
        int y1[] = new int[4];
```

```
        int y2[] = new int[4];
        int y3[] = new int[4];
        Random random = new Random();
        for (int i = 0; i < y1.length; i++) {
            y1[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
            y2[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
            y3[i] = random.nextInt((int) ymax - (int) ymin + 1) + (int) ymin;
        }
        double y[][] = {{y1[0], y2[0], y3[0]},
                {y1[1], y2[1], y3[1]},
                {y1[2], y2[2], y3[2]},
                {y1[3], y2[3], y3[3]}
        };
        double xm[][] = {{10, 10, 40, 40},
                {-15, 35, -15, 35},
                {-15, 5, 5, -15}};
        double yavr[] = {avarege(y[0]), avarege(y[1]), avarege(y[2]),
avarege(y[3])};
        double mx1 = avarege(xm[0]);
        double mx2 = avarege(xm[1]);
        double mx3 = avarege(xm[2]);
        double my = avarege(yavr);
        double a1 = (xm[0][0] * yavr[0] + xm[0][1] * yavr[1] + xm[0][2] * yavr[2] +
xm[0][3] * yavr[3]) / 4;
        double a2 = (xm[1][0] * yavr[0] + xm[1][1] * yavr[1] + xm[1][2] * yavr[2] +
xm[1][3] * yavr[3]) / 4;
        double a3 = (xm[2][0] * yavr[0] + xm[2][1] * yavr[1] + xm[2][2] * yavr[2] +
xm[2][3] * yavr[3]) / 4;
        double a11 = (xm[0][0] * xm[0][0] + xm[0][1] * xm[0][1] + xm[0][2] *
xm[0][2] + xm[0][3] * xm[0][3]) / 4;
        double a22 = (xm[1][0] * xm[1][0] + xm[1][1] * xm[1][1] + xm[1][2] *
xm[1][2] + xm[1][3] * xm[1][3]) / 4;
        double a33 = (xm[2][0] * xm[2][0] + xm[2][1] * xm[2][1] + xm[2][2] *
xm[2][2] + xm[2][3] * xm[2][3]) / 4;
        double a12 = (xm[0][0] * xm[1][0] + xm[0][1] * xm[1][1] + xm[0][2] *
xm[1][2] + xm[0][3] * xm[1][3]) / 4;
        double a21 = a12;
        double a13 = (xm[0][0] * xm[2][0] + xm[0][1] * xm[2][1] + xm[0][2] *
xm[2][2] + xm[0][3] * xm[2][3]) / 4;
        double a31 = a13;
        double a23 = (xm[1][0] * xm[2][0] + xm[1][1] * xm[2][1] + xm[1][2] *
xm[2][2] + xm[1][3] * xm[2][3]) / 4;
        double a32 = a23;
        double b0m[][] = {{1, mx1, mx2, mx3},
                          {mx1, a11, a12, a13},
                          {mx2, a12, a22, a23},
                          {mx3, a13, a23, a33}};
        double b1m[][] = {{my, mx1, mx2, mx3},
                          {a1, a11, a12, a13},
                          {a2, a12, a22, a23},
                          {a3, a13, a23, a33}};
        double b2m[][] = {{1, my, mx2, mx3},
                          {mx1, a1, a12, a13},
                          {mx2, a2, a22, a23},
                          {mx3, a3, a23, a33}};
        double b3m[][] = {{1, mx1, my, mx3},
                          {mx1, a11, a1, a13},
                          {mx2, a12, a2, a23},
                          {mx3, a13, a3, a33}};
        double b4m[][] = {{1, mx1, mx2, my},
                          {mx1, a11, a12, a1},
                          {mx2, a12, a22, a2},
                          {mx3, a13, a23, a3}};
        double b0=determinant4(b1m)/determinant4(b0m);
```

```
        double b1=determinant4(b2m)/determinant4(b0m);
        double b2=determinant4(b3m)/determinant4(b0m);
        double b3=determinant4(b4m)/determinant4(b0m);
        double ypr1=b0+b1*xm[0][0]+b2*xm[1][0]+b3*xm[2][0];
        double ypr2=b0+b1*xm[0][1]+b2*xm[1][1]+b3*xm[2][1];
        double ypr3=b0+b1*xm[0][2]+b2*xm[1][2]+b3*xm[2][2];
        double ypr4=b0+b1*xm[0][3]+b2*xm[1][3]+b3*xm[2][3];
        double S[]=dispersion(y);
        double Gp=max(S)/(S[0]+S[1]+S[2]+S[3]);
        double Gt=0.7679;
        if (Gp>Gt) {
            System.out.println("Неоднорідна дисперсія");
            System.exit(1337);
        }
        double Sbs=Math.sqrt((S[0]+S[1]+S[2]+S[3])/(3*4*4));
        double
beta0=(x[0][0]*yavr[0]+x[0][1]*yavr[1]+x[0][2]*yavr[2]+x[0][3]*yavr[3])/4;
        double
beta1=(x[1][0]*yavr[0]+x[1][1]*yavr[1]+x[1][2]*yavr[2]+x[1][3]*yavr[3])/4;
        double
beta2=(x[2][0]*yavr[0]+x[2][1]*yavr[1]+x[2][2]*yavr[2]+x[2][3]*yavr[3])/4;
        double
beta3=(x[3][0]*yavr[0]+x[3][1]*yavr[1]+x[3][2]*yavr[2]+x[3][3]*yavr[3])/4;
        double bmat[]={b0,b1,b2,b3};
        double
t[]={Math.abs(beta0)/Sbs,Math.abs(beta1)/Sbs,Math.abs(beta2)/Sbs,Math.abs(beta3)/Sbs
};
        //f3 = (m-1)*N= 2*4=8  tтабл = 2.306
        String k ="";
        int n = 0;
        for (int i = 0; i < t.length; i++) {
            if(t[i]<2.306){
                k+="b"+i+" ";
                n+=1;
                bmat[i]=0;
            }
        }

        double ys1=bmat[0]+bmat[1]*xm[0][0]+bmat[2]*xm[1][0]+bmat[3]*xm[2][0];
        double ys2=bmat[0]+bmat[1]*xm[0][1]+bmat[2]*xm[1][1]+bmat[3]*xm[2][1];
        double ys3=bmat[0]+bmat[1]*xm[0][2]+bmat[2]*xm[1][2]+bmat[3]*xm[2][2];
        double ys4=bmat[0]+bmat[1]*xm[0][3]+bmat[2]*xm[1][3]+bmat[3]*xm[2][3];
        int f4=(4-(4-n));
        //f3 = (m-1)*N= 2*4=8
        double [] Fisher={5.3, 4.5, 4.1, 3.8};
        double Sad=3/(4-(4-n))*(Math.pow(yavr[0]-ys1,2)+Math.pow(yavr[1]-
ys2,2)+Math.pow(yavr[2]-ys3,2)+Math.pow(yavr[3]-ys4,2));
        double Fp=Sad/Sbs;
        System.out.println("Нормована матриця планування");
        for (int i = 0; i < x.length; i++) {
            for (int j = 0; j < x[i].length; j++) {
                System.out.print(x[i][j]+" ");
            }
            System.out.println();
        }
        System.out.println("Середні значення: "+yavr[0]+" "+yavr[1]+" "+yavr[2]+"
"+yavr[3]);
        System.out.println("З нормованими коефіцієнтами: "+ypr1+" "+ypr2+" "+ypr3+"
"+ypr4);
        System.out.println(k+"коефіцієнти рівняння регресії приймаємо незначними при
рівні значимості 0.05, тобто вони виключаються з рівняння.");
        System.out.println("Отримані за критерієм Стьюдента: "+ys1+" "+ys2+" "+ys3+"
"+ys4);
        if (Fp<Fisher[4-n-1]){
```

```
            System.out.println("Fp="+Fp+" Ft="+Fisher[4-n-1]+" рівняння регресії адекватно оригіналу при рівні значимості 0.05");
        }
        else System.out.println("Fp="+Fp+" Ft="+Fisher[4-n-1]+" рівняння регресії неадекватно оригіналу при рівні значимості 0.05");
    }


}
```

Відповіді на контрольні запитання

1.Що називається дробовим факторним експериментом?
Дробовий факторний експеримент – це частина ПФЕ, який мінімізує число дослідів, за рахунок тієї інформації, яка не дуже істотна для побудови лінійної моделі.

2. Для чого потрібно розрахункове значення Кохрена?
Розрахункове значення Кохрена використовують для перевірки однорідності дисперсій.

3. Для чого перевіряється критерій Стьюдента?
За допомогою критерію Стьюдента перевіряється значущість коефіцієнтів рівняння регресії.

4. Чим визначається критерій Фішера і як його застосовувати?
Критерій Фішера використовують при перевірці отриманого рівняння регресії досліджуваному об'єкту.

Результати роботи програми

```
Нормована матриця планування
1.0 1.0 1.0 1.0
-1.0 -1.0 1.0 1.0
-1.0 1.0 -1.0 1.0
-1.0 1.0 1.0 -1.0
Середні значення: 214.0 208.0 205.66666666666666 210.66666666666666
З нормованими коефіцієнтами: 213.99999999999977 208.00000000000009 205.6666666666666 210.66666666666663
b1 b2 b3 коефіцієнти рівняння регресії приймаємо незначними при рівні значимості 0.05, тобто вони виключаються з рівняння.
Отримані за критерієм Стьюдента: 210.66944444444437 210.66944444444437 210.66944444444437 210.66944444444437
Fp=15.320949104180865 Ft=5.3 рівняння регресії неадекватно оригіналу при рівні значимості 0.05
```