

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

ВИКОНАВ:  
студент II курсу ФІОТ  
групи ІВ-92  
Подкур Антон Олександрович  
Варіант: 217  
ПЕРЕВІРИВ:  
Регіда П. Г.

## Хід роботи

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

### Завдання:

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 Y. Значення функції відгуку знайти у відповідності з варіантом діапазону, зазначеного далі (випадковим чином)
2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.
3. Провести 3 статистичні перевірки.
4. Написати комп'ютерну програму, яка усе це виконує.

№ <sub>варіанта</sub>	X <sub>1</sub>		X <sub>2</sub>		X <sub>3</sub>	
	min	max	min	max	min	max
217	20	70	5	40	20	45

## Лістинг програми

```
using System;
using ConsoleTables;

namespace Lab3
{
    public class Lab3
    {
        public static void Main(string[] args)
        {
            Experiment experiment = new Experiment(20, 70, 5, 40, 20, 45);
            experiment.run();
        }
    }

    class Experiment
    {
        ConsoleTable table;
        private int X1min, X1max, X2min, X2max, X3min, X3max;

        public Experiment(int x1min, int x1max, int x2min, int x2max, int x3min,
int x3max)
        {
            X1min = x1min;
            X1max = x1max;
            X2min = x2min;
            X2max = x2max;
            X3min = x3min;
            X3max = x3max;
        }

        public void run()
        {

            double Ymin = (X1min + X2min + X3min) / 3.0 + 200;
            double Ymax = (X1max + X2max + X3max) / 3.0 + 200;

            int[,] normalizeMatrix = new int[4, 8];

            normalizeMatrix[0, 0] = X1min;
            normalizeMatrix[0, 1] = X2min;
            normalizeMatrix[0, 2] = X3min;
            normalizeMatrix[1, 0] = X1min;
            normalizeMatrix[1, 1] = X2max;
            normalizeMatrix[1, 2] = X3max;
            normalizeMatrix[2, 0] = X1max;
            normalizeMatrix[2, 1] = X2min;
            normalizeMatrix[2, 2] = X3max;
            normalizeMatrix[3, 0] = X1max;
```

```

        normalizeMatrix[3, 1] = X2max;
        normalizeMatrix[3, 2] = X3min;

        Random rnd = new Random();
        for (int i = 0; i < 4; i++)
        {
            for (int j = 3; j < 8; j++)
            {
                normalizeMatrix[i, j] = (int)(Ymin + (Ymax - Ymin) * rnd.Next
Double());
            }
        }

        Console.WriteLine();
        table = new ConsoleTable(" ", " ", " ", " ", " ", "Naturalized matrix", " ", " ",
, " ", " ");
        table.AddRow("N", "X1max", "X2max", "X3max", "Y1", "Y2", "Y3", "Y4", "Y5");
        for (int i = 0; i < 4; i++)
        {
            int[] row = new int[9];
            row[0] = i+1;
            // row.Add(i+1);
            for (int j = 0; j < 8; j++)
            {
                row[j+1] = normalizeMatrix[i, j];
            }
            table.AddRow(row[0], row[1], row[2], row[3], row[4], row[5], row[6], row
[7], row[8]);
        }
        table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Forma
t.Alternative);

        Console.WriteLine();

        double[] averageY = new double[4];
        double mY = 0;
        for (int i = 0; i < 4; i++)
        {
            for (int j = 3; j < 8; j++)
            {
                averageY[i] += normalizeMatrix[i, j];
            }
            averageY[i] /= 5.0;
            mY += averageY[i];
        }
        mY /= 4.0;
        double mX1 = (normalizeMatrix[0, 0] + normalizeMatrix[1, 0] + normali
zeMatrix[2, 0] + normalizeMatrix[3, 0]) / 4.0;
        double mX2 = (normalizeMatrix[0, 1] + normalizeMatrix[1, 1] + normali
zeMatrix[2, 1] + normalizeMatrix[3, 1]) / 4.0;

```

```

        double mX3 = (normalizeMatrix[0, 2] + normalizeMatrix[1, 2] + normalizeMatrix[2, 2] + normalizeMatrix[3, 2]) / 4.0;
        double a1 = (normalizeMatrix[0, 0] * averageY[0] + normalizeMatrix[1, 0] * averageY[1] + normalizeMatrix[2, 0] * averageY[2] + normalizeMatrix[3, 0] * averageY[3]) / 4.0;
        double a2 = (normalizeMatrix[0, 1] * averageY[0] + normalizeMatrix[1, 1] * averageY[1] + normalizeMatrix[2, 1] * averageY[2] + normalizeMatrix[3, 1] * averageY[3]) / 4.0;
        double a3 = (normalizeMatrix[0, 2] * averageY[0] + normalizeMatrix[1, 2] * averageY[1] + normalizeMatrix[2, 2] * averageY[2] + normalizeMatrix[3, 2] * averageY[3]) / 4.0;
        double a11 = (normalizeMatrix[0, 0] * normalizeMatrix[0, 0] + normalizeMatrix[1, 0] * normalizeMatrix[1, 0]) / 4.0;
        a11 += (normalizeMatrix[2, 0] * normalizeMatrix[2, 0] + normalizeMatrix[3, 0] * normalizeMatrix[3, 0]) / 4.0;
        double a22 = (normalizeMatrix[0, 1] * normalizeMatrix[0, 1] + normalizeMatrix[1, 1] * normalizeMatrix[1, 1]) / 4.0;
        a22 += (normalizeMatrix[2, 1] * normalizeMatrix[2, 1] + normalizeMatrix[3, 1] * normalizeMatrix[3, 1]) / 4.0;
        double a33 = (normalizeMatrix[0, 2] * normalizeMatrix[0, 2] + normalizeMatrix[1, 2] * normalizeMatrix[1, 2]) / 4.0;
        a33 += (normalizeMatrix[2, 2] * normalizeMatrix[2, 2] + normalizeMatrix[3, 2] * normalizeMatrix[3, 2]) / 4.0;
        double a12 = (normalizeMatrix[0, 0] * normalizeMatrix[0, 1] + normalizeMatrix[1, 0] * normalizeMatrix[1, 1]) / 4.0;
        a12 += (normalizeMatrix[2, 0] * normalizeMatrix[2, 1] + normalizeMatrix[3, 0] * normalizeMatrix[3, 1]) / 4.0;
        double a13 = (normalizeMatrix[0, 0] * normalizeMatrix[0, 2] + normalizeMatrix[1, 0] * normalizeMatrix[1, 2]) / 4.0;
        a13 += (normalizeMatrix[2, 0] * normalizeMatrix[2, 2] + normalizeMatrix[3, 0] * normalizeMatrix[3, 2]) / 4.0;
        double a23 = (normalizeMatrix[0, 1] * normalizeMatrix[0, 2] + normalizeMatrix[1, 1] * normalizeMatrix[1, 2]) / 4.0;
        a23 += (normalizeMatrix[2, 1] * normalizeMatrix[2, 2] + normalizeMatrix[3, 1] * normalizeMatrix[3, 2]) / 4.0;
        double det = determinantOf4
            (1, mX1, mX2, mX3,
             mX1, a11, a12, a13,
             mX2, a12, a22, a23,
             mX3, a13, a23, a33);
        double det1 = determinantOf4
            (mY, mX1, mX2, mX3,
             a1, a11, a12, a13,
             a2, a12, a22, a23,
             a3, a13, a23, a33);
        double det2 = determinantOf4
            (1, mY, mX2, mX3,
             mX1, a1, a12, a13,
             mX2, a2, a22, a23,
             mX3, a3, a23, a33);

```

```

double det3 = determinantOf4
    (1, mX1, mY, mX3,
     mX1, a11, a1, a13,
     mX2, a12, a2, a23,
     mX3, a13, a3, a33);
double det4 = determinantOf4
    (1, mX1, mX2, mY,
     mX1, a11, a12, a1,
     mX2, a12, a22, a2,
     mX3, a13, a23, a3);

double b0 = det1 / det;
double b0_Round = Math.Round(b0 * 100) / 100.0;
double b1 = det2 / det;
double b1_Round = Math.Round(b1 * 100) / 100.0;
double b2 = det3 / det;
double b2_Round = Math.Round(b2 * 100) / 100.0;
double b3 = det4 / det;
double b3_Round = Math.Round(b3 * 100) / 100.0;
Console.WriteLine();

table = new ConsoleTable("Regression equation");
table.AddRow("Y = " + b0_Round + " + " + b1_Round + "*X1max + " + b2_
Round + "*X2max + " + b3_Round + "*X3max");
table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.
Alternative);
Console.WriteLine();

table = new ConsoleTable("", "New values", "", "");
table.AddRow("Y1", "Y2", "Y3", "Y4");
table.AddRow(Math.Round(100 * (b0 + b1 * X1min + b2 * X2min + b3 * X3
min)) / 100.0, Math.Round(100 * (b0 + b1 * X1min + b2 * X2max + b3 * X3max)) / 100
.0, Math.Round(100 * (b0 + b1 * X1max + b2 * X2min + b3 * X3max)) / 100.0, Math.Rou
nd(100 * (b0 + b1 * X1max + b2 * X2max + b3 * X3min)) / 100.0);
table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.
Alternative);
Console.WriteLine();

int[,] matrix = normalizeMatrix;
matrix[0, 0] = matrix[1, 0] = matrix[2, 1] = matrix[3, 2] = matrix[0,
1] = matrix[0, 2] = -1;
matrix[1, 1] = matrix[1, 2] = matrix[2, 0] = matrix[2, 2] = matrix[3,
0] = matrix[3, 1] = 1;

table = new ConsoleTable(" ", " ", " ", " ", " ", "Normalized matrix", " ", " ",
" ", " ");
table.AddRow("N", "X1max", "X2max", "X3max", "Y1", "Y2", "Y3", "Y4", "Y5");
for (int i = 0; i < 4; i++)
{
    int[] row = new int[9];

```

```

        row[0] = i+1;
        // row.Add(i+1);
        for (int j = 0; j < 8; j++)
        {
            row[j+1] = normalizeMatrix[i, j];
        }
        table.AddRow(row[0],row[1],row[2],row[3],row[4],row[5],row[6],row
[7],row[8]);
    }
    table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);

    Console.WriteLine();

    double[] Sigma = new double[4];
    double[] Sigma_Round = Sigma;
    for (int i = 0; i < 4; i++)
    {
        for (int j = 3; j < 8; j++)
            Sigma[i] += (averageY[i] - matrix[i, j]) * (averageY[i] - mat
rix[i, j]);
        Sigma[i] /= 5.0;
        Sigma_Round[i] = Math.Round(Sigma[i] * 100) / 100.0;
    }

    table = new ConsoleTable("", "Average Ys", "", "");
    table.AddRow("Y1", "Y2", "Y3", "Y4");
    table.AddRow(averageY[0], averageY[1], averageY[2], averageY[3]);
    table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);
    Console.WriteLine();

    table = new ConsoleTable("", "Dispersions", "", "");
    table.AddRow("D(Y1)", "D(Y2)", "D(Y3)", "D(Y4)");
    table.AddRow(Sigma_Round[0], Sigma_Round[1], Sigma_Round[2], Sigma_Round
[3]);
    table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);
    Console.WriteLine();

    Sigma_Round = Sigma;
    Array.Sort(Sigma_Round);
    double Gp = Sigma_Round[3] / (Sigma[0] + Sigma[1] + Sigma[2] + Sigma[
3]);

    Console.WriteLine("Cochren's test:  " + Math.Round(Gp * 10000) / 1000
0.0);
    if (Gp <= 0.6287)

```

```

        Console.WriteLine("\nThe variance is homogeneous with a probability of 95%.");
    }
    else
    {
        Console.WriteLine("\nThe variance is not homogeneous.");

        double Sb = (Sigma[0] + Sigma[1] + Sigma[2] + Sigma[3]) / 4.0;
        double Sbs = Math.Sqrt(Sb / 20.0);
        double beta0 = (averageY[0] + averageY[1] + averageY[2] + averageY[3]) / 4.0;
        double beta1 = (averageY[0] * matrix[0, 0] + averageY[1] * matrix[1, 0] + averageY[2] * matrix[2, 0] + averageY[3] * matrix[3, 0]) / 4.0;
        double beta2 = (averageY[0] * matrix[0, 1] + averageY[1] * matrix[1, 1] + averageY[2] * matrix[2, 1] + averageY[3] * matrix[3, 1]) / 4.0;
        double beta3 = (averageY[0] * matrix[0, 2] + averageY[1] * matrix[1, 2] + averageY[2] * matrix[2, 2] + averageY[3] * matrix[3, 2]) / 4.0;
        double[] t = new double[4];
        double[] t_Round = t;
        t[0] = Math.Abs(beta0) / Sbs;
        t_Round[0] = Math.Round(t[0] * 100000) / 100000.0;
        t[1] = Math.Abs(beta1) / Sbs;
        t_Round[1] = Math.Round(t[1] * 100000) / 100000.0;
        t[2] = Math.Abs(beta2) / Sbs;
        t_Round[2] = Math.Round(t[2] * 100000) / 100000.0;
        t[3] = Math.Abs(beta3) / Sbs;
        t_Round[3] = Math.Round(t[3] * 100000) / 100000.0;
        Console.WriteLine();

        table = new ConsoleTable("", "Student's test", "", "");
        table.AddRow("t1", "t2", "t3", "t4");
        table.AddRow(t_Round[0], t_Round[1], t_Round[2], t_Round[3]);
        table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);
        Console.WriteLine();

        double[] b = { b0, b1, b2, b3 };
        for (int i = 0; i < 4; i++)
        {
            if (t[i] < 2.12)
                b[i] = 0;
        }

        table = new ConsoleTable("Regression equation");
        String str = "Y = ";
        int f4 = 4;
        if (b[0] != 0)
        {
            str += (b0_Round);
            f4--;
        }
        if (b[1] != 0)

```



```

        {
            str+=(" + " + b1_Round + "*X1max");
            f4--;
        }
        if (b[2] != 0)
        {
            str+=(" + " + b2_Round + "*X2max");
            f4--;
        }
        if (b[3] != 0)
        {
            str+=(" + " + b3_Round + "*X3max");
            f4--;
        }
        table.AddRow(str);
        table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);
        Console.WriteLine();

        double[] Yj = new double[4];
        Yj[0] = b[0] + b[1] * X1min + b[2] * X2min + b[3] * X3min;
        Yj[1] = b[0] + b[1] * X1min + b[2] * X2max + b[3] * X3max;
        Yj[2] = b[0] + b[1] * X1max + b[2] * X2min + b[3] * X3max;
        Yj[3] = b[0] + b[1] * X1max + b[2] * X2max + b[3] * X3min;

        table = new ConsoleTable("", "New values", "", "");
        table.AddRow("Y1", "Y2", "Y3", "Y4");
        table.AddRow(Math.Round(100 * (Yj[0])) / 100.0, Math.Round(100 * (Yj[1])) / 100.0, Math.Round(100 * (Yj[2])) / 100.0, Math.Round(100 * (Yj[3])) / 100.0);
        table.Configure(o => o.NumberAlignment = Alignment.Right).Write(Format.Alternative);
        Console.WriteLine();

        double[] fisher = { 4.5, 3.6, 3.2, 3.0 };
        double Sad = 0;
        for (int i = 0; i < 4; i++)
            Sad += (averageY[i] - Yj[i]) * (averageY[i] - Yj[i]);
        Sad *= 5.0 / f4;
        double F = Sad / Sb;
        Console.WriteLine("\nFisher's test: " + Math.Round(F * 100) / 100.0);
;

        if (F < fisher[f4])
            Console.WriteLine("\nThe regression equation is adequate at a significance level of 5%.");
        else
            Console.WriteLine("\nThe regression equation is inadequate at a significance level of 5%.");
        Console.WriteLine();
    }
}

```

```

public double determinantOf3(
    double a11, double a12, double a13,
    double a21, double a22, double a23,
    double a31, double a32, double a33)
{
    return a11 * a22 * a33 - a13 * a22 * a31 +
        a12 * a23 * a31 - a12 * a21 * a33 +
        a13 * a21 * a32 - a11 * a23 * a32;
}

public double determinantOf4(
    double a11, double a12, double a13, double a14,
    double a21, double a22, double a23, double a24,
    double a31, double a32, double a33, double a34,
    double a41, double a42, double a43, double a44)
{
    return a11 * determinantOf3(a22, a23, a24, a32, a33, a34, a42, a43, a
44) -
        a12 * determinantOf3(a21, a23, a24, a31, a33, a34, a41, a43,
a44) -
        a13 * determinantOf3(a22, a21, a24, a32, a31, a34, a42, a41,
a44) -
        a14 * determinantOf3(a22, a23, a21, a32, a33, a31, a42, a43,
a41);
}
}
}

```

## Результат роботи програми

				Naturalized matrix					
N	X1max	X2max	X3max	Y1	Y2	Y3	Y4	Y5	
1	20	5	20	237	247	237	237	220	
2	20	40	45	250	250	228	243	217	
3	70	5	45	223	249	251	229	215	
4	70	40	20	228	246	240	219	236	

Regression equation
$Y = 235.99 + -0.06 * X1max + 0.03 * X2max + 0.03 * X3max$

	New values			
Y1	Y2	Y3	Y4	
235.6	237.6	233.4	233.8	

				Normalized matrix					
N	X1max	X2max	X3max	Y1	Y2	Y3	Y4	Y5	
1	-1	-1	-1	237	247	237	237	220	
2	-1	1	1	250	250	228	243	217	
3	1	-1	1	223	249	251	229	215	
4	1	1	-1	228	246	240	219	236	

Average Ys			
Y1	Y2	Y3	Y4
235.6	237.6	233.4	233.8

Dispersions			
D(Y1)	D(Y2)	D(Y3)	D(Y4)
75.84	170.64	203.84	88.96

Cochren's test: 0.378

The variance is homogeneous with a probability of 95%.

Student's test			
t1	t2	t3	t4
90.55042	0.57774	0.23109	0.15406

Regression equation
$Y = 235.99$

New values			
Y1	Y2	Y3	Y4
235.99	235.99	235.99	235.99

Fisher's test: 0.18

The regression equation is adequate at a significance level of 5%.

## Відповіді на контрольні запитання

### 1. Що називається дробовим факторним експериментом?

У деяких випадках немає необхідності проводити повний факторний експеримент (ПФЕ). Якщо буде використовуватися лінійна регресія, то можливо зменшити кількість рядків матриці ПФЕ до кількості коефіцієнтів регресійної моделі. Кількість дослідів слід скоротити, використовуючи для планування так звані регулярні дробові репліки від повного факторного експерименту, що містять відповідну кількість дослідів і зберігають основні властивості матриці планування – це означає дробовий факторний експеримент (ДФЕ).

### 2. Для чого потрібно розрахункове значення Кохрена?

Статистична перевірка за критерієм Кохрена використовується для перевірки гіпотези про однорідність дисперсії з довірчою ймовірністю  $p$ . Якщо експериментальне значення  $G < G_{кр}$ , яке обирається з таблиці, то гіпотеза підтверджується, якщо ні, то відповідно не підтверджується.

### 3. Для чого перевіряється критерій Стюдента?

Критерій Стюдента використовується для перевірки значимості коефіцієнта рівняння регресії. Якщо з'ясувалось, що будь-який коефіцієнт рівняння регресії не значимий, то відповідний  $b_i = 0$  і відповідний член рівняння регресії треба викреслити. Іноді ця статистична перевірка має назву «нуль-гіпотеза». Якщо експериментальне значення  $t > t_{кр}$ , нуль-гіпотеза не підтверджується і даний коефіцієнт значимий, інакше нуль-гіпотеза підтверджується і даний коефіцієнт рівняння регресії не значимий.

### 4. Чим визначається критерій Фішера і як його застосовувати?

Критерій Фішера застосовується для перевірки адекватності моделі (рівняння регресії) оригіналу (експериментальним даним).

Обчислюється експериментальне значення  $F$ , яке порівнюється з  $F_{кр}$ , взятим з таблиці залежно від кількості значимих коефіцієнтів та ступенів вільності. Якщо  $F < F_{кр}$ , то модель адекватна оригіналу, інакше – ні.

**Висновок:** під час виконання лабораторної роботи було проведено дробовий трьохфакторний експеримент, складено матрицю планування, знайдено коефіцієнти рівняння регресії, проведено 3 статистичні перевірки.