

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 № 4**

з дисципліни «МНД» на тему  
«Проведення трьохфакторного експерименту при використанні рівняння  
регресії з урахуванням ефекту взаємодії»

ВИКОНАВ:  
студент II курсу ФІОТ  
групи ІВ-91  
Красновський О. В.  
Залікова - 9116

ПЕРЕВІРИВ:  
ас. Регіда П. Г.

Київ – 2021

**Мета:** провести повний трьохфакторний експеримент. Знайти рівняння регресії адекватне об'єкту.

**Завдання на лабораторну роботу:**

1. Скласти матрицю планування для повного трьохфакторного експерименту.
2. Провести експеримент, повторивши N раз досліди у всіх точках факторного простору і знайти значення відгуку Y. Знайти значення Y шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту. Варіанти вибираються за номером в списку в журналі викладача.

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

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

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

3. Знайти коефіцієнти рівняння регресії і записати його.
4. Провести 3 статистичні перевірки – за критеріями Кохрена, Стьюдента, Фішера.
5. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
6. Написати комп'ютерну програму, яка усе це моделює.

**Варіант завдання:**

№варіанта	X <sub>1</sub>		X <sub>2</sub>		X <sub>3</sub>	
	min	max	min	max	min	max
115	-25	75	25	65	25	40

## **Лістинг програми:**

```
import java.util.Arrays;
```

```
import java.util.Random;
```

```
public class Main {
```

```
    static double square(double x) {
```

```
        return x * x;
```

```
    }
```

```
    static double average(int[] arr) {
```

```
        double sum = 0;
```

```
        for (int i = 0; i < arr.length; ++i) {
```

```
            sum += arr[i];
```

```
        }
```

```
        return sum / arr.length;
```

```
    }
```

```
    static double average(double[] arr) {
```

```
        double sum = 0;
```

```
        for (int i = 0; i < arr.length; ++i) {
```

```
            sum += arr[i];
```

```
        }
```

```
        return sum / arr.length;
```

```
    }
```

```
    static double max(double[] arr) {
```

```
        double max = arr[0];
```

```
        for (int i = 1; i < arr.length; i++) {
```

```

        if (arr[i] > max) {
            max = arr[i];
        }
    }
    return max;
}

```

```

static double dispersion(int[] arr) {
    double avg = average(arr);
    double dispersion = 0;
    for (int i : arr) {
        dispersion += ((i - avg) * (i - avg)) / arr.length;
    }
    return dispersion;
}

```

```

static boolean cochrane(double[] dispersionList) {
    System.out.println("Dispersions:");
    System.out.println(Arrays.toString(dispersionList));
    double GPDenom = 0;
    for (double el : dispersionList) {
        GPDenom += el;
    }
    double GP = max(dispersionList) / GPDenom;
    // Візьмемо 0.05 рівень значимості
    // F1 = 2, F2 = 8
    // GT=0.5157
    final double GT = 0.5157;
    System.out.println("GP = " + GP);
    System.out.println("GT = " + GT);
}

```

```

    return GP < GT;
}

```

```

static double[] studentsCriterion(int m, int N, double[] dispersionList, double[] bList) {

    double sbSquare = average(dispersionList) / (N * m);

    double sb = Math.sqrt(sbSquare);

    double[] tList = new double[N];

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

        tList[i] = Math.abs(bList[i]) / sb;

    }

    System.out.println("beta: " + Arrays.toString(bList));

    System.out.println("t: " + Arrays.toString(tList));

    // Число ступенів свободи F3 = (m-1) * N = 16

    double tCriterion = 2.12;

    double[] student = Arrays.copyOf(bList, bList.length);

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

        if (tList[i] < tCriterion) {

            student[i] = 0;

        }

    }

    return student;

}

```

```

static boolean fishersCriterion(int m, int N, double[] student, double[] avgList, double[] regList, double[]
dispersionList) {

    int ctr = 0;

    for (double d : student) {

        if (d != 0) {

            ctr++;

        }

    }

}

```

```

        int f4 = N - ctr;

        double s2Adequacy = 0;
        for (int i = 0; i < N; i++) {
            s2Adequacy += square(regList[i] - avgList[i]);
        }

        s2Adequacy *= (double)m / f4;

        double s2Reproducibility = average(dispersionList);

        double fp = s2Adequacy / s2Reproducibility;

        double ft = 4.5;

        return fp <= ft;
    }

    static void insertMatrix(int[][] arr, int min, int max) {
        Random random = new Random();

        for (int i = 0; i < arr.length; i++) {
            for (int j = 0; j < arr[0].length; j++) {
                arr[i][j] = random.ints(min, max).findFirst().getAsInt();
            }
        }
    }

    public static void main(String[] args) {
        int m, N = 8;

        double[] yAverage;

        double b0;

        double b1;

        double b2;

        double b3;

        double b12;

        double b13;
    }

```

```

double b23;

double b123;

int[][] planMatrix;

double[] dispersionList;

do {

    m = 3;

    int[] xMin = {-25, 25, 25};

    int[] xMax = {75, 65, 40};

    double yMin = 200 + average(xMin);

    double yMax = 200 + average(xMax);

    System.out.println("The minimum value of the response function: " + yMin);

    System.out.println("The maximum value of the response function: " + yMax);

    int[][] yMatrix = new int[N][m];

    insertMatrix(yMatrix, (int)Math.floor(yMin), (int)Math.floor(yMax));

    System.out.println("Y matrix:");

    for (int i = 0; i < yMatrix.length; i++) {

        System.out.println(Arrays.toString(yMatrix[i]));

    }

    yAverage = new double[N];

    for (int i = 0; i < yAverage.length; i++) {

        yAverage[i] = average(yMatrix[i]);

    }

    System.out.println("Y matrix averages:");

    System.out.println(Arrays.toString(yAverage));

    int[][] xNormalized = {{-1, -1, -1},

        {-1, -1, 1},

        {-1, 1, -1},

        {-1, 1, 1},

        {1, -1, -1},

        {1, -1, 1},

```

```

        {1, 1, -1},
        {1, 1, 1}};

// b0
double sum = 0;
for (double avg : yAverage) {
    sum += avg;
}
b0 = sum / N;

// b1
sum = 0;
for (int i = 0; i < N; i++) {
    sum += yAverage[i] * xNormalized[i][0];
}
b1 = sum / N;

// b2
sum = 0;
for (int i = 0; i < N; i++) {
    sum += yAverage[i] * xNormalized[i][1];
}
b2 = sum / N;

// b3
sum = 0;
for (int i = 0; i < N; i++) {
    sum += yAverage[i] * xNormalized[i][2];
}
b3 = sum / N;

// b12
sum = 0;
for (int i = 0; i < N; i++) {
    sum += yAverage[i] * xNormalized[i][0] * xNormalized[i][1];
}

```



```

    }

    b12 = sum / N;

    // b13

    sum = 0;

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

        sum += yAverage[i] * xNormalized[i][0] * xNormalized[i][2];

    }

    b13 = sum / N;

    // b23

    sum = 0;

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

        sum += yAverage[i] * xNormalized[i][1] * xNormalized[i][2];

    }

    b23 = sum / N;

    // b123

    sum = 0;

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

        sum += yAverage[i] * xNormalized[i][0] * xNormalized[i][1] *
xNormalized[i][2];

    }

    b123 = sum / N;

    planMatrix= new int[][]{ { xMin[0], xMin[1], xMin[2], xMin[0] * xMin[1], xMin[0] *
xMin[2], xMin[1] * xMin[2], xMin[0] * xMin[1] * xMin[2]},

        { xMin[0], xMin[1], xMax[2], xMin[0] * xMin[1], xMin[0] * xMax[2],
xMin[1] * xMax[2], xMin[0] * xMin[1] * xMax[2]},

        { xMin[0], xMax[1], xMin[2], xMin[0] * xMax[1], xMin[0] * xMin[2],
xMax[1] * xMin[2], xMin[0] * xMax[1] * xMin[2]},

        { xMin[0], xMax[1], xMax[2], xMin[0] * xMax[1], xMin[0] * xMax[2],
xMax[1] * xMax[2], xMin[0] * xMax[1] * xMax[2]},

        { xMax[0], xMin[1], xMin[2], xMax[0] * xMin[1], xMax[0] * xMin[2],
xMin[1] * xMin[2], xMax[0] * xMin[1] * xMin[2]},

        { xMax[0], xMin[1], xMax[2], xMax[0] * xMin[1], xMax[0] *
xMax[2], xMin[1] * xMax[2], xMax[0] * xMin[1] * xMax[2]},

```

```

        {xMax[0], xMax[1], xMin[2], xMax[0] * xMax[1], xMax[0] *
xMin[2], xMax[1] * xMin[2], xMax[0] * xMax[1] * xMin[2]},

        {xMax[0], xMax[1], xMax[2], xMax[0] * xMax[1], xMax[0] *
xMax[2], xMax[1] * xMax[2], xMax[0] * xMax[1] * xMax[2]}};

    System.out.println("Planning matrix:");

    for (int i = 0; i < planMatrix.length; i++) {

        System.out.println(Arrays.toString(planMatrix[i]));

    }

    double[] yResult = new double[N];

    for (int i = 0; i < yResult.length; i++) {

        yResult[i] = b0 + b1 * planMatrix[i][0] + b2 * planMatrix[i][1] + b3 *
planMatrix[i][2] +

        b12 * planMatrix[i][3] + b13 * planMatrix[i][4] + b23 *
planMatrix[i][5] + b123 * planMatrix[i][6];

    }

    dispersionList = new double[N];

    for (int i = 0; i < yMatrix.length; i++) {

        dispersionList[i] = dispersion(yMatrix[i]);

    }

    System.out.println("~~~~~");

    System.out.println("Cochren test:");

    if (cochrane(dispersionList)) {

        System.out.println("The dispersion is homogeneous");

    } else {

        System.out.println("The dispersion is inhomogeneous");

        m++;

    }

    } while (!cochrane(dispersionList));

    System.out.println("~~~~~");

    System.out.println("Checking the significance of the coefficients according to Student's criterion:");

    double[] student = studentsCriterion(m, N, dispersionList, new double[]{b0, b1, b2, b3, b12, b13,
b23, b123});

```

```

double[] regression = new double[N];

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

    regression[i] = student[0] + student[1] * planMatrix[i][0] + student[2] * planMatrix[i][1]
+
    student[3] * planMatrix[i][2] + student[4] * planMatrix[i][3] +
student[5] * planMatrix[i][4] +
    student[6] * planMatrix[i][5] + student[7] * planMatrix[i][6];

}

System.out.println("Values of regression equations:");
System.out.println(Arrays.toString(regression));
System.out.println("~~~~~");
System.out.println("Fisher's test:");

if (fishersCriterion(m, N, student, yAverage, regression, dispersionList)) {

    System.out.println("The regression equation is adequate to the original at a significance
level of 0.05");

} else {

    System.err.println("The regression equation is inadequate to the original at a significance
level of 0.05");

}

System.out.println("Equation:");

System.out.println(b0 + b1 + " * x1 + " + b2 + " * x2 + " + b3 + " * x3 + " + b12 + " * x1x2 + " +
b13 + " * x1x3 + " + b23 + " * x2x3 + " + b123 + " * x1x2x3");

}

}

```

## Результат роботи програми:

```
Y matrix:
[216, 211, 209]
[250, 210, 215]
[247, 256, 217]
[208, 219, 233]
[238, 239, 219]
[210, 248, 251]
[221, 215, 257]
[237, 229, 230]
Y matrix averages:
[212.0, 225.0, 240.0, 220.0, 232.0, 236.33333333333334, 231.0, 232.0]
Planning matrix:
[-25, 25, 25, -625, -625, 625, -15625]
[-25, 25, 40, -625, -1000, 1000, -25000]
[-25, 65, 25, -1625, -625, 1625, -40625]
[-25, 65, 40, -1625, -1000, 2600, -65000]
[75, 25, 25, 1875, 1875, 625, 46875]
[75, 25, 40, 1875, 3000, 1000, 75000]
[75, 65, 25, 4875, 1875, 1625, 121875]
[75, 65, 40, 4875, 3000, 2600, 195000]
~~~~~
Cochren test:
Dispersions:
[206.8888888888889, 216.2222222222223, 117.55555555555554, 156.2222222222223, 12.666666666666668, 89.55555555555556, 27.555555555555554, 80.2222222222223]
GP = 0.23842195540308747
GT = 0.5157
The dispersion is homogeneous
Dispersions:
[206.8888888888889, 216.2222222222223, 117.55555555555554, 156.2222222222223, 12.666666666666668, 89.55555555555556, 27.555555555555554, 80.2222222222223]
GP = 0.23842195540308747
GT = 0.5157
~~~~~
Checking the significance of the coefficients according to Student's criterion:
Values of regression equations:
[233.04166666666666, 233.04166666666666, 233.04166666666666, 233.04166666666666, 233.04166666666666, 233.04166666666666, 233.04166666666666, 233.04166666666666]
~~~~~
Fisher's test:
The regression equation is adequate to the original at a significance level of 0.05
Equation:
235.3333333333331 * x1 + -3.791666666666643 * x2 + -2.791666666666668 * x3 + -1.708333333333215 * x1x2 + -3.541666666666668 * x1x3 + 0.0416666666666785 * x2x3 + -2.
20833333333332 * x1x2x3
Press any key to continue . . .
```

## Висновки:

Виконуючи дану лабораторну роботу, я склав матрицю планування для повного трьохфакторного експерименту, провів експеримент, повторивши N раз досліди у всіх точках факторного простору і знайшов значення відгуку Y, шляхом моделювання випадкових чисел у певному діапазоні відповідно варіанту 115.

Знайшов коефіцієнти рівняння регресії і записав його. Провів статистичні перевірки за критеріями Кохрена, Стюдента, Фішера. Зробив висновки по адекватності регресії та значимості окремих коефіцієнтів і записав скореговане рівняння регресії. Написав комп'ютерну програму, яка все це моделює. Написав .bat файл який запускає програму та перезапускає її в разі неадекватності моделі.

Результати роботи програми наведені вище підтверджують правильність обраних рішень. Кінцеву мету роботи досягнуто.