Національний технічний університет України

«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

Методи оптимізації та планування експерименту

Лабораторна робота №2

“ПРОВЕДЕННЯ ДВОФАКТОРНОГО ЕКСПЕРИМЕНТУ З ВИКОРИСТАННЯМ ЛІНІЙНОГО РІВНЯННЯ РЕГРЕСІЇ”

Виконав:

студент групи ІВ-83

Поліщук Д.Я.

Перевірив:

ас. Регіда П.Г.

Київ

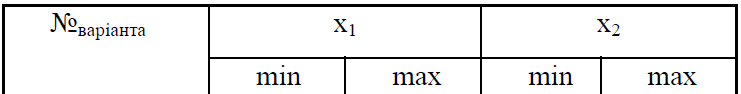
2020 р.

**Мета:** провести двофакторний експеримент, перевірити однорідність дисперсії за критерієм Романовського, отримати коефіцієнти рівняння регресії, провести

натуралізацію рівняння регресії.

Номер у списку: 18.

Варіант завдання: 318.





1. Лістинг програми:

import java.util.Random;

/\*\*

\* @author Danya Polishchuk

\* @information Student IV-83, number 18

\* Task 318:

\*/

public class MOPE\_lab2 {

static Random random = new Random();

static int myNumber = 18;

static int m = 5;

static int y\_max = (30 - myNumber) \* 10;

static int y\_min = (20 - myNumber) \* 10;

static float[][] matrix\_X = new float[][]{{-1, -1}, {1, -1}, {-1, 1}};

static float[][] matrix\_Y = new float[3][m];

static float[] matrix\_AverageOfY = new float[3];

static float[] dispersia = new float[3];

static float sigma = (float) Math.sqrt((float) (2 \* (2 \* m - 2)) / (float) (m \* (m - 4)));

static float[] F\_uv = new float[3];

static float[] O\_uv = new float[3];

static float[] R\_uv = new float[3];

static float[] mx = new float[2];

static float my = 0;

static float a1 = 0;

static float a2 = 0;

static float a3 = 0;

static float a11 = 0;

static float a22 = 0;

static float[][] matrix\_b;

static float[][] matrix\_b1;

static float[][] matrix\_b2;

static float[][] matrix\_b3;

static float b0 = 0;

static float b1 = 0;

static float b2 = 0;

static float x1min = 20;

static float x1max = 70;

static float x2min = -15;

static float x2max = 45;

static float[][] matrixForNaturalization = new float[][]{{x1min, x1max}, {x2min, x2max}};

static float delta\_x1 = Math.abs(x1max - x1min) / 2;

static float delta\_x2 = Math.abs(x2max - x2min) / 2;

static float x10 = (x1max + x1min) / 2;

static float x20 = (x2max + x2min) / 2;

static float a\_0 = 0;

static float a\_1 = 0;

static float a\_2 = 0;

public static void main(String[] args) {

showY();

fillMatrixOfY();

averageOfY();

fillDispersia();

criterionRomanovskiy();

}

private static void showY() {

System.out.println("Мій номер в списку: " + myNumber + "\n\ny\_max = (30 - 18)\*10 = " + y\_max

+ "\ny\_min = (20 - 18)\*10 = " + y\_min + "\n");

}

private static void fillMatrixOfY() {

for (int i = 0; i < matrix\_Y.length; i++) {

for (int j = 0; j < matrix\_Y[i].length; j++) {

matrix\_Y[i][j] = (float) (random.nextInt(y\_max + 1));

if (matrix\_Y[i][j] < (float) y\_min) {

matrix\_Y[i][j] += (float) y\_min;

}

}

}

iteration(matrix\_Y);

}

private static void iteration(float[][] array) {

if (array == matrix\_Y) {

System.out.println("Значення функції відгуку в інтервалі від [" + y\_min + "-" + y\_max + "]:");

}

for (float[] i : array) {

for (float j : i) {

System.out.print(j + " ");

}

System.out.println();

}

}

private static void iteration(float[] array) {

if (array == matrix\_AverageOfY) {

System.out.println("\nСереднє значення функції відгуку в кожній точці плану:");

} else if (array == dispersia) {

System.out.println("\nДисперсії для кожної точки планування:");

}

for (float i : array) {

System.out.print(i + "\n");

}

}

private static void averageOfY() {

for (int i = 0; i < matrix\_Y.length; i++) {

matrix\_AverageOfY[i] = averageValueOfY(matrix\_Y[i]);

}

iteration(matrix\_AverageOfY);

}

private static float averageValueOfY(float[] array) {

float sum = 0;

for (float i : array) {

sum += i;

}

return sum / array.length;

}

private static void fillDispersia() {

for (int i = 0; i < matrix\_Y.length; i++) {

float sum = 0;

for (int j = 0; j < matrix\_Y[i].length; j++) {

sum += Math.pow((matrix\_Y[i][j] - matrix\_AverageOfY[i]), 2);

}

dispersia[i] = sum / matrix\_Y[i].length;

}

iteration(dispersia);

System.out.println("\nОсновне відхилення:\nsigma = " + sigma);

}

private static void criterionRomanovskiy() {

F\_uv[0] = calculateF(dispersia[0], dispersia[1]);

F\_uv[1] = calculateF(dispersia[0], dispersia[2]);

F\_uv[2] = calculateF(dispersia[1], dispersia[2]);

for (int i = 0; i < O\_uv.length; i++) {

O\_uv[i] = ((float) (m - 2) / (float) m) \* F\_uv[i];

R\_uv[i] = Math.abs(O\_uv[i] - 1) / sigma;

}

boolean bool = false;

System.out.println();

for (int i = 0; i < R\_uv.length; i++) {

System.out.print("R\_uv" + i + " = " + R\_uv[i]);

if (R\_uv[i] < 2) {

System.out.print(" < " + " R\_кр = 2");

} else {

System.out.print(" > " + " R\_кр = 2");

bool = true;

}

System.out.println();

}

if (bool) {

System.out.println("\nОтже, дисперсія не однорідна...");

} else {

System.out.println("\nОтже, дисперсія однорідна.");

calculateNormCoefficients();

naturalizationCoefficients();

}

}

private static float calculateF(float dispersia1, float dispersia2) {

if (dispersia1 >= dispersia2) {

return dispersia1 / dispersia2;

} else {

return dispersia2 / dispersia1;

}

}

private static void calculateNormCoefficients() {

for (int i = 0; i < matrix\_X[0].length; i++) {

mx[i] = calculateAverage(matrix\_X[0][i], matrix\_X[1][i], matrix\_X[2][i]);

}

my = calculateAverage(matrix\_AverageOfY[0], matrix\_AverageOfY[1], matrix\_AverageOfY[2]);

a1 = calculateAverage((float) Math.pow(matrix\_X[0][0], 2), (float) Math.pow(matrix\_X[1][0], 2), (float) Math.pow(matrix\_X[2][0], 2));

a2 = calculateAverage(matrix\_X[0][0] \* matrix\_X[0][1], matrix\_X[1][0] \* matrix\_X[1][1], matrix\_X[2][0] \* matrix\_X[2][1]);

a3 = calculateAverage((float) Math.pow(matrix\_X[0][1], 2), (float) Math.pow(matrix\_X[1][1], 2), (float) Math.pow(matrix\_X[2][1], 2));

a11 = calculateAverage(matrix\_X[0][0] \* matrix\_AverageOfY[0], matrix\_X[1][0] \* matrix\_AverageOfY[1], matrix\_X[2][0] \* matrix\_AverageOfY[2]);

a22 = calculateAverage(matrix\_X[0][1] \* matrix\_AverageOfY[0], matrix\_X[1][1] \* matrix\_AverageOfY[1], matrix\_X[2][1] \* matrix\_AverageOfY[2]);

matrix\_b = new float[][]{{1, mx[0], mx[1]}, {mx[0], a1, a2}, {mx[1], a2, a3}};

matrix\_b1 = new float[][]{{my, mx[0], mx[1]}, {a11, a1, a2}, {a22, a2, a3}};

matrix\_b2 = new float[][]{{1, my, mx[1]}, {mx[0], a11, a2}, {mx[1], a22, a3}};

matrix\_b3 = new float[][]{{1, mx[0], my}, {mx[0], a1, a11}, {mx[1], a2, a22}};

b0 = discriminant(matrix\_b1) / discriminant(matrix\_b);

b1 = discriminant(matrix\_b2) / discriminant(matrix\_b);

b2 = discriminant(matrix\_b3) / discriminant(matrix\_b);

System.out.println();

System.out.println("Розрахунок нормованих коефіцієнтів рівняння регресії:");

for (int i = 0; i < matrix\_X.length; i++) {

for (int j = 0; j < matrix\_X[i].length - 1; j++) {

System.out.println("y = b0 + b1\*x1 + b2\*x2 = " + b0 + " + " + b1 + "\*" + matrix\_X[i][j] + " + " +

b2 + "\*" + matrix\_X[i][j + 1] + " = " + (b0 + b1 \* matrix\_X[i][j] + b2 \* matrix\_X[i][j + 1]));

}

}

}

private static void naturalizationCoefficients() {

a\_0 = b0 - b1 \* (x10 / delta\_x1) - b2 \* (x20 / delta\_x2);

a\_1 = b1 / delta\_x1;

a\_2 = b2 / delta\_x2;

System.out.println("\nЗапишемо натуралізоване рівняння регресії:");

System.out.println("y = a0 + a1\*x1 + a2\*x2 = " + a\_0 + " + " + a\_1 + "\*" + x1min + " + " +

a\_2 + "\*" + x2min + " = " + (a\_0 + a\_1 \* x1min + a\_2 \* x2min));

System.out.println("y = a0 + a1\*x1 + a2\*x2 = " + a\_0 + " + " + a\_1 + "\*" + x1max + " + " +

a\_2 + "\*" + x2min + " = " + (a\_0 + a\_1 \* x1max + a\_2 \* x2min));

System.out.println("y = a0 + a1\*x1 + a2\*x2 = " + a\_0 + " + " + a\_1 + "\*" + x1min + " + " +

a\_2 + "\*" + x2max + " = " + (a\_0 + a\_1 \* x1min + a\_2 \* x2max));

}

private static float calculateAverage(float... arg) {

float sum = 0;

for (float i : arg) {

sum += i;

}

return sum / arg.length;

}

private static float discriminant(float[][] matrix) {

return (matrix[0][0] \* matrix[1][1] \* matrix[2][2]) + (matrix[1][0] \* matrix[2][1] \* matrix[0][2])

+ (matrix[0][1] \* matrix[1][2] \* matrix[2][0]) - ((matrix[2][0] \* matrix[1][1] \* matrix[0][2])

+ (matrix[0][0] \* matrix[2][1] \* matrix[1][2]) + (matrix[1][0] \* matrix[0][1] \* matrix[2][2]));

}

}

1. Відповіді на контрольні запитання:

1) Що таке регресійні поліноми і де вони застосовуються?

2) Визначення однорідності дисперсії.

3) Що називається повним факторним експериментом?