Міністерство освіти і науки України

Національний технічний університет України «КПІ» імені Ігоря Сікорського

Кафедра обчислювальної техніки ФІОТ

ЗВІТ

з лабораторної роботи №3

з навчальної дисципліни «МЕТОДИ ОБЧИСЛЕННЯ ТА ПЛАНУВАННЯ ЕКСПЕРИМЕНТУ»

Тема:

« проведення трьохфакторного експерименту з використанням лінійного рівняння регресії»

Виконав:

студент 2-го курсу ФІОТ

групи ІО-91

Денисенко М. О.

Варіант - 7

Перевірив:

Регіда П. Г.

Київ 2021

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 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 $x_{cp\ max} = (1/3)(x_{1max}+x_{2max}+x_{3max})$, $x_{cp\ min} = (1/3)(x_{1min}+x_{2min}+x_{3min})$

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

3. Провести 3 статистичні перевірки.

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

107	-5	15	-15	35	15	30

Код програми:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;


public class MopeLab3 {
    public static double determinant(double[][] m) {
        return
                m[0][3] * m[1][2] * m[2][1] * m[3][0] - m[0][2] * m[1][3] * m[2][1] * m[3][0] -
                        m[0][3] * m[1][1] * m[2][2] * m[3][0] + m[0][1] * m[1][3] * m[2][2] * m[3][0] +
                        m[0][2] * m[1][1] * m[2][3] * m[3][0] - m[0][1] * m[1][2] * m[2][3] * m[3][0] -
                        m[0][3] * m[1][2] * m[2][0] * m[3][1] + m[0][2] * m[1][3] * m[2][0] * m[3][1] +
                        m[0][3] * m[1][0] * m[2][2] * m[3][1] - m[0][0] * m[1][3] * m[2][2] * m[3][1] -
                        m[0][2] * m[1][0] * m[2][3] * m[3][1] + m[0][0] * m[1][2] * m[2][3] * m[3][1] +
                        m[0][3] * m[1][1] * m[2][0] * m[3][2] - m[0][1] * m[1][3] * m[2][0] * m[3][2] -
                        m[0][3] * m[1][0] * m[2][1] * m[3][2] + m[0][0] * m[1][3] * m[2][1] * m[3][2] +
                        m[0][1] * m[1][0] * m[2][3] * m[3][2] - m[0][0] * m[1][1] * m[2][3] * m[3][2] -
                        m[0][2] * m[1][1] * m[2][0] * m[3][3] + m[0][1] * m[1][2] * m[2][0] * m[3][3] +
                        m[0][2] * m[1][0] * m[2][1] * m[3][3] - m[0][0] * m[1][2] * m[2][1] * m[3][3] -
                        m[0][1] * m[1][0] * m[2][2] * m[3][3] + m[0][0] * m[1][1] * m[2][2] * m[3][3];
    }

    public static void main(String[] args) {

        int MinX1 = -5;
        int MaxX1 = 15;
        int MinX2 = -15;
```

```
int MaxX2 = 35;
int MinX3 = 15;
int MaxX3 = 30;

double MaxY = 221.667;
double MinY = 198.333;

int[][] x = {
        {1, -1, -1, -1},
        {1, -1, 1, 1},
        {1, 1, -1, 1},
        {1, 1, 1, -1}
};

int[][] xArr = {
        {MinX1, MinX2, MinX3},
        {MinX1, MaxX2, MaxX3},
        {MaxX1, MinX2, MaxX3},
        {MaxX1, MaxX2, MinX3}
};

double[][] aKoef = new double[3][3];

double[] mx = new double[3];
double sum;
double my;
double[] a = new double[3];
double[] yAverage = new double[4];
double[] bArr = new double[4];
double[] dispersionArr = new double[4];
List<String> symbols = new ArrayList<>();
int f1 = 0;
int f2 = 0;
int m;
double q = 0;
boolean work = false;

Scanner st = new Scanner(System.in);
System.out.println("Задайте значення m: ");
try {
    m = st.nextInt();
    if (m > 0) {
        symbols.add("\u2219");
        symbols.add("\u2260");
        for (int i = 0; i <= m; i++) {
            symbols.add("" + i);
        }
        work = true;

    } else {
        System.out.println("Потрібно задати додатнє значення...");
    }
} catch (Exception e) {
    System.out.println("Потрібно задати ціле значення...");
    m = 0;
}

while (work) {

    List<double[]> y = new ArrayList<>();
    System.out.printf("Лінійне рівняння регресії для нормованих значень x має вигляд : " +
                    "y = b%s + b%s%sx%s + b%s%sx%s + b%s%sx%s",
            symbols.get(2), symbols.get(3), symbols.get(0), symbols.get(3),
```

```
        symbols.get(4), symbols.get(0), symbols.get(4),
        symbols.get(5), symbols.get(0), symbols.get(5));
System.out.println();

System.out.println("Нормована матриця планування експерименту : ");
System.out.print("X0\tX1\tX2\tX3\t");
for (int i = 0; i < m; i++) {
   System.out.print("Y" + (i + 1) + "\t\t\t\t");
}
System.out.println();
for (int i = 0; i < 4; i++) {
   double[] yTemp = new double[m];
   for (int j = 0; j < 4; j++) {
      if (x[i][j] == 1){
         System.out.print(" " + x[i][j] + "\t");
      }
      else {
         System.out.print(x[i][j] + "\t");
      }

   }
   for (int j = 0; j < m; j++) {
      yTemp[j] = (Math.random() * (MaxY - MinY)) + MinY;
      System.out.print((float)yTemp[j] + "\t\t");
   }
   System.out.println();
   y.add(yTemp);
}

System.out.println("Матриця планування експерименту : ");
System.out.print("X1\tX2\tX3\t");
for (int i = 0; i < m; i++) {
   System.out.print("Y" + (i + 1) + "\t\t\t\t");
}
System.out.println();
for (int i = 0; i < 4; i++) {
   double[] yTemp = new double[m];
   for (int j = 0; j < 3; j++) {
      if (xArr[i][j] > 0){
         System.out.print(" " + xArr[i][j] + "\t");
      }
      else {
         System.out.print(xArr[i][j] + "\t");
      }
   }
   yTemp = y.get(i);
   for (int j = 0; j < m; j++) {
      System.out.print((float)yTemp[j] + "\t\t");
   }
   System.out.println();
}

for (int i = 0; i < 4; i++) {
   sum = 0;
   double[] yTemp = new double[m];
   yTemp = y.get(i);
   for (int j = 0; j < m; j++) {
      sum += yTemp[j];
   }
   yAverage[i] = sum / m;
}

for (int i = 0; i < 3; i++) {
```

```
            sum = 0;
            for (int j = 0; j < 4; j++) {
                sum += xArr[j][i];
            }
            mx[i] = sum / 4;
        }
        sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += yAverage[i];
        }
        my = sum / 4;

        for (int i = 0; i < 3; i++) {
            sum = 0;
            for (int j = 0; j < 4; j++) {
                sum += xArr[j][i] * yAverage[j];
            }
            a[i] = sum / 4;
        }

        for (int i = 0; i < 3; i++) {
            sum = 0;
            for (int j = 0; j < 4; j++) {
                sum += Math.pow(xArr[j][i], 2);
            }
            aKoef[i][i] = sum / 4;
        }

        aKoef[0][1] = aKoef[1][0] = (xArr[0][0] * xArr[0][1] + xArr[1][0] * xArr[1][1] + xArr[2][0] * xArr[2][1] +
xArr[3][0] * xArr[3][1]) / 4.;
        aKoef[0][2] = aKoef[2][0] = (xArr[0][0] * xArr[0][2] + xArr[1][0] * xArr[1][2] + xArr[2][0] * xArr[2][2] +
xArr[3][0] * xArr[3][2]) / 4.;
        aKoef[1][2] = aKoef[2][1] = (xArr[0][1] * xArr[0][2] + xArr[1][1] * xArr[1][2] + xArr[2][1] * xArr[2][2] +
xArr[3][1] * xArr[3][2]) / 4.;

        double[][] matrixTemp1 = {
                {my, mx[0], mx[1], mx[2]},
                {a[0], aKoef[0][0], aKoef[0][1], aKoef[0][2]},
                {a[1], aKoef[0][1], aKoef[1][1], aKoef[2][1]},
                {a[2], aKoef[0][2], aKoef[1][2], aKoef[2][2]}
        };

        double[][] matrixTemp2 = {
                {1, mx[0], mx[1], mx[2]},
                {mx[0], aKoef[0][0], aKoef[0][1], aKoef[0][2]},
                {mx[1], aKoef[0][1], aKoef[1][1], aKoef[2][1]},
                {mx[2], aKoef[0][2], aKoef[1][2], aKoef[2][2]}
        };

        bArr[0] = determinant(matrixTemp1) / determinant(matrixTemp2);

        double[][] matrixTemp3 = {
                {1, my, mx[1], mx[2]},
                {mx[0], a[0], aKoef[0][1], aKoef[0][2]},
                {mx[1], a[1], aKoef[1][1], aKoef[2][1]},
                {mx[2], a[2], aKoef[1][2], aKoef[2][2]}
        };

        bArr[1] = determinant(matrixTemp3) / determinant(matrixTemp2);

        double[][] matrixTemp4 = {
                {1, mx[0], my, mx[2]},
                {mx[0], aKoef[0][0], a[0], aKoef[0][2]},
```

```
                {mx[1], aKoef[0][1], a[1], aKoef[2][1]},
                {mx[2], aKoef[0][2], a[2], aKoef[2][2]}
        };
        bArr[2] = determinant(matrixTemp4) / determinant(matrixTemp2);

        double[][] matrixTemp5 = {
                {1, mx[0], mx[1], my},
                {mx[0], aKoef[0][0], aKoef[0][1], a[0]},
                {mx[1], aKoef[0][1], aKoef[1][1], a[1]},
                {mx[2], aKoef[0][2], aKoef[1][2], a[2]}
        };

        bArr[3] = determinant(matrixTemp5) / determinant(matrixTemp2);

        System.out.println("\nНатуралізоване рівняння регресії: ");
        System.out.printf("y = %.2f", bArr[0]);
        if (bArr[1] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[1]), symbols.get(0), symbols.get(3));
        if (bArr[2] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[2]), symbols.get(0), symbols.get(4));
        if (bArr[3] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[3]), symbols.get(0), symbols.get(5));
        System.out.println();

        System.out.println("\nПеревірка: ");
        boolean ok = false;
        for (int i = 0; i < 4; i++) {
           ok = (float) (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] * xArr[i][2]) == (float)
yAverage[i];
           System.out.printf("%.2f = %.2f\n", (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] *
xArr[i][2]), yAverage[i]);
        }
        if (ok) System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії b%s,b%s,b%s,b%s визначено
правильно",
              symbols.get(2), symbols.get(3), symbols.get(4), symbols.get(5));
        else System.out.printf("\nНатуралізовані коефіцієнти рівняння регресії b%s,b%s,b%s,b%s визначено
неправильно",
              symbols.get(2), symbols.get(3), symbols.get(4), symbols.get(5));

        double[] aNorm = new double[4];
        sum = 0;
        for (int i = 0; i < 4; i++) {
           sum += yAverage[i];
        }

        aNorm[0] = sum /4.;
        aNorm[1] = bArr[1]*(MaxX1 - MinX1)/2.;
        aNorm[2] = bArr[2]*(MaxX2 - MinX2)/2.;
        aNorm[3] = bArr[3]*(MaxX3 - MinX3)/2.;


        System.out.println("\nНормоване рівняння регресії: ");
        System.out.printf("y = %.2f", aNorm[0]);
        if (aNorm[1] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(aNorm[1]), symbols.get(0), symbols.get(3));
        if (aNorm[2] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(aNorm[2]), symbols.get(0), symbols.get(4));
        if (aNorm[3] < 0) System.out.print(" - ");
```

```
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(aNorm[3]), symbols.get(0), symbols.get(5));
        System.out.println();


        System.out.println("\nПеревірка: ");
        for (int i = 0; i < 4; i++) {
            ok = (float) (aNorm[0] + aNorm[1] * x[i][1] + aNorm[2] * x[i][2] + aNorm[3] * x[i][3]) == (float)
yAverage[i];
            System.out.printf("%.2f = %.2f\n", (aNorm[0] + aNorm[1] * x[i][1] + aNorm[2] * x[i][2] + aNorm[3] *
x[i][3]), yAverage[i]);

        }
        if (ok) {
            System.out.printf("\nНормовані коефіцієнти рівняння регресії a%s,a%s,a%s,a%s визначено
правильно",
                    symbols.get(2), symbols.get(3), symbols.get(4), symbols.get(5));
            System.out.println();
        }
        else {
            System.out.printf("\nНормовані коефіцієнти рівняння регресії a%s,a%s,a%s,a%s визначено
неправильно",
                    symbols.get(2), symbols.get(3), symbols.get(4), symbols.get(5));
            System.out.println();
        }



        //критерій Кохрена


        for (int i = 0; i < 3; i++) {
            sum = 0;
            double[] yTemp = y.get(i);
            for (int j = 0; j < m; j++) {
                sum += Math.pow((yTemp[j] - yAverage[i]), 2);
            }
            dispersionArr[i] = sum / m;
        }

        double maxDispersion = dispersionArr[0];
        for (int i = 0; i < 4; i++) {
            if (maxDispersion < dispersionArr[i]) maxDispersion = dispersionArr[i];
        }

        double Gp;
        sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += dispersionArr[i];
        }
        Gp = maxDispersion / sum;

        f1 = m - 1;
        f2 = 4;
        q = 0.05;

        double[] KTable = {0.9065, 0.7679, 0.6841, 0.6287, 0.5892, 0.5598, 0.5365, 0.5175, 0.5017, 0.4884, 0.4366,
0.372, 0.3093, 0.25};
        double Gt;

        if (f1 <= 1) Gt = KTable[0];
        else if (f1 <= 2) Gt = KTable[1];
        else if (f1 <= 3) Gt = KTable[2];
```

```
            else if (f1 <= 4) Gt = KTable[3];
            else if (f1 <= 5) Gt = KTable[4];
            else if (f1 <= 6) Gt = KTable[5];
            else if (f1 <= 7) Gt = KTable[6];
            else if (f1 <= 8) Gt = KTable[7];
            else if (f1 <= 9) Gt = KTable[8];
            else if (f1 <= 10) Gt = KTable[9];
            else if (f1 <= 16) Gt = KTable[10];
            else if (f1 <= 36) Gt = KTable[11];
            else if (f1 <= 144) Gt = KTable[12];
            else {Gt = KTable[13];}



            if (Gp < Gt) {
                System.out.printf("Gp = %.2f < Gt = %.2f\n" , Gp, Gt);
                System.out.println("Дисперсії однорідні\n");
                work = false;
            } else {work = true; System.out.printf("Gp = %.2f > Gt = %.2f\n" , Gp, Gt);}
            m++;
            if (work)
                System.out.println("ДИСПЕРСІЇ НЕОДНОРІДНІ\nПОМИЛКА : Gp > Gt \nЗБІЛЬШУЄМО
КІЛЬКІСТЬ ДОСЛІДІВ : m+1\n");
        }

        //критерій Стьюдента
        double Beta;
        double BetaKvadrat;
        double BetaKvadratAverage;
        sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += dispersionArr[i];
        }
        BetaKvadratAverage = sum / 4;
        BetaKvadrat = BetaKvadratAverage/(4.*m);
        Beta = Math.sqrt(BetaKvadrat);

        double[] beta = new double[4];
        for (int i = 0; i < 4; i++) {
            sum = 0;
            for (int j = 0; j < 4; j++) {
                sum += yAverage[j] * x[j][i];
            }
            beta[i] = sum / 4;
        }

        double[] t = new double[4];

        for (int i = 0; i < 4; i++) {
            t[i] = Math.abs(beta[i])/Beta;
        }

        int f3 = f1*f2;
        double[] studentTable = {2.306,2.262,2.228,2.201,2.179,2.16};

        double stNow;
        if (f3>=16) {
            stNow = studentTable[5];
        }
        else {
            stNow = studentTable[f3-8];
        }
```

```
        int d = 4;

        if (t[0] < stNow) {bArr[0] = 0;d--;}
        if (t[1] < stNow) {bArr[1] = 0;d--;}
        if (t[2] < stNow) {bArr[2] = 0;d--;}
        if (t[3] < stNow) {bArr[3] = 0;d--;}

        System.out.println("Рівняння регресії після критерію Стьюдента: ");
        System.out.printf("y = %.2f", bArr[0]);
        if (bArr[1] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[1]), symbols.get(0), symbols.get(3));
        if (bArr[2] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[2]), symbols.get(0), symbols.get(4));
        if (bArr[3] < 0) System.out.print(" - ");
        else System.out.print(" + ");
        System.out.printf("%.2f%sx%s", Math.abs(bArr[3]), symbols.get(0), symbols.get(5));
        System.out.println();

        double[] yAverageAfterStudent = new double[4];


        System.out.println("\nПеревірка: ");
        for (int i = 0; i < 4; i++) {
            System.out.printf("%.2f %s %.2f\n",
                    yAverageAfterStudent[i] = (bArr[0] + bArr[1] * xArr[i][0] + bArr[2] * xArr[i][1] + bArr[3] *
xArr[i][2]),
                    symbols.get(1), yAverage[i]);
        }

        //критерій Фішера
        int f4 = 4 - d;
        double sKvadratAdekv;
        sum = 0;
        for (int i = 0; i < 4; i++) {
            sum += Math.pow(yAverageAfterStudent[i] - yAverage[i],2);
        }
        sKvadratAdekv = sum * (float)(m/(4-d));

        double Fp = sKvadratAdekv/BetaKvadratAverage;

        double[][] fisherTable = {
                {5.3,4.5,4.1,3.8,3.7,3.6,3.3,3.1,2.9},
                {4.8,3.9,3.5,3.3,3.1,3.0,2.7,2.5,2.3},
                {4.5,3.6,3.2,3.0,2.9,2.7,2.4,2.2,2.0},
                {4.4,3.5,3.1,2.9,2.7,2.6,2.3,2.1,1.9}
        };

        double fisherNow = 0;
        if (f4<=1) fisherNow = fisherTable[m-3][0];
        else if (f4<=2) fisherNow = fisherTable[m-3][1];
        else if (f4<=3) fisherNow = fisherTable[m-3][2];
        else fisherNow = fisherTable[m-3][3];
        if (Fp < fisherNow) {
            System.out.printf("\nFp = %.2f < Ft = %.2f\n" , Fp, fisherNow);}
        else if (Fp > fisherNow) {
            System.out.printf("\nFp = %.2f > Ft = %.2f\n" , Fp, fisherNow);}

        if (Fp > fisherNow) System.out.println("\nРівняння регресії неадекватно оригіналу при q = 0.05");
        else System.out.println("\nРівняння регресії адекватно оригіналу при q = 0.05");
    }
}
```

Результат роботи:

```
Задайте значення m:
3
Лінійне рівняння регресії для нормованих значень x має вигляд : y = b0 + b1·x1 + b2·x2 + b3·x3
Нормована матриця планування експерименту :
X0  X1  X2  X3  Y1              Y2              Y3
 1  -1  -1  -1  204.44423       208.53166       220.73506
 1  -1   1   1  219.47406       214.74936       214.40819
 1   1  -1   1  211.67345       205.7705        214.89761
 1   1   1  -1  216.33434       210.09538       199.21185
Матриця планування експерименту :
X1  X2  X3  Y1              Y2              Y3
-5  -15  15 204.44423       208.53166       220.73506
-5   35  30 219.47406       214.74936       214.40819
 15 -15  30 211.67345       205.7705        214.89761
 15  35  15 216.33434       210.09538       199.21185

Натуралізоване рівняння регресії:
y = 207,03 - 0,20·x1 + 0,03·x2 + 0,24·x3

Перевірка:
211,24 = 211,24
216,21 = 216,21
210,78 = 210,78
208,55 = 208,55

Натуралізовані коефіцієнти рівняння регресії b0,b1,b2,b3 визначено правильно
Нормоване рівняння регресії:
y = 211,69 - 2,03·x1 + 0,69·x2 + 1,80·x3

Перевірка:
211,24 = 211,24
216,21 = 216,21
210,78 = 210,78
208,55 = 208,55

Нормовані коефіцієнти рівняння регресії a0,a1,a2,a3 визначено правильно
Gp = 0,71 < Gt = 0,77
Дисперсії однорідні

Рівняння регресії після критерію Стьюдента:
y = 207,03 + 0,00·x1 + 0,00·x2 + 0,00·x3

Перевірка:
207,03 ≠ 211,24
207,03 ≠ 216,21
207,03 ≠ 210,78
207,03 ≠ 208,55

Fp = 7,01 > Ft = 3,50

Рівняння регресії неадекватно оригіналу при q = 0.05
```

Відповіді на теоретичні питання:

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 Стьюдента?

   Критерій Стьюдента 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 $t > t_{кр}$, тонуль-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.