Національний технічний університет України
«Київський політехнічний інститут ім. І. 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 ІО-93
Варченко Є. В.
Номер у списку групи – 3

Перевірив:
ас. Регіда П. Г.

Київ – 2021

Лабораторна робота №2

Тема: «Проведення двофакторного експерименту з використанням лінійного рівняння регресії».

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

Завдання

1. Записати лінійне рівняння регресії.
2. Обрати тип двофакторного експерименту і скласти матрицю планування для нього з використанням додаткового нульового фактору (x_o=1).
3. Провести експеримент в усіх точках повного факторного простору (знайти значення функції відгуку y). Значення функції відгуку задати випадковим чином у відповідності до варіанту у діапазоні $y_{min} \div y_{max}$

   $y_{max} = (30 - N_{варіанту})*10$,
   $y_{min} = (20 - N_{варіанту})*10$.

   Варіанти обираються по номеру в списку в журналі викладача.

№варіанта	x_1		x_2	
	min	max	min	max
303	-20	30	-20	40

4. Перевірити однорідності дисперсії за критерієм Романовського
5. Знайти коефіцієнти нормованих рівнянь регресії і виконати перевірку (підставити значення нормованих факторів і коефіцієнтів у рівняння).
6. Провести натуралізацію рівняння регресії й виконати перевірку натуралізованого рівняння.
7. Написати комп'ютерну програму, яка все це виконує.

$y_{min} = (30 - N_{варіанту}) * 10 = (30 - 3) * 10 = 270$

$y_{max} = (20 - N_{варіанту}) * 10 = (20 - 3) * 10 = 170$

Код програми

```
import kotlin.math.*
import kotlin.random.*

const val variant = 3

const val yMin = (20 - variant) * 10
const val yMax = (30 - variant) * 10

const val m = 6
const val p = 0.95f
const val rCritical = 2.1f

const val x1Min =  -20
const val x1Max = 30
const val x2Min = -20
const val x2Max = 40

fun variance(yArray: IntArray, yAverage: Float): Float {
    var result = 0f

    for (i in 0 until m) {
        result += (yArray[i] - yAverage).pow(2)
    }

    return result
}
```

```
fun main() {
    val yMatrix = Array(3) { IntArray(m) { Random.nextInt(yMin, yMax) } }
    val yAveragesArray = Array(3) { yMatrix[it].sum() / m.toFloat() }

    val sigmasArray = arrayOf(
        variance(yMatrix[0], yAveragesArray[0]),
        variance(yMatrix[1], yAveragesArray[1]),
        variance(yMatrix[2], yAveragesArray[2])
    )

    val sigmaTheta = sqrt((2 * (2 * m - 2)) / (m * (m - 4)).toFloat())

    val fuvArray = arrayOf(
        arrayOf(sigmasArray[0], sigmasArray[1]).maxOrNull()!! / arrayOf(sigmasArray[0],
sigmasArray[1]).minOrNull()!!,
        arrayOf(sigmasArray[0], sigmasArray[2]).maxOrNull()!! / arrayOf(sigmasArray[0],
sigmasArray[2]).minOrNull()!!,
        arrayOf(sigmasArray[2], sigmasArray[1]).maxOrNull()!! / arrayOf(sigmasArray[2],
sigmasArray[1]).minOrNull()!!,
    )

    val thetauvArray = arrayOf(
        ((m - 2) / m.toFloat()) * fuvArray[0],
        ((m - 2) / m.toFloat()) * fuvArray[1],
        ((m - 2) / m.toFloat()) * fuvArray[2]
    )

    val ruvArray = arrayOf(
        (thetauvArray[0] - 1).absoluteValue / sigmaTheta,
        (thetauvArray[1] - 1).absoluteValue / sigmaTheta,
        (thetauvArray[2] - 1).absoluteValue / sigmaTheta,
    )

    val mx1 = (-1 + 1 - 1) / 3f
    val mx2 = (-1 - 1 + 1) / 3f
```

```
    val my = yAveragesArray.sum() / 3f

    val a1 = (1 + 1 + 1) / 3f
    val a2 = (1 - 1 - 1) / 3f
    val a3 = (1 + 1 + 1) / 3f

    val a11 = ((-1 * yAveragesArray[0]) + (1 * yAveragesArray[1]) - (1 * yAveragesAr-
ray[2])) / 3f
    val a22 = ((-1 * yAveragesArray[0]) - (1 * yAveragesArray[1]) + (1 * yAveragesAr-
ray[2])) / 3f

    val determinant = ((a1 * a3) + (mx1 * a2 * mx2) + (mx2 * mx1 * a2) - (mx2 * mx2 * a1)
- (mx1 * mx1 * a3) - (a2 * a2))

    val b0 = ((my * a1 * a3) + (mx1 * a2 * a22) + (mx2 * a11 * a2) - (a22 * a1 * mx2) -
(mx1 * a11 * a3) - (my * a2 * a2)) / determinant
    val b1 = ((a3 * a11) + (a22 * mx1 * mx2) + (a2 * my * mx2) - (mx2 * mx2 * a11) - (a22
* a2) - (mx1 * my * a3)) / determinant
    val b2 = ((a1 * a22) + (a2 * mx1 * my) + (mx1 * mx2 * a11) - (mx2 * my * a1) - (mx1 *
mx1 * a22) - (a2 * a11)) / determinant

    val deltaX1 = (x1Max - x1Min).absoluteValue / 2f
    val deltaX2 = (x2Max - x2Min).absoluteValue / 2f

    val x10 = (x1Max + x1Min) / 2f
    val x20 = (x2Max + x2Min) / 2f

    val a0s = b0 - (b1 * x10 / deltaX1) - (b2 * x20 / deltaX2)
    val a1s = b1 / deltaX1
    val a2s = b2 / deltaX2

    for (i in yMatrix) {
        for (j in i) print("$j ")

        println()
    }
```

```
print("\nСередні значення: ")

for (i in yAveragesArray) print("$i ")

println("\n\nsigma1 = ${sigmasArray[0]}")
println("sigma2 = ${sigmasArray[1]}")
println("sigma3 = ${sigmasArray[2]}\n")

println("sigmatheta = $sigmaTheta\n")

println("Fuv1 = ${fuvArray[0]}")
println("Fuv2 = ${fuvArray[1]}")
println("Fuv3 = ${fuvArray[2]}\n")

println("thetauv1 = ${thetauvArray[0]}")
println("thetauv2 = ${thetauvArray[1]}")
println("thetauv3 = ${thetauvArray[2]}\n")

println("Ruv1 = ${ruvArray[0]}")
println("Ruv2 = ${ruvArray[1]}")
println("Ruv3 = ${ruvArray[2]}\n")

if (ruvArray[0] < rCritical && ruvArray[1] < rCritical && ruvArray[2] < rCritical) {
    println("Дисперсія однорідна\n")
} else {
    println("Дисперсія неоднорідна\n")
}

println("a0 = $a0s")
println("a1 = $a1s")
println("a2 = $a2s\n")

println("Натуралізоване рівняння регресії: ($a0s, $a1s, $a2s)\n")

println("Перевірка натуралізованого рівняння регресії:")
println("${yAveragesArray[0]} = ${a0s + x1Min * a1s + x2Min * a2s}")
println("${yAveragesArray[1]} = ${a0s + x1Max * a1s + x2Min * a2s}")
```

```
    println("${yAveragesArray[2]} = ${a0s + x1Min * a1s + x2Max * a2s}\n")

    println("b0 = $b0")
    println("b1 = $b1")
    println("b2 = $b2\n")

    println("Нормоване рівняння регресії: ($b0, $b1, $b2)\n")

    println("Перевірка нормованого рівняння регресії:")
    println("${yAveragesArray[0]} = ${b0 - b1 - b2}")
    println("${yAveragesArray[1]} = ${b0 + b1 - b2}")
    println("${yAveragesArray[2]} = ${b0 - b1 + b2}")
}
```

Результати роботи програми

```
181 219 170 243 220 184
268 242 185 243 236 221
217 269 172 185 174 269

Середні значення: 202.83333 232.5 214.33333

sigma1 = 4078.8335
sigma2 = 3861.5
sigma3 = 10263.334

sigmatheta = 1.2909944

Fuv1 = 1.0562822
Fuv2 = 2.5162425
Fuv3 = 2.657862

thetauv1 = 0.7041881
thetauv2 = 1.677495
thetauv3 = 1.771908

Ruv1 = 0.22913492
Ruv2 = 0.5247854
Ruv3 = 0.59791744

Дисперсія однорідна
```

```
a0 = 220.2444
a1 = -0.026666906
a2 = 0.38055542

Натуралізоване рівняння регресії: (220.2444, -0.026666906, 0.38055542)

Перевірка натуралізованого рівняння регресії:
213.16667 = 213.16663
211.83333 = 211.83328
236.0 = 235.99995

b0 = 223.91663
b1 = -0.66667265
b2 = 11.416662

Нормоване рівняння регресії: (223.91663, -0.66667265, 11.416662)

Перевірка нормованого рівняння регресії:
213.16667 = 213.16664
211.83333 = 211.8333
236.0 = 235.99995
```

Висновки

- Ознайомилися з темою роботи.
- Були здобуті необхідні навички для виконання завдань.
- Розроблено програму, яка виконує поставлену задачу.
- Вище приведені результати свідчать про успішне виконання умов завдань.
- Основну мету лабораторної роботи було досягнуто.