

Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра обчислювальної техніки

Лабораторна робота №4
«Проведення трьохфакторного експерименту
при використанні рівняння регресії з урахуванням ефекту взаємодії»

Виконав:
студент II курсу ФІОТ
групи ІВ-92
Залога Андрій

Перевірив:
Регіда П.Г.

Київ – 2021

Мета роботи: 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.

$$y_{\max} = 200 + x_{\text{cp max}};$$

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

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

2. Знайти коефіцієнти лінійного рівняння регресії. Записати лінійне рівняння регресії.
3. Провести 3 статистичні перевірки.
4. Зробити висновки по адекватності регресії та значимості окремих коефіцієнтів і записати скореговане рівняння регресії.
5. Написати комп'ютерну програму, яка усе це виконує.

Варіант завдання:

Варіант	X ₁		X ₂		X ₃	
	min	max	min	max	min	max
208	-5	15	-35	10	-35	-10

Довірча ймовірність дорівнює 0.95, а рівень значимості $q = 0.05$.

Роздруківка тексту програми:

```
import random
from prettytable import PrettyTable

m = 3
n = 8
x1_min = -5
x1_max = 15
x2_min = -35
x2_max = 10
x3_min = -35
x3_max = -10
y_min = 200 + (x1_min + x2_min + x3_min) / 3
y_max = 200 + (x1_max + x2_max + x3_max) / 3
y_matrix = [[random.randint(int(y_min), int(y_max)) for _ in range(m)] for _ in
range(n)]
y_average = [round(sum(i) / len(i), 3) for i in y_matrix]
xn = [[-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 = sum(y_average) / n
b1 = sum([y_average[i] * xn[i][0] for i in range(n)]) / n
b2 = sum([y_average[i] * xn[i][1] for i in range(n)]) / n
b3 = sum([y_average[i] * xn[i][2] for i in range(n)]) / n
b12 = sum([y_average[i] * xn[i][0] * xn[i][1] for i in range(n)]) / n
b13 = sum([y_average[i] * xn[i][0] * xn[i][2] for i in range(n)]) / n
b23 = sum([y_average[i] * xn[i][1] * xn[i][2] for i in range(n)]) / n
b123 = sum([y_average[i] * xn[i][0] * xn[i][1] * xn[i][2] for i in range(n)]) / n

plan_matrix = [[x1_min, x2_min, x3_min, x1_min * x2_min, x1_min * x3_min, x2_min *
x3_min, x1_min * x2_min * x3_min],
               [x1_min, x2_min, x3_max, x1_min * x2_min, x1_min * x3_max, x2_min *
x3_max, x1_min * x2_min * x3_max],
               [x1_min, x2_max, x3_min, x1_min * x2_max, x1_min * x3_min, x2_max *
x3_min, x1_min * x2_max * x3_min],
               [x1_min, x2_max, x3_max, x1_min * x2_max, x1_min * x3_max, x2_max *
x3_max, x1_min * x2_max * x3_max],
               [x1_max, x2_min, x3_min, x1_max * x2_min, x1_max * x3_min, x2_min *
x3_min, x1_max * x2_min * x3_min],
               [x1_max, x2_min, x3_max, x1_max * x2_min, x1_max * x3_max, x2_min *
x3_max, x1_max * x2_min * x3_max],
               [x1_max, x2_max, x3_min, x1_max * x2_max, x1_max * x3_min, x2_max *
x3_min, x1_max * x2_max * x3_min],
               [x1_max, x2_max, x3_max, x1_max * x2_max, x1_max * x3_max, x2_max *
x3_max, x1_max * x2_max * x3_max]]

y_result = []
for i in range(n):
    y_result.append(b0 + b1 * plan_matrix[i][0] + b2 * plan_matrix[i][1] + b3 *
plan_matrix[i][2] +
                    b12 * plan_matrix[i][3] + b13 * plan_matrix[i][4] + b23 *
plan_matrix[i][5] +
                    b123 * plan_matrix[i][6])
```

```

dispersion = [round(sum([(y_matrix[j][i] - y_average[i]) ** 2 for i in range(m)]) /
m, 3) for j in range(n)]

table1 = PrettyTable()
table1.field_names = ["X0", "X1", "X2", "X3", "X12", "X13", "X23", "X123", "Y1",
"Y2", "Y3", "Y average", "S^2"]
x0 = [[1] for _ in range(n)]
for i in range(n):
    table1.add_row([*x0[i], *plan_matrix[i], *y_matrix[i], y_average[i],
dispersion[i]])
print('Матриця планування:')
print(table1)
print()

# Критерій Кохрена
print("Перевірка за критерієм Кохрена:")
gp = max(dispersion) / sum(dispersion)
gt = 0.5157
if gp < gt:
    print("За критерієм Кохрена дисперсія однорідна")
    print("{} < {}".format(round(gp, 3), round(gt, 3)))
else:
    print("За критерієм Кохрена дисперсія неоднорідна")
    print("{} > {}".format(round(gp, 3), round(gt, 3)))
print()

# Критерій Стюдента
print("Перевірка значущості коефіцієнтів за критерієм Стюдента")
d = 8
sb = sum(dispersion) / n
s_beta_2 = sb / (n * m)
s_beta = s_beta_2 ** (1 / 2)
bb = [b0, b1, b2, b3, b12, b13, b23, b123]
t_list = [abs(bb[i]) / s_beta for i in range(n)]
tt = 2.120
b_list = [b0, b1, b2, b3, b12, b13, b23, b123]
for i in range(n):
    if t_list[i] < tt:
        b_list[i] = 0
        d -= 1
for i in range(len(t_list)):
    print('t{} = {}'.format(i, round(t_list[i], 3)))
print()

# Критерій Фішера
print("Перевірка адекватності за критерієм Фішера")
y_reg = [b0 + b1 * plan_matrix[i][0] + b2 * plan_matrix[i][1] + b3 *
plan_matrix[i][2] +
        b12 * plan_matrix[i][3] + b13 * plan_matrix[i][4] + b23 * plan_matrix[i][5]
+
        b123 * plan_matrix[i][6] for i in range(n)]
sad = (m / (n - d)) * int(sum([(y_reg[i] - y_average[i]) ** 2 for i in range(n)]))
fp = sad / sb
if fp < 4.5:
    print('Рівняння регресії адекватне оригіналу на рівні 0.05')
else:
    print('Рівняння регресії неадекватне оригіналу на рівні 0.05')
print()

print('Рівняння:')
print('y = {} + {} * x1 + {} * x2 + {} * x3 + {} * x1x2 + {} * x1x3 + {} * x2x3 + {}
* x1x2x3'

```

```

        .format(round(b0, 3), round(b1, 3), round(b2, 3), round(b3, 3), round(b12, 3),
round(b13, 3), round(b23, 3),
                round(b123, 3)))
print()
for i in range(len(y_result)):
    print('y{0} = {0}'.format(i+1, round(y_result[i], 3)))

```

Результати роботи програми:

Матриця планування:

X0	X1	X2	X3	X12	X13	X23	X123	Y1	Y2	Y3	Y average	S^2
1	-5	-35	-35	175	175	1225	-6125	187	204	201	197.333	134.445
1	-5	-35	-10	175	50	350	-1750	204	191	204	199.667	168.786
1	-5	10	-35	-50	175	-350	1750	176	184	193	184.333	258.556
1	-5	10	-10	-50	50	-100	500	182	198	195	191.667	117.222
1	15	-35	-35	-525	-525	1225	18375	175	186	182	181.0	230.331
1	15	-35	-10	-525	-150	350	5250	176	192	202	190.0	275.334
1	15	10	-35	150	-525	-350	-5250	184	205	193	194.0	93.776
1	15	10	-10	150	-150	-100	-1500	203	192	190	195.0	41.004

Перевірка за критерієм Кохрена:

За критерієм Кохрена дисперсія однорідна

0.209 < 0.516

Перевірка значущості коефіцієнтів за критерієм Стюдента

t0 = 73.098

t1 = 0.62

t2 = 0.143

t3 = 0.938

t4 = 1.86

t5 = 0.016

t6 = 0.143

t7 = 0.62

Перевірка адекватності за критерієм Фішера

Рівняння регресії неадекватне оригіналу на рівні 0.05

Рівняння:

$y = 191.625 + -1.625 * x1 + -0.375 * x2 + 2.459 * x3 + 4.875 * x1x2 + 0.041 * x1x3 + -0.375 * x2x3 + -1.625 * x1x2x3$

y1 = 10480.965

y2 = 3755.99

y3 = -2839.035

y4 = -845.26

y5 = -32805.585

y6 = -11072.31

y7 = 9449.415

y8 = 3338.94

Висновок:

У ході лабораторної роботи я змодлював трьохфакторний експеримент. Також я реалізував 3 статистичні перевірки за критерієм Кохрена, Стюдента та Фішера. Знайшов рівняння регресії адекватне об'єкту.