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 ІІ курсу ФІОТ
групи ІВ-91
Чопик Н.О.
Залікова - 9130

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

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

Завдання:

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.

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

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

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: 128

№ варіанта	X1		X2		X3	
	min	max	min	max	min	max
128	-40	20	-25	10	-25	-10

Лістинг програми:

```
import math
import numpy as np
from numpy import average, transpose
from numpy.linalg import solve
from prettytable import PrettyTable
from scipy.stats import f
from scipy.stats import t as criterion_t
from functools import partial
from random import randint


def get_average(matrix):
    return [round(sum(matrix[k1]) / m, 3) for k1 in range(n)]


def get_dispersion(matrix):
    return [round(sum([((k1 - get_average(matrix)[j]) ** 2) for k1 in matrix[j]]) /
m, 3) for j in range(n)]


def fill_x_matrix():
    [x1_list.append(x1[0 if i == -1 else 1]) for i in x1_factor]
    [x2_list.append(x2[0 if i == -1 else 1]) for i in x2_factor]
    [x3_list.append(x3[0 if i == -1 else 1]) for i in x3_factor]
    [x1x2_list.append(a * b) for a, b in zip(x1_list, x2_list)]
    [x1x3_list.append(a * b) for a, b in zip(x1_list, x3_list)]
    [x2x3_list.append(a * b) for a, b in zip(x2_list, x3_list)]
    [x1x2x3_list.append(a * b * c) for a, b, c in zip(x1_list, x2_list, x3_list)]


def cohren():
    def cohren_teor(f1, f2, q=0.05):
        q1 = q / f1
        fisher_value = f.ppf(q=1 - q1, dfn=f2, dfd=(f1 - 1) * f2)
        return fisher_value / (fisher_value + f1 - 1)

    Gp = max(get_dispersion(matrix_y)) / sum(get_dispersion(matrix_y))
    Gt = cohren_teor(F1, F2)
    # print("\nGp = ", Gp, " Gt = ", Gt)
    return Gp < Gt


def fisher():
    fisher_teor = partial(f.ppf, q=1 - 0.05)
    Ft = fisher_teor(dfn=F4, dfd=F3)
    return Ft


m, n, d = 3, 8, 8

x0_factor = [1, 1, 1, 1, 1, 1, 1, 1]
x1_factor = [-1, -1, 1, 1, -1, -1, 1, 1]
x2_factor = [-1, 1, -1, 1, -1, 1, -1, 1]
x3_factor = [-1, 1, 1, -1, 1, -1, -1, 1]
x1x2_factor = [a * b for a, b in zip(x1_factor, x2_factor)]
x1x3_factor = [a * b for a, b in zip(x1_factor, x3_factor)]
x2x3_factor = [a * b for a, b in zip(x2_factor, x3_factor)]
x1x2x3_factor = [a * b * c for a, b, c in zip(x1_factor, x2_factor, x3_factor)]

x1_list = []
```

```
x2_list = []
x3_list = []
x1x2_list = []
x1x3_list = []
x2x3_list = []
x1x2x3_list = []
x_main_list = [x0_factor, x1_list, x2_list, x3_list, x1x2_list, x1x3_list, x2x3_list,
x1x2x3_list]
x_factor_list = [x0_factor, x1_factor, x2_factor, x3_factor, x1x2_factor,
x1x3_factor, x2x3_factor, x1x2x3_factor]

list_bi = []

F1 = m - 1
F2 = n
F3 = F1 * F2
F4 = n - d

x1 = [-40, 20]
x2 = [-25, 10]
x3 = [-25, -10]
x_tuple = (x1, x2, x3)
x_max_average = average([i[1] for i in x_tuple])
x_min_average = average([i[0] for i in x_tuple])
y_max = int(200 + x_max_average)
y_min = int(200 + x_min_average)
y_min_max = [y_min, y_max]
matrix_y = [[randint(y_min_max[0], y_min_max[1]) for _ in range(m)] for _ in
range(n)]

fill_x_matrix()

dispersion = get_dispersion(matrix_y)
sum_dispersion = sum(dispersion)
y_average = get_average(matrix_y)

column_names1 = ["X0", "X1", "X2", "X3", "X1X2", "X1X3", "X2X3", "X1X2X3", "Y1",
"Y2", "Y3", "Y", "S^2"]
trans_y_mat = transpose(matrix_y).tolist()

list_for_solve_a = list(zip(*x_main_list))
list_for_solve_b = x_factor_list

for k in range(n):
    S = 0
    for i in range(n):
        S += (list_for_solve_b[k][i] * y_average[i]) / n
    list_bi.append(round(S, 5))

pt = PrettyTable()
cols = x_factor_list
[cols.extend(ls) for ls in [trans_y_mat, [y_average], [dispersion]]]
[pt.add_column(column_names1[coll_id], cols[coll_id]) for coll_id in range(13)]
print(pt)
print('Рівняння регресії з коефіцієнтами від нормованих значень факторів')
print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 + {}*x2x3 + {}*x1x2x3
\n".format(*list_bi))

pt = PrettyTable()
cols = x_main_list
[cols.extend(ls) for ls in [trans_y_mat, [y_average], [dispersion]]]
[pt.add_column(column_names1[coll_id], cols[coll_id]) for coll_id in range(13)]
```

```
print(pt)

list_ai = [round(i, 5) for i in solve(list_for_solve_a, y_average)]
print('Рівняння регресії з коефіцієнтами від натуральних значень факторів')
print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 + {}*x2x3 +
{}*x1x2x3\n".format(*list_ai))

print('Критерій Кохрена:')
if cohren():
    print("Дисперсія однорідна\n")
    Dispersion_B = sum_dispersion / n
    Dispersion_beta = Dispersion_B / (m * n)
    S_beta = math.sqrt(abs(Dispersion_beta))
    b_list = np.zeros(8).tolist()
    for i in range(n):
        b_list[0] += (y_average[i] * x0_factor[i]) / n
        b_list[1] += (y_average[i] * x1_factor[i]) / n
        b_list[2] += (y_average[i] * x2_factor[i]) / n
        b_list[3] += (y_average[i] * x3_factor[i]) / n
        b_list[4] += (y_average[i] * x1x2_factor[i]) / n
        b_list[5] += (y_average[i] * x1x3_factor[i]) / n
        b_list[6] += (y_average[i] * x2x3_factor[i]) / n
        b_list[7] += (y_average[i] * x1x2x3_factor[i]) / n
    t_list = [abs(b_list[i]) / S_beta for i in range(0, n)]

    significant_coefficients = 0
    non_significant_coefficients = 0
    for i, j in enumerate(t_list):
        # print(f"t{i} = {b_list[i]}")
        if j < criterion_t.ppf(q=0.975, df=F3):
            b_list[i] = 0
            d -= 1
            non_significant_coefficients += 1
        else:
            significant_coefficients += 1
    print(f'Кількість значущих коефіцієнтів за критерієм Стьюдента:
{significant_coefficients}')
    print(f'Кількість не значущих коефіцієнтів за критерієм Стьюдента:
{non_significant_coefficients}\n')
    print("Рівняння: \ny = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 + {}*x2x3 +
{}*x1x2x3\n".format(*b_list))

    Y_counted = [sum([b_list[0], *[b_list[i] * x_main_list[1:][j][i] for i in
range(n)]])
                 for j in range(n)]
    Dispersion_ad = 0
    for i in range(len(Y_counted)):
        Dispersion_ad += ((Y_counted[i] - y_average[i]) ** 2) * m / (n - d)
    Fp = Dispersion_ad / Dispersion_beta
    Ft = fisher()

    if Ft > Ft:
        print('За критерієм Фішера рівняння регресії неадекватно оригіналу')
    else:
        print('За критерієм Фішера рівняння регресії адекватно оригіналу')

else:
    print("Дисперсія не однорідна\n")
```

Результат виконання роботи:

```
  "C:/Users/Nazar/Desktop/КПІ/Методи наукових досліджень/Lab4/Lab4.py"
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| X0 | X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |    Y    |   S^2   |
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| 1  | -1 | -1 | -1 |  1   |  1   |  1   |   -1   | 196 | 185 | 190 | 190.333 |  20.222 |
| 1  | -1 | 1  | 1  |  -1  |  -1  |  1   |   -1   | 188 | 179 | 197 |  188.0  |   54.0  |
| 1  | 1  | -1 | 1  |  -1  |  1   |  -1  |   -1   | 191 | 179 | 172 | 180.667 |  61.556 |
| 1  | 1  | 1  | -1 |  1   |  -1  |  -1  |   -1   | 198 | 181 | 178 | 185.667 |  77.556 |
| 1  | -1 | -1 | 1  |  1   |  -1  |  -1  |   1    | 189 | 184 | 186 | 186.333 |  4.222  |
| 1  | -1 | 1  | -1 |  -1  |  1   |  -1  |   1    | 172 | 180 | 183 | 178.333 |  21.556 |
| 1  | 1  | -1 | -1 |  -1  |  -1  |  1   |   1    | 177 | 188 | 182 | 182.333 |  20.222 |
| 1  | 1  | 1  | 1  |  1   |  1   |  1   |   1    | 206 | 175 | 176 | 185.667 | 206.889 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
Рівняння регресії з коефіцієнтами від нормованих значень факторів
y = 184.66663 + -1.08312*x1 + -0.24987*x2 + 0.50013*x3 + 2.33338*x1x2 + -0.91662*x1x3 + 1.91662*x2x3 +
 -1.50013*x1x2x3


+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+---------+---------+
| X0 |  X1 |  X2 |  X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |    Y    |   S^2   |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+---------+---------+
| 1  | -40 | -25 | -25 | 1000 | 1000 | 625  | -25000 | 196 | 185 | 190 | 190.333 |  20.222 |
| 1  | -40 |  10 | -10 | -400 | 400  | -100 |  4000  | 188 | 179 | 197 |  188.0  |   54.0  |
| 1  |  20 | -25 | -10 | -500 | -200 | 250  |  5000  | 191 | 179 | 172 | 180.667 |  61.556 |
| 1  |  20 |  10 | -25 | 200  | -500 | -250 | -5000  | 198 | 181 | 178 | 185.667 |  77.556 |
| 1  | -40 | -25 | -10 | 1000 | 400  | 250  | -10000 | 189 | 184 | 186 | 186.333 |  4.222  |
| 1  | -40 |  10 | -25 | -400 | 1000 | -250 | 10000  | 172 | 180 | 183 | 178.333 |  21.556 |
| 1  |  20 | -25 | -25 | -500 | -500 | 625  | 12500  | 177 | 188 | 182 | 182.333 |  20.222 |
| 1  |  20 |  10 | -10 | 200  | -200 | -100 | -2000  | 206 | 175 | 176 | 185.667 | 206.889 |
+----+-----+-----+-----+------+------+------+--------+-----+-----+-----+---------+---------+
Рівняння регресії з коефіцієнтами від натуральних значень факторів
y = 186.40244 + -0.12407*x1 + 0.21904*x2 + 0.10689*x3 + -0.00222*x1x2 + -0.00693*x1x3 + 0.01079*x2x3 +
 -0.00038*x1x2x3

Критерій Кохрена:
Дисперсія однорідна

Кількість значущих коефіцієнтів за критерієм Стьюдента: 1
Кількість не значущих коефіцієнтів за критерієм Стьюдента: 7

Рівняння:
y = 184.666625 + 0*x1 + 0*x2 + 0*x3 + 0*x1x2 + 0*x1x3 + 0*x2x3 + 0*x1x2x3
```