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
Мусійчук Я. С.
Залікова - 9121

ПЕРЕВІРИВ:
ас. Регіда П. Г.

Київ – 2021

Мета: 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.

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

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: 119

№ варіанта	X1		X2		X3	
	min	max	min	max	min	max
119	-20	30	5	40	5	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 t_criterium
from functools import partial
from random import randint

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, x2, x3 = (-20, 30), (5, 40), (5, 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]
mat_Y = [[randint(y_min_max[0], y_min_max[1]) for _ in range(m)] for _ in range(N)]

def get_average_y():
    return [round(sum(mat_Y[k1]) / m, 3) for k1 in range(N)]

def get_dispersion():
    return [round(sum([((k1 - get_average_y()[j]) ** 2) for k1 in mat_Y[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():
    q = 0.05
    Gp = max(get_dispersion()) / sum(get_dispersion())
    q1 = q / F1
    fisher_value = f.ppf(q=1 - q1, dfn=F2, dfd=(F1 - 1) * F2)
    Gt = fisher_value / (fisher_value + F1 - 1)
    return Gp < Gt

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


fill_x_matrix()
dispersion = get_dispersion()
sum_dispersion = sum(dispersion)
y_average = get_average_y()

column_names1 = ["X0", "X1", "X2", "X3", "X1X2", "X1X3", "X2X3", "X1X2X3", "Y1",
"Y2", "Y3", "Y", "S^2"]
trans_y_mat = transpose(mat_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("Матриця планування")
print(pt, "\n")


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("Нормована матриця")
print(pt, "\n")
list_ai = [round(i, 5) for i in solve(list_for_solve_a, y_average)]
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))
    beta_list = np.zeros(8).tolist()
    for i in range(N):
        beta_list[0] += (y_average[i] * x0_factor[i]) / N
```

```
        beta_list[1] += (y_average[i] * x1_factor[i]) / N
        beta_list[2] += (y_average[i] * x2_factor[i]) / N
        beta_list[3] += (y_average[i] * x3_factor[i]) / N
        beta_list[4] += (y_average[i] * x1x2_factor[i]) / N
        beta_list[5] += (y_average[i] * x1x3_factor[i]) / N
        beta_list[6] += (y_average[i] * x2x3_factor[i]) / N
        beta_list[7] += (y_average[i] * x1x2x3_factor[i]) / N
    t_list = [abs(beta_list[i]) / S_beta for i in range(0, N)]
    print("Критерій Стьюдента")
    for i, j in enumerate(t_list):
        print(f't{i}={beta_list[i]}')
        if j < t_criterium.ppf(q=0.975, df=F3):
            beta_list[i] = 0
            d -= 1
    print()
    print('Рівняння регресії з коефіцієнтами від нормованих значень факторів')
    print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 + {}*x2x3 +
{}*x1x2x3".format(*list_bi))
    print('Рівняння регресії з коефіцієнтами від натуральних значень факторів')
    print("y = {} + {}*x1 + {}*x2 + {}*x3 + {}*x1x2 + {}*x1x3 + {}*x2x3 +
{}*x1x2x3".format(*list_ai))
    print()
    Y_counted = [sum([beta_list[0], *[beta_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()
    print("Критерій Фіішера")
    if Ft > Fp:
        print("Рівняння регресії адекватне!")
    else:
        print("Рівняння регресії неадекватне.")
else:
    print("Дисперсія неоднорідна")
```

Результат виконання роботи:

```
Матриця планування
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| X0 | X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |    Y    |   S^2   |
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| 1  | -1 | -1 | -1 |  1   |  1   |  1   |   -1   | 224 | 211 | 209 | 214.667 |  44.222 |
| 1  | -1 | 1  | 1  |  -1  |  -1  |  1   |   -1   | 197 | 196 | 213 |  202.0  |  60.667 |
| 1  | 1  | -1 | 1  |  -1  |  1   |  -1  |   -1   | 201 | 219 | 206 | 208.667 |  57.556 |
| 1  | 1  | 1  | -1 |  1   |  -1  |  -1  |   -1   | 214 | 221 | 197 | 210.667 | 101.556 |
| 1  | -1 | -1 | 1  |  1   |  -1  |  -1  |   1    | 201 | 215 | 209 | 208.333 |  32.889 |
| 1  | -1 | 1  | -1 |  -1  |  1   |  -1  |   1    | 209 | 207 | 222 | 212.667 |  44.222 |
| 1  | 1  | -1 | -1 |  -1  |  -1  |  1   |   1    | 225 | 203 | 216 | 214.667 |  81.556 |
| 1  | 1  | 1  | 1  |  1   |  1   |  1   |   1    | 201 | 218 | 220 |  213.0  |  72.667 |
+----+----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+

Нормована матриця
+----+-----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| X0 |  X1 | X2 | X3 | X1X2 | X1X3 | X2X3 | X1X2X3 |  Y1 |  Y2 |  Y3 |    Y    |   S^2   |
+----+-----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+
| 1  | -20 | 5  | 5  | -100 | -100 |  25  |  -500  | 224 | 211 | 209 | 214.667 |  44.222 |
| 1  | -20 | 40 | 10 | -800 | -200 | 400  | -8000  | 197 | 196 | 213 |  202.0  |  60.667 |
| 1  |  30 | 5  | 10 | 150  | 300  |  50  |  1500  | 201 | 219 | 206 | 208.667 |  57.556 |
| 1  |  30 | 40 | 5  | 1200 | 150  | 200  |  6000  | 214 | 221 | 197 | 210.667 | 101.556 |
| 1  | -20 | 5  | 10 | -100 | -200 |  50  | -1000  | 201 | 215 | 209 | 208.333 |  32.889 |
| 1  | -20 | 40 | 5  | -800 | -100 | 200  | -4000  | 209 | 207 | 222 | 212.667 |  44.222 |
| 1  |  30 | 5  | 5  | 150  | 150  |  25  |  750   | 225 | 203 | 216 | 214.667 |  81.556 |
| 1  |  30 | 40 | 10 | 1200 | 300  | 400  | 12000  | 201 | 218 | 220 |  213.0  |  72.667 |
+----+-----+----+----+------+------+------+--------+-----+-----+-----+---------+---------+

Критерій Кохрена
Дисперсія однорідна!

Критерій Стьюдента
t0=210.5835
t1=1.1667499999999968
t2=-1.0
t3=-2.583500000000001
t4=1.0832499999999996
t5=1.6667500000000004
t6=0.5000000000000036
t7=1.583249999999989

Рівняння регресії з коефіцієнтами від нормованих значень факторів
y = 210.5835 + 1.16675*x1 + -1.0*x2 + -2.5835*x3 + 1.08325*x1x2 + 1.66675*x1x3 + 0.5*x2x3 + 1.58325*x1x2x3
Рівняння регресії з коефіцієнтами від натуральних значень факторів
y = 221.37217 + 0.03522*x1 + -0.10095*x2 + -1.26103*x3 + -0.00838*x1x2 + -0.0059*x1x3 + 0.00419*x2x3 + 0.00145*x1x2x3

Критерій Фішера
Рівняння регресії неадекватне.
```